МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Ростовский государственный экономический университет (РИНХ)»

На правах рукописи

Джанунц Гарик Апетович

МЕТОДЫ ОБРАБОТКИ ДАННЫХ В ИНФОРМАЦИОННО-ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ ДЛЯ МОДЕЛЕЙ ПЕРИОДИЧЕСКИХ ПРОЦЕССОВ

Специальность: 2.3.8 – Информатика и информационные процессы

Диссертация на соискание ученой степени

доктора технических наук

Научный консультант: доктор технических наук,
профессор Ромм Я.Е.

Таганрог – 2023

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность проблемы. Разработка и исследование высокоточных быстродействующих методов обработки данных в информационных вычислительных системах (ИВС) актуальны для моделей многих реальных процессов. В частности это относится к моделям периодических процессов, включая модели периодических автоколебательных реакций, модели процессов переноса и модели прогнозирования движения космических аппаратов с периодической орбитой. Создание таких методов остро актуально для глобальных навигационный спутниковых систем. Точность и скорость расчета пространственных данных об объектах имеет важное значение, в частности, для геоинформационных систем (ГИС). Глобальная навигационная спутниковая система (ГЛОНАСС) предназначена для определения местоположения, скорости движения, точного времени морских, воздушных, сухопутных и космических потребителей, а также для выполнения дополнительных информационных функций. При определении местоположения в навигационном приемнике текущее положение навигационного космического аппарата (НКА) прогнозируется на основе обработки данных оперативной эфемеридной информации из навигационного сообщения с использованием модели периодического возмущенного движения НКА ГЛОНАСС на околоземной орбите. Модель движения НКА представляет собой систему обыкновенных дифференциальных уравнений (ОДУ). Точность и скорость прогноза параметров движения НКА существенно зависит от выбора метода обработки данных дифференциальной модели. В диссертационном исследовании вопросу выбора наиболее точного и быстродействующего метода обработки данных уделяется первоочередное внимание.

ОДУ являются теоретической основой моделирования многих процессов, представляющих собой объекты актуальных научно-технических исследований. Такие процессы известны в механике, физике, химии, биологии, экономике, а также в других областях науки, техники и технологии,

включающих фундаментальные и прикладные исследования. Приближенное решение ОДУ актуально в применении к математическому, численному и компьютерному моделированию процессов. Хорошо известно применение ОДУ для моделирования динамических процессов в механике, планетной астрономии, астрофизике, теории автоматического управления, включая управление движением и стабилизацией космических аппаратов, движением роботов и коррекцией движения объектов к заданной цели. В целом, ОДУ составляет одну из основ математического моделирования объективной реальности. Исследование процессов гидродинамики и газодинамики выполняется с применением моделей на основе дифференциальных уравнений (ДУ) в частных производных, наряду с тем можно говорить о моделировании с помощью интегро-дифференциальных уравнений. В этом аспекте можно принять во внимание модели гидродинамических, волновых процессов на основе систем линейных и квазилинейных гиперболических уравнений. Относительно универсальным инструментом исследования данных моделей является численное моделирование. При этом актуально применение численных методов, характеризующихся высокой точностью приближения и одновременно возможностью минимизации временной сложности (как, например, в планетной астрономии и при решении жестких задач). Такие противоречивые требования достаточно типичны для применения современных средств вычислительной техники. Повышение точности приближения численных методов обычно достигается ценой уменьшения шага метода и дополнительными операциями, обеспечивающими снижение погрешности математическими приемами. Как следствие, время решения задачи может возрастать с превышением допустимых ограничений. Помимо того, рост времени решения влечет дополнительное накопление погрешности, связанной с искажением численного метода в процессе компьютерной реализации операций с плавающей точкой. В том числе и по этой причине, несмотря на неуклонный рост мощности современной вычислительной техники по параметрам

быстродействия и памяти, одним из наиболее важных требований к численным методам решения ДУ является минимизация количества операций, включая число обращений к правой части уравнений [1, 2].

Противоречивость практических требований в аспекте точности и быстродействия фактически исключает выбор универсального метода и требует разработки широкого класса методов обработки данных дифференциальных моделей, имеющих, как правило, проблемно ориентированный или специализированный характер [3]. Актуальные тенденции при построении означенных методов также связаны с расширением их возможностей для решения жестких систем высокой размерности [4]. Вместе с тем минимизация временной сложности актуальна и для решения жестких систем малой размерности. Например, при моделировании периодической химической реакции Белоусова-Жаботинского для достижения высокой точности приближения при использовании явных методов Рунге-Кутты может неприемлемо возрастать время решения по причине требуемой малости шага [3]. Для решения задач небесной механики с учетом развития астрономических средств наблюдения создаются специальные высокоточные методы численного интегрирования, основанные на итерационном решении ОДУ с помощью полиномиальной аппроксимации [5]. Данный подход получил обобщение для решения систем ОДУ в общем виде, исследовался в работах [6, 7], в случае полиномиальной правой части – [8, 9], численное применение полиномов Тейлора для кусочно-полиномиального решения – [10, 11]. На основе первообразной от интерполяционного полинома Ньютона построен классический метод Адамса, однако постоянные коэффициенты вычислены перед конечными разностями, что определяет его разностную форму [12].

На эффективность численного интегрирования часто влияет форма представления вычислительной схемы и ее программная реализация. Так, благодаря оригинальному представлению вычислительной схемы, интегратор Эверхарта, построенный на основе неявных методов Рунге-Кутты, с точки

зрения численного интегрирования имеет ряд преимуществ [13, 14]. Близкие идеи приводятся в работе [15], содержащей оригинальный подход к решению проблемы получения коэффициентов для алгебраического многочлена. В частности, выбор разбиения шага интегрирования предлагается с помощью узлов квадратурной формулы Маркова. Новая программная реализация интегратора Эверхарта, предложенная в [16], позволила повысить эффективностьметода для решения задач небесной механики. К идеям этих работ примыкают работы [17, 18], в которых авторы предлагают реализацию подхода на основе приближения решения и его производной частичными суммами смещенных рядов Чебышева. При этом, коэффициенты рядов определяются с помощью итерационного процесса с применением квадратурной формулы Маркова.

Означенные проблемы актуальны также при численном решении краевых задач, например, в приложении к управлению движением объекта к заданной цели, когда объект может отклониться от цели в случае высокой погрешности численного решения задачи. Снижение погрешности обычно достигается ценой уменьшения шага конечно-разностных схем [19, 20] с итерационной «пристрелкой».

В сходной ситуации находится моделирование периодических процессов на основе уравнений в частных производных, где основу для высокоточной обработки данных модели составляют разностные схемы высоких порядков. Подробные исследования проблемы снижения погрешности приближенного решения с классическим изложением алгоритмов построения разностных схем проведены в монографиях [21, 22]. Характерной особенностью широкого класса приближенных методов, разработанных для решения ДУ в частных производных, является замена частных производных конечно-разностными отношениями. Такие методы дают дискретное приближение решения, привязанное к равномерной или неравномерной конечно-разностной сетке, покрывающей область решения. Иногда требуемые значения приближения в

промежуточных точках области интегрирования интерполируются, в частности, для повышения качества визуализации данных численного моделирования. Продвижение в аспекте точности приближений связывают с повышением порядка разностных схем, с методом конечных элементов [23], а также с его адаптацией к конкретному классу задач [24, 25]. Однако, если методы инвариантны, как, например, реализованные в информационно-вычислительных системах компьютерной математики, в частности, *MathCAD* или *Maple*, то часто не позволяют строить высокоточные приближения даже в случае одномерных уравнений с гладкими решениями. Так, решение уравнения переноса с гладкими начальными данными с применением методов, реализованных в означенных системах, приближается с границей абсолютной погрешности порядка 10^{-8} – 10^{-3} в прямоугольной области единичного размера. При расширении области из-за быстрого накопления погрешности разностной схемы в ряде примеров может не достигаться даже грубое приближение [26]. Разновидности существующих подходов используют интерполяцию [27, 28], адаптацию разностной сетки к решению, учитывают степень заполненности ячеек. Специфика компьютеризации современных схем приближенного решения ДУ в частных производных обсуждается в работах [29 – 31]. Однако границы погрешности применяемых методов зачастую остаются в отмеченном диапазоне [32, 33].

Таким образом, проблема обработки данных для дифференциальных моделей периодических процессов с высокой точностью и одновременно с минимизацией временной сложности является актуальной. Следует отметить, что для моделирования существенны проблемы непрерывности приближения данных, в частности, актуальна задача оптимизации распределения узлов интегрирования в зависимости от гладкости моделируемых данных на различных участках области интегрирования.

В целом, говоря об обосновании постановки задач и методов, предложенных в диссертации, необходимо отметить следующее. Хорошо

известно, что при решении задачи Коши разностные методы неизбежно накапливают погрешность с ростом области приближенного решения. В особенности, это сказывается в случае неустойчивых по Ляпунову решений задачи Коши, при решении жестких задач, в случае большого промежутка (больших размеров области) приближенного решения, как это часто требуется в математических моделях планетной астрономии, в моделях астрофизических процессов, в моделях управления движущимися объектами, в частности, спутниками и летательными аппаратами. При этом даже если входные данные и начальные условия нельзя задать с высокой точностью по причинам неточности измерений, накопление погрешности численного метода в большой области может повлечь недопустимое значение приближенного решения на выходе моделируемого процесса. Банальным примером этой трудности может служить решение задачи $y' = y$, $y(0) = 1$, которое задает экспоненту и является неустойчивым в смысле Ляпунова. В рассматриваемом случае численный метод необходимо инициирует возмущение решения, и его приближение независимо от метода на сколь угодно большую величину отклоняется от точного решения на достаточно большом промежутке приближения. Иногда аналогичные примеры интерпретируются как примеры жестких задач. Проблема устойчивости по Ляпунову актуальна для большинства задач математического, в частности, численного моделирования. Отсюда актуальна разработка компьютерного метода, обладающего малым накоплением погрешности на большом интервале (в большой области) приближенного решения. В диссертации такая задача ставится с дополнительным требованием минимизации временной сложности решения. Под временной сложностью здесь и в дальнейшем понимается количество последовательных шагов вычислительного алгоритма без учета межпроцессорного обмена и обмена с памятью вычислительной системы. В диссертации, помимо того, традиционно учитывается число обращений метода к правой части дифференциальной системы.

Основной подход к разработке и исследованию искомых вычислительных алгоритмов опирается на кусочную интерполяцию. Здесь также необходимы специальные пояснения. Во-первых, снижение роста погрешности будет достигаться за счет малости текущего подынтервала (подобласти), на котором вычисляется функция, интерполируется правая часть дифференциальной системы и приближается решение. Существенно, что снижение погрешности достигается не за счет роста степени полинома. Во-вторых, выбраны наиболее простые полиномы Лагранжа и Ньютона. Ясно, что сами по себе многие классические интерполяционные полиномы по аналитическим оценкам погрешности интерполяции зачастую точнее выбранных полиномов. Однако это реализуется за счет дополнительных вычислительных операций и специального распределения узлов интерполяции. На малом подынтервале дополнительные операции означают дополнительные вычисления, часто сложного характера. В компьютерной реализации это происходит с дополнительным накоплением погрешности, и аналитическое преимущество может оказаться потерянным. В-третьих, для выбранных полиномов применяется перевод из известной их формы в форму алгебраического полинома с числовыми коэффициентами. Это оказывается возможным выполнить с простым и универсальным в компьютерной реализации видоизменением формул Виета. В итоге, количество вычислительных операций на подынтервале (в подобласти) оказывается минимизированным, эти операции не влекут вычислительной сложности, и не происходит накопления дополнительной погрешности на малом подынтервале. Поэтому последующее применение модифицированной таким образом интерполяции делает накопление погрешности в наибольшей мере зависимым только от малости подынтервала, а не от степени полинома.

Диссертация посвящена разработке и исследованию высокоточных быстродействующих методов обработки данных в ИВС для моделей периодических процессов. При этом на основе интерполяционных полиномов с

итерационным уточнением алгоритмически достигается и программно реализуется гладкость аналитического приближения данных на отрезке произвольной длины. При разработке требуется, чтобы искомые методы были основаны на инвариантном вычислительном алгоритме для широкого класса моделей периодических процессов в ИВС, в числе которых модели жестких задач и задач с неустойчивыми по Ляпунову решениями. Аналогичное требование распространяется на случай кусочно-интерполяционной обработки числовых данных модели переноса с целью компьютерной реализации методов с помощью по возможности единых программных процедур для создания инвариантного комплекса программ обработки данных в ИВС. В частности, с помощью искомого комплекса программ требуется выполнять моделирование динамики химических и биохимических осцилляторов, автоколебаний в задачах радиоэлектроники, управляемого движения космических аппаратов (КА), устойчивого периодического движения КА на околоземной орбите, процесса переноса волны. С применением комплекса требуется кроме того выполнять моделирование движения НКА ГЛОНАСС по данным оперативной эфемеридной информации из навигационного сообщения для ускорения процесса прогнозирования координат и составляющих вектора скорости центра масс. При этом актуально осуществлять обработку данных эфемерид с наиболее высокой точностью в произвольно заданные моменты времени из интервала прогнозирования параметров движения НКА.

Согласно изложенному актуальна следующая цель диссертационного исследования.

Целью диссертационной работы является разработка и исследование высокоточных быстродействующих алгоритмов и программ обработки данных в ИВС для моделей периодических процессов, в числе которых периодические автоколебательные реакции, модели процессов переноса и модели прогнозирования движения космических аппаратов с периодической орбитой. В частности, для этого необходимо построение библиотеки стандартных

программ ИВС высокоточного вычисления функций и обработки данных интегральных моделей в режиме реального времени.

Для достижения поставленной цели в диссертационной работе решаются следующие **задачи**:

1. Разработать высокоточный метод разностно-полиномиальной обработки данных в ИВС с автоматизированным выбором варьируемых параметров и итерационным уточнением для моделей периодических процессов.

2. Исследовать сходимость, скорость сходимости и трудоемкость разрабатываемого метода разностно-полиномиальной обработки данных на временном промежутке произвольной длительности.

3. Разработать модификацию метода обработки данных в ИВС с автоматизированным выбором варьируемых параметров и итерационным уточнением для моделей периодических процессов, полностью исключающую использование разностных схем и достигающую повышенной точности обработки данных на больших временных промежутках.

4. Исследовать сходимость, скорость сходимости и трудоемкость разрабатываемой модификации кусочно-интерполяционного метода обработки данных с итерационным уточнением на временном промежутке произвольной длительности.

5. Построить программную реализацию метода с автоматизированным выбором параметров, адаптирующихся к структуре модели и реализующих динамическую коррекцию начальных данных для достижения наибольшей точности при наименьшем времени обработки.

6. Разработать метод варьируемой кусочно-интерполяционной обработки числовых данных модели переноса с итерационным уточнением с целью получить аналитическое приближение данных, характеризующееся сравнительно высокой точностью и кусочной гладкостью.

7. Исследовать сходимость, скорость сходимости и трудоемкость разрабатываемой кусочно-интерполяционной обработки данных с итерационным уточнением для модели переноса в прямоугольной области.

8. Выполнить алгоритмизацию и программную реализацию обработки данных модели переноса с автоматическим выбором параметров, обеспечивающим наибольшую точность при наименьшем времени обработки.

9. Разработать метод создания библиотеки стандартных программ в ИВС на основе кусочно-интерполяционной обработки данных с хранимыми значениями полиномиальных коэффициентов, позволяющий параллельно воспроизводить значения стандартных и специальных функций, используемых в моделях периодических процессов, на произвольном множестве точек фиксированной области за время единичного порядка.

10. Разработать разновидность кусочно-интерполяционной обработки данных дифференциальной модели с повышенным быстродействием без потери точности обработки за счет пользовательского подбора и фиксирования параметров. На этой основе синтезировать быстродействующие алгоритмы высокоточного воспроизведения специальных функций и их производных, применяемых в моделях периодических процессов, с гладкостью приближения на больших временных промежутках.

11. Создать комплекс программ обработки данных в ИВС на основе разрабатываемого метода для моделей жестких и нежестких задач, включающий возможность адаптации к различным классам моделей. Комплекс также должен включать программы для кусочно-интерполяционной обработки данных модели переноса. На основе комплекса выполнить моделирование различных периодических автоколебательных реакций, получить уточнение физико-химических параметров автоколебаний и повысить качество моделирования процессов за счет высокой точности и гладкости обработки данных моделей в допустимых границах трудоемкости.

12. С помощью разработанного программного комплекса выполнить моделирование в ИВС управляемого движения КА, устойчивого периодического движения КА на околоземной орбите с целью прогнозирования положения КА. Представить уточненные результаты моделирования, необходимые для отладки технологических процессов, для эффективного управления движением КА в режиме реального времени, для расчета координат КА в произвольно заданные моменты времени.

13. С помощью разработанного комплекса выполнить моделирование движения НКА ГЛОНАСС по данным эфемерид. На этой основе синтезировать алгоритмы, применение которых существенно ускорит процесс расчета координат и составляющих вектора скорости центра масс НКА ГЛОНАСС с превышением требуемой точности в произвольно заданные моменты времени интервала прогнозирования.

14. Разработать алгоритмы и программы, которые позволят сохранять гладкое аналитическое приближение компонентов траектории и скорости движения НКА ГЛОНАСС в памяти компьютера и восстанавливать его без повторного вычисления траектории в произвольной точке интервала прогнозирования за время единичного порядка без снижения точности приближений.

15. С целью интегральной обработки данных на основе использования кусочно-интерполяционного метода обработки данных получить инвариантные относительно степени полинома аналоги формул Ньютона-Котеса с коэффициентами, не зависящими от подынтегральной функции и промежутка интегрирования. На этой основе разработать и программно реализовать методы высокоточного вычисления интегралов и первообразных при условии минимизации времени вычисления. Исследовать сходимость, скорость сходимости и трудоемкость методов, представить таблицы хранимых коэффициентов для стандартизации программ.

Научная новизна результатов диссертационной работы заключается в следующем:

1. Предложен метод разностно-полиномиальной обработки данных в ИВС с программным выбором варьируемых параметров для моделей периодических процессов. Метод отличается от аналогов обработкой данных на временных подынтервалах интерполяционными полиномами Ньютона, программно преобразуемыми в форму алгебраических полиномов с числовыми коэффициентами, разностной обработкой узловых значений, применением итерационного уточнения, автоматизированным выбором параметров, что позволяет повысить точность и уменьшить время обработки данных относительно известных методов, а также улучшить качество моделирования исследуемых процессов.

2. Выполнено исследование, показана сходимость, дана оценка скорости сходимости предложенного метода. Для корректного применения метода достаточно двукратной дифференцируемости правой части дифференциальной системы, что положительно отличается от условий применения аналогов и улучшает качество моделирования в ИВС периодических процессов с быстро меняющейся динамикой.

3. Как развитие метода обработки данных в ИВС с автоматизированным выбором варьируемых параметров и итерационным уточнением для моделей периодических процессов предложена модификация, не использующая разностные схемы. Модифицированный метод отличается от аналогов кусочной интерполяцией на временных подынтервалах правой части дифференциальной системы и интегральным приближением решения в виде алгебраических полиномов с числовыми коэффициентами, а также выполнением итераций на подынтервалах, аналогичных интегральным приближениям Пикара. Метод отличается повышением точности обработки данных на больших отрезках времени относительно известных методов, а также улучшением качества численного моделирования.

4. Показана сходимость предложенной модификации кусочно-интерполяционного метода обработки данных и сходимость итерационного уточнения, даны оценки скорости сходимости. Качества равномерной сходимости и аналитический вид приближения числовых данных отличают предложенную модификацию от известных аналогов.

5. Предложенный метод реализован в виде стандартной программы ИВС, на вход которой поступает правая часть дифференциальной системы, моделирующей процесс, начальные данные и параметры, включающие границы временного отрезка обработки данных модели. Автоматический выбор параметров обеспечивает наибольшую точность при наименьшем времени обработки. Это отличительное качество программной реализации позволяет превышать точность аналогов на два десятичных порядка, при этом варьируемые параметры программно адаптируются к структуре модели и реализуют динамическую коррекцию начальных данных. В результате достигается вычислительная устойчивость, высокая точность, гладкость аналитического приближения данных на отрезке произвольной длины, что составляет отличие метода для широкого класса моделей периодических процессов в ИВС, в числе которых модели жестких задач и задач с неустойчивыми по Ляпунову решениями.

6. Предложен аналог метода обработки данных в ИВС для моделей с частными производными. Более точно, разработана варьируемая кусочно-интерполяционная обработка числовых данных модели переноса, которая отличается от известных применением интерполяционного полинома Ньютона от двух переменных, программно преобразуемого в алгебраический полином с числовыми коэффициентами. Варьируемая интерполяция данных выполняется в каждой прямоугольной подобласти, на которые в зависимости от значений параметров автоматически делится исходная прямоугольная область. Применяется итерационное уточнение обработанных данных. На этой основе получается гладкое приближение данных в прямоугольной подобласти, которое

отличает предложенный метод от известных, кроме того, метод отличается значительно более высокой точностью. Согласно численному эксперименту достигается превышение точности известных методов моделирования переноса на несколько десятичных порядков, что позволяет улучшить качество моделирования волновых процессов в ИВС.

7. Показана сходимость кусочно-интерполяционной обработки данных модели переноса, даны оценки скорости сходимости. Для случая прямоугольной области оценивается скорость сходимости итерационного уточнения. Для квазилинейной модели переноса предложена схема вывода аналогичных оценок.

8. Выполнена алгоритмизация и программная реализация обработки данных модели переноса, реализован автоматический выбор параметров, обеспечивающий наибольшую точность при наименьшем времени обработки. Программная реализация отличается устойчивостью и отмеченной точностью обработки, что в сочетании с кусочной гладкостью приближения данных позволяет детализировать и повысить качество моделирования процесса переноса.

9. Предложен метод создания библиотеки стандартных программ в ИВС на основе кусочно-интерполяционной обработки данных. Разновидность кусочной интерполяции позволяет приближать функции с точностью до 10^{-20} на произвольном временном отрезке, при этом стандартные и специальные функции приближаются за время единичного порядка, что положительно отличает метод от известных. Реализованы варианты хранения коэффициентов в разделе констант программы, в типизированном файле, в постоянном запоминающем устройстве, дан алгоритм считывания. Вычисления функции взаимно независимы по значениям аргумента, что влечет параллелизм метода. На основе хранимых коэффициентов любая функция библиотеки может параллельно воспроизводиться на произвольном множестве точек

фиксированной области. На аналогичной основе разработано приближение производных в ИВС с точностью до 10^{-16}.

10. Показана возможность существенного снижения трудоемкости без потери точности в предложенном методе обработки данных дифференциальной модели при замене автоматического выбора параметров их пользовательским подбором и фиксированием. Согласно численному и программному эксперименту предложенная модификация превосходит известные методы по быстродействию, при этом достигает более высокой точности обработки данных. На этой основе разработаны быстродействующие алгоритмы воспроизведения с помощью хранимых коэффициентов специальных и стандартных функций с гладкостью приближения. В частности, функция Бесселя и гипергеометрическая функция, применяемые в моделях периодических процессов, реализуются гладким приближением на больших временных промежутках с быстродействием и точностью, превосходящими характеристики известных аналогов.

11. Разработан комплекс программ обработки данных в ИВС на основе предложенного метода для моделей жестких и нежестких задач, включающий автоматический и пользовательский выбор параметров для адаптации к классам моделей. С помощью комплекса выполнено моделирование периодических автоколебательных реакций (реакции Белоусова-Жаботинского, релаксационных автоколебаний в системе гликолиза, гетерогенной колебательной реакции окисления молекулярного водорода). Результаты отличаются сравнительно высокой точностью и гладкостью обработки данных моделей в допустимых границах трудоемкости, что позволяет уточнить физико-химические параметры автоколебаний и повысить качество моделирования. Комплекс включает программы моделирования процессов переноса, согласно эксперименту точность обработки данных с помощью этих программ по сравнению с аналогами повышается в среднем на восемь десятичных порядков,

отличается на порядок меньшим временем обработки и гладкостью приближения, что позволяет уточнить форму и динамику перемещения волн.

12. С помощью разработанного комплекса программ выполнено моделирование в ИВС движения КА, рассчитано уточненное время необходимое для вывода КА на устойчивую периодическую орбиту при управлении, соответствующем внешнему воздействию гравитационных сил. Получены уточненные параметры орбиты устойчивого движения КА. Достаточно малая временная сложность предложенной обработки данных позволяет применять метод для управления движением КА в режиме реального времени. Выполнено моделирование возмущенного движения искусственного спутника Земли (ИСЗ), выведенного на низкую околоземную орбиту. Сравнительно высокая точность предложенного метода обработки данных в сочетании с гладкостью и малой временной сложностью дают координаты ИСЗ с требуемой точностью в произвольный момент времени, что необходимо для измерений с помощью лазерного дальномера от пункта наблюдения до ИСЗ и позволяет предупреждать аварийный сход с орбиты.

13. С помощью разработанного программного комплекса выполнена обработка данных эфемерид на модели движения НКА ГЛОНАСС. Точность и гладкость предложенного метода обработки данных позволяют получить координаты и составляющие вектора скорости центра масс НКА с превышением границ требуемой точности в произвольно заданные моменты времени из 30-минутного интервала прогнозирования, при этом время расчета примерно вдвое меньше времени известных методов. Процесс расчета реализован в виде процедуры с параметрами определяемыми из навигационного сообщения и может быть полностью автоматизирован с сохранением быстродействия, точности и гладкости обработки данных. Предложен способ дополнительного ускорения обработки данных на модели движения НКА ГЛОНАСС за счет хранения коэффициентов кусочной

интерполяции траектории и динамики движения для задач постобработки данных.

14. Кусочно-интерполяционное приближение координат непрерывно, сходится к траектории НКА ГЛОНАСС и приближает скорость НКА на всем интервале прогнозирования. На каждом подынтервале интервала прогнозирования интерполирующий компонент траектории полином имеет вид алгебраического полинома фиксированной степени с числовыми коэффициентами. Это позволяет сохранять в памяти компьютера приближение компонента траектории в виде типизированного файла коэффициентов полиномов соответственных подынтервалам разбиения. Непрерывное и гладкое приближение каждого компонента траектории оказывается хранимым и восстанавливается без повторного вычисления траектории в произвольной точке интервала прогнозирования простым алгоритмом считывания как соответственное значение полинома, что качественно отличается от известных методов. На данной основе процесс прогнозирования естественным образом архивируется. Создаются предпосылки прогнозирования не отдельных точек траектории, а всего непрерывного отрезка траектории движения НКА с расширением интервала прогнозирования. Приводится программа обработки данных и результаты численного эксперимента, подтверждающие улучшение качества прогнозирования параметров движения НКА.

15. Обработка данных использует многообразие методов, в числе которых вычисление интеграла. Разработанный кусочно-интерполяционный метод используется для приближения подынтегральной функции. Интерполяция выполняется с помощью полиномов Лагранжа и Ньютона, преобразуемых в форму алгебраических полиномов с числовыми коэффициентами. Полученный полином интегрируется, приводя к инвариантным относительно степени полинома формулам Ньютона-Котеса. Коэффициенты формул не зависят от подынтегральной функции, промежутка интегрирования, хранятся в разделе констант программы. Пользовательский интерфейс программ

стандартизируется до задания подынтегральной функции, промежутка интегрирования, как вариант может включать степень полинома и число подынтервалов. Показана сходимость метода, даны оценки скорости сходимости, представлены таблицы коэффициентов для стандартизации программ. Приведены коды программ и результаты эксперимента, согласно которым на промежутке длины 500 достигается граница погрешности приближения интеграла порядка 10^{-20}. На промежутках стандартной для обработки данных длины достигается нулевая граница погрешности. Метод распространяется на приближение первообразной функции, в этом случае погрешность имеет порядок 10^{-19}. Одновременно с минимизацией погрешности минимизируется время вычисления интегралов и первообразных, что отличает метод от известных.

Научная и практическая значимость

Научную и практическую значимость в области обработки данных в ИВС моделей периодических процессов имеют полученные в диссертации высокоточные быстродействующие методы и программная реализация обработки данных дифференциальных моделей, включая модели периодических автоколебательных реакций, модели процессов переноса и модели прогнозирования движения космических аппаратов. Научную значимость в области численных методов имеют представленные в диссертации кусочно-интерполяционные методы приближенного решения дифференциальных систем, отличающиеся от известных способом построения, инвариантностью вычислительного алгоритма относительно задач различных классов, возможностью достижения высокой точности и гладкости полученного приближения при одновременной минимизации временной сложности. Значимы также представленные в работе методы варьируемого кусочно-интерполяционного вычисления функций, отличающиеся малой погрешностью и гладкостью непрерывного приближения с минимизацией временной сложности, и аналог подхода Ньютона-Котеса для произвольной

степени интерполяционного полинома, реализованный на основе кусочно-интерполяционного приближения подынтегральных функций. Помимо того, научно значимым является метод кусочно-интерполяционного решения уравнения переноса на основе интерполяционного полинома Ньютона от функции двух переменных варьируемой степени с итерационным уточнением. Метод позволяет достигать высокой точности и кусочной непрерывности приближения решения линейного и квазилинейного уравнения переноса и уточнить на этой основе параметры волнового процесса в результате обработки данных модели переноса. Все предложенные методы основаны на инвариантном программном переводе интерполяционных полиномов в форму алгебраических полиномов с числовыми коэффициентами с помощью алгоритма отличного от формул Виета и уравнений Ньютона для симметрических функций.

Практическая значимость диссертационного исследования заключается в прикладном характере предложенных методов обработки данных, включающих кусочно-интерполяционные решения ОДУ и ДУ в частных производных, которые применяются для компьютерной реализации моделей колебательных динамических процессов, включая химические и биохимические реакции (реакция Белоусова-Жаботинского, релаксационные автоколебания в системе гликолиза, колебательная реакция окисления молекулярного водорода), электрическое равновесие автогенератора с внутренней обратной связью, движение КА с управлением при помощи «малой тяги», возмущенное движение КА, прогнозирование положения НКА системы ГЛОНАСС, а также для моделирования переноса волны. Результаты моделирования необходимы для отладки технологических процессов на основе периодических реакций, для эффективного управления движением КА в режиме реального времени, для прогнозирования движения и расчета координат ИСЗ в произвольно заданные моменты времени, повышения точности и скорости расчета пространственных данных об объектах для ГИС. На основе инвариантных кусочно-

интерполяционных методов разработан программный комплекс, реализующий методы как с автоматическим, так и с пользовательским подбором параметров. Комплекс программ предназначен для решения актуальных задач обработки данных моделей, связанных с исследованием динамических систем и автоколебательных процессов. В аспекте практического применения существенно сочетание в инвариантных кусочно-интерполяционных методах высокой точности, быстродействия и гладкости приближения, позволяющее применять методы для повышения качества численного моделирования, в том числе в режиме реального времени. Данное сочетание характеристик сохраняется при численном моделировании процессов, описываемых жесткими и нежесткими системами ОДУ, и в целом характеризует разработанный комплекс программ. Практическую значимость имеет в частности то, что точность и гладкость предложенного метода обработки данных позволяют получить координаты и составляющие вектора скорости центра масс НКА ГЛОНАСС с превышением требуемой точности в произвольно заданные моменты времени из 30-минутного интервала прогнозирования, при этом время расчета примерно вдвое меньше времени известных методов. Кроме того, значима возможность сохранять в памяти компьютера приближение компонента траектории в виде типизированного файла. Хранимым оказывается непрерывное и гладкое приближение каждого компонента траектории, при этом траектория восстанавливается без повторного вычисления в произвольной точке интервала прогнозирования. Процесс прогнозирования естественным образом архивируется, создаются предпосылки прогнозирования всего непрерывного отрезка траектории движения НКА с увеличением существующего интервала прогнозирования.

Внедрение и использование результатов работы. Полученные в работе результаты приняты к использованию:

1. В АО НКБ ВС для моделирования движения и автоматического управления подвижными объектами; для повышения точности численного

моделирования автоколебательных процессов при автоматизированном управлении движением объектов; при расширении библиотеки стандартных программ вычисления элементарных, повторяющихся и специальных функций для бортовых вычислителей систем автоматического управления движением подвижными объектами и с целью вычисления композиций сложных функций с минимальной временной сложностью при высокой точности приближения функций. Разработанный программный комплекс используется для численного моделирования процессов управления с широко варьируемыми параметрами с целью уточнения физико-технологических и динамических характеристик моделируемых процессов.

2. В АО «ВНИИЖТ» для повышения точности прогнозирования положения объекта, а также для моделирования навигационного управления; для расширения библиотеки стандартных программ вычисления элементарных, часто повторяющихся и специальных функций с целью ускорения процесса численного моделирования, а также уточнения значения параметров в моделях дистанционного зондирования с помощью спутников и БПЛА. Разработанный программный комплекс принят к использованию с целью уточнения физико-технологических характеристик моделируемых процессов в режиме реального времени; программный комплекс для численного моделирования движения навигационных космических аппаратов ГЛОНАСС принят к использованию с целью ускорения и повышения точности расчета координат местоположения и скорости движения исследуемых объектов.

3. В учебном процессе кафедры информатики Таганрогского института имени А.П. Чехова (филиала) ФГБОУ ВО «РГЭУ (РИНХ)» в курсах «Специальные разделы информатики», «Современные инструментальные средства», «Программирование», «Компьютерное моделирование», «Алгоритмы численного интегрирования и анализа устойчивости», «Современные технологии программирования», «Абстрактная и компьютерная алгебра», «Численные методы в анализе данных», «Визуализация данных»,

«Математическое и имитационное моделирование», «Численные методы», «Математическое моделирование и численные эксперименты», «Современные методы построения программ» и «Параллельные алгоритмы».

Кроме того, полученные в работе результаты были положены в основу исследований по проектам, поддержанным Российским фондом фундаментальных исследований: 10-07-00178-а «Численная оптимизация на основе сортировки с приложением к анализу устойчивости, поиску и распознаванию»; 12-07-00143-а «Компьютерные методы численной оптимизации на основе сортировки с приложением к анализу устойчивости, разностно-полиномиальному решению дифференциальных уравнений, распознаванию изображений и цифровой обработке сигналов»; 16-07-00100-А «Компьютерные методы варьируемого кусочно-полиномиального решения дифференциальных уравнений и анализа устойчивости».

Методы исследования включают методы обработки данных в ИВС моделей периодических процессов, вычислительные методы алгебры и математического анализа, методы интерполяции и численного анализа, элементы теории сложности алгоритмов, методы прогнозирования движения космических объектов, методы объектного и прикладного программирования.

Положения, выносимые на защиту:

1. Метод разностно-полиномиальной обработки данных в ИВС с программным выбором варьируемых параметров для моделей периодических процессов, построенный на основе обработки данных на временных подынтервалах интерполяционными полиномами Ньютона, программно преобразуемыми в форму алгебраических полиномов с числовыми коэффициентами. Метод использует разностную обработку узловых значений, применяет итерационное уточнение и обеспечивает повышение точности при уменьшении времени обработки данных для улучшения качества моделирования исследуемых процессов.

2. Обоснование сходимости и оценки скорости сходимости предложенного метода разностно-полиномиальной обработки данных на произвольном промежутке времени.

3. Модификация метода обработки данных в ИВС с автоматизированным выбором варьируемых параметров для моделей периодических процессов, исключающая использование разностных схем. Модификация построена на основе кусочной интерполяции на временных подынтервалах правой части дифференциальной системы и интегральном приближении решения в виде алгебраических полиномов с числовыми коэффициентами, использует итерационные уточнения аналогичные интегральным приближениям Пикара. Модификация отличается повышенной точностью обработки данных на больших отрезках времени и позволяет улучшать качество моделирования.

4. Обоснование сходимости и оценки скорости сходимости предложенной модификации кусочно-интерполяционного метода обработки данных с итерационным уточнением на произвольном временном промежутке.

5. Программная реализация предложенного метода с автоматизированным выбором параметров, адаптирующихся к структуре модели и реализующих динамическую коррекцию начальных данных, что позволяет достигать наибольшей точности гладкого аналитического приближения данных на отрезке произвольной длины при минимальном времени обработки. При этом достигается вычислительная устойчивость обработки данных в ИВС для широкого класса моделей периодических процессов, в числе которых модели жестких задач и задач с неустойчивыми по Ляпунову решениями.

6. Метод варьируемой кусочно-интерполяционной обработки числовых данных модели переноса с итерационным уточнением, построенный на основе интерполяционного полинома Ньютона от двух переменных, программно преобразуемого в алгебраический полином с числовыми коэффициентами. Применяется двумерное итерационное уточнение обработанных данных, что

позволяет получить кусочно гладкое аналитическое приближение движения волны в прямоугольной области со сравнительно высокой точностью и повысить качество моделирования процесса переноса в ИВС.

7. Обоснование сходимости и оценки скорости сходимости кусочно-интерполяционной обработки данных с итерационным уточнением для модели переноса в прямоугольной области.

8. Алгоритмизация и программная реализация метода обработки данных модели переноса с автоматизированным выбором параметров, обеспечивающим наибольшую точность при наименьшем времени обработки и вычислительную устойчивость, что в сочетании с кусочной гладкостью приближения данных позволяет детализировать и повысить качество моделирования процесса переноса.

9. Метод создания библиотеки стандартных программ в ИВС на основе кусочно-интерполяционной обработки данных с хранением полиномиальных коэффициентов, позволяющий параллельно воспроизводить приближения стандартных и специальных функций в моделях периодических процессов с точностью порядка 10^{-20} на произвольном множестве точек фиксированной области за время единичного порядка.

10. Модификация кусочно-интерполяционной обработки данных дифференциальной модели, позволяющая существенно повысить быстродействие без потери точности обработки за счет замены автоматического выбора параметров их пользовательским подбором и фиксированием. Разработанные на этой основе с помощью хранимых коэффициентов быстродействующие алгоритмы высокоточного воспроизведения специальных функций и их производных позволяют в моделях периодических процессов получать приближения данных со свойством гладкости на больших интервалах времени.

11. Комплекс программ кусочно-интерполяционной обработки данных в ИВС для моделей жестких и нежестких задач, включающий автоматический и

пользовательский выбор параметров для адаптации к классам моделей. Комплекс позволяет выполнять быстродействующее высокоточное моделирование периодических автоколебательных реакций, что необходимо для отладки технологических процессов на их основе. Комплекс также включает моделирование процессов переноса для уточнения форм и динамики перемещения волн.

12. Комплекс программ для моделирования в ИВС движения КА, позволяющий рассчитывать время и параметры вывода КА на устойчивую периодическую орбиту при управлении соответствующем внешнему воздействию гравитационных сил. Сравнительно малая временная сложность предложенной обработки данных позволяет применять комплекс для управления движением КА в режиме реального времени, кроме того программный комплекс позволяет рассчитать с повышенной точностью координаты ИСЗ, выведенного на низкую околоземную орбиту, в произвольный момент времени, что необходимо для измерений с помощью лазерного дальномера от пункта наблюдения до ИСЗ.

13. Алгоритмы и программы для моделирования движения НКА ГЛОНАСС по данным эфемерид, позволяющие существенно ускорить процесс расчета координат и составляющих вектора скорости центра масс НКА ГЛОНАСС с превышением требуемой точности в произвольно заданные моменты времени из 30-минутного интервала прогнозирования. Предложенные алгоритмы и программы позволяют сохранять гладкое аналитическое приближение координат траектории и скорости движения НКА в памяти компьютера и восстанавливать его без повторного вычисления траектории в произвольной точке интервала прогнозирования за время единичного порядка без снижения точности приближений. На данной основе улучшается процесс прогнозирования и его архивация, создаются предпосылки прогнозирования непрерывного отрезка траектории движения НКА и расширения интервала прогнозирования.

14. Формулы обработки интегральных данных, представляющие собой разновидность формул Ньютона-Котеса с коэффициентами, не зависящими от подынтегральной функции и промежутка интегрирования, полученными на основе кусочной интерполяции. Вычисление интегралов с хранимыми в памяти ИВС коэффициентами реализовано программно. Пользовательский интерфейс программ стандартизирован до задания подынтегральной функции, промежутка интегрирования, как вариант может включать степень полинома и число подынтервалов. Интегральная обработка данных реализуется с высокой точностью на временных интервалах большой длины. На промежутках стандартной для обработки данных длины достигается нулевая граница погрешности. Одновременно с минимизацией погрешности минимизируется время вычисления интегралов.

Степень достоверности и апробация результатов работы. Достоверность полученных научных результатов вытекает из корректного математического обоснования с помощью аналитических оценок сходимости и погрешности приближений, а также временной сложности формализованных алгоритмов, подтверждается результатами численного и компьютерного моделирования, программного и вычислительного эксперимента.

Основные результаты работы были представлены на III Всероссийской студенческой научно-технической конференции «Прикладная информатика и математическое моделирование» (Москва, МГУП, 2009 г.); International Conference Parallel Computer Algebra ‘2010 (Tambov, Tambov State University named after G.R. Derzhavin, 2010); XI международной научно-практической конференции «Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности» (Санкт-Петербург, СПБПУ, 2011 г.); XII Всероссийском симпозиуме по прикладной и промышленной математике (весенняя сессия) (Казань, 2011 г.); Всероссийской НТК с международным участием: «Компьютерные и информационные технологии в науке, инженерии и управлении» «КомТех-2011» (Таганрог, ТТИ

ЮФУ, 2011 г.); Международной научно-практической конференции: «Перспективы развития технических наук» (Челябинск, 2015 г.); III Международной научно-практической конференции «Инновационные технологии и дидактика в обучении» (Таганрог, 2015 г.); III Всероссийской научно-практической конференции с международным участием «Аспекты развития науки, образования и модернизации промышленности». Технические науки (Таганрог, 2017 г.); Всероссийском форуме молодых ученых (Екатеринбург, 2017 г.); 16-ой Международной молодежной научно-практической конференции «Фундаментальные исследования, методы и алгоритмы прикладной математики в технике, медицине и экономике» (Новочеркасск, 2017 г.); IV Международной научной конференции «Актуальные проблемы прикладной математики» (Нальчик, 2018 г.); 17-ой Международной молодежной научно-практической конференции «Фундаментальные исследования, методы и алгоритмы прикладной математики в технике, медицине и экономике» (Новочеркасск, 2018 г.); Международной научной конференции «Актуальные проблемы прикладной математики, информатики и механики» (Воронеж, 2019 г.); IV Всероссийской научно-практической конференции «Информационные и инновационные технологии в науке и образовании» (Таганрог, 2019 г.); XVI Международной научно-технической конференции «Новые информационные технологии и системы» (Пенза, 2019 г.); Международной научной конференции «Актуальные проблемы прикладной математики, информатики и механики» (Воронеж, 2020 г.); на научных семинарах кафедры информатики Таганрогского института имени А.П. Чехова (филиала) ФГБОУ ВО «Ростовский государственный экономический университет (РИНХ)» (2010 – 2023 гг.).

Публикации. По материалам диссертации опубликовано 52 научные работы общим объемом около 82 печатных листов, включая одну монографию и 17 статей в ведущих рецензируемых научных журналах, входящих в Перечень ВАК при Минобрнауки России. Требованиям диссертационного

совета ЮФУ соответствует 15 опубликованных статей, из которых 5 научных работ опубликовано в российских и зарубежных изданиях, индексируемых в системах Web of Science, Scopus.

Соответствие паспорту специальности. Диссертационная работа соответствует пунктам 1, 3, 6, 8, 16 паспорта специальности 2.3.8. Информатика и информационные процессы (технические науки).

Структура и объем работы. Диссертационная работа состоит из введения, шести глав основного раздела, заключения, списка литературы и приложений к пяти главам. Основное содержание работы изложено на 386 страницах, включая список литературы из 206 наименований, приложение изложено на 106 страницах, включает коды программ, реализующих математические модели и предложенные методы обработки данных.

ГЛАВА 1. АНАЛИЗ ИЗВЕСТНЫХ МЕТОДОВ ОБРАБОТКИ ДАННЫХ ДЛЯ МОДЕЛЕЙ ПЕРИОДИЧЕСКИХ ПРОЦЕССОВ В ИНФОРМАЦИОННО-ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

В главе выполнен анализ современного состояния проблем обработки данных моделей периодических процессов в различных актуальных областях науки и техники. Основное внимание уделено процессам и методам обработки данных, к которым предъявляются требования высокой точности и одновременно возможно малой трудоемкости. Представлена детализация алгоритмов, наиболее распространенных при численном исследовании дифференциальных моделей, характеризующихся высокой степенью жесткости, в частности, при моделировании автоколебательных химических реакций, а также алгоритмов, обладающих высокой точностью при решении нежестких задач, с целью исследования особенностей и принципов их использования в соответствии со специфическими качествами моделирующих дифференциальных систем. Отмечены особенности обработки данных моделей в области астрофизики и планетной астрономии, затрудняющие искомое высокоточное моделирование на основе широко распространенных разностных методов и приводящие к необходимости разработки специализированных методов обработки данных. Аналогично, представлены примеры задач в области гидродинамики, при моделировании которых актуальны проблемы повышения точности обработки данных для моделей с частными производными, в частности, обработки данных модели переноса.

1.1. Современные методы обработки данных для моделей периодических процессов. Математические модели различных периодических процессов, представляющих собой объекты актуальных научно-технических исследований, описываются задачей Коши для системы ОДУ вида [2]

$$\frac{d\,\mathbf{y}}{d\,t} = \mathbf{f}\,(t,\,\mathbf{y})\,,\ \mathbf{y}\,(t_0) = \mathbf{y}_0\,, \tag{1.1}$$

где $\mathbf{y} = \left(y_1(t),\, y_2(t),\, \ldots,\, y_n(t)\right)$, $\mathbf{f}(t, \mathbf{y}) = \left(f_1(t,\, \mathbf{y}),\, f_2(t,\, \mathbf{y}),\, \ldots,\, f_n(t,\, \mathbf{y})\right)$. Применение приближенных аналитических способов решения (1.1), таких как метод медленно меняющихся амплитуд, малого параметра, гармонической линеаризации, функционального ряда Вольтерра, как правило, связано с существенными ограничениями по отношению к нелинейной части системы [34]. Относительно универсальным инструментом исследования таких моделей является численное моделирование с применением методов, характеризующихся достаточно высокой точностью и одновременно сравнительно малой трудоемкостью [2, 35, 36]. Ниже приводятся наиболее типичные и вместе с тем актуальные процессы из различных областей науки, при моделировании которых необходимо использование методов приближенного решения задачи (1.1), отличающихся высокой точностью и быстродействием.

1.1.1. Методы обработки данных в моделях образования космических пылевых облаков Кордылевского. Стремительное развитие астрономических средств наблюдения вызвало повышение точности и увеличение количества наблюдательной информации о движении небесных тел. В связи с этим возникает актуальная проблема разработки новых математических моделей, интерпретирующих наблюдательный материал. При этом «от численно реализуемых математических моделей требуется высокая эффективность, связанная в основном с точностью вычисляемых результатов и с быстродействием их получения» [37].

Так, исследование математической модели образования космических пылевых облаков Кордылевского, обнаруженных в 1961 году в окрестности точки либрации $L5$ системы Земля–Луна, показало возможность существования двух таких облаков, обращающихся вокруг точки либрации $L4$, и двух облаков, обращающихся вокруг $L5$ [38]. Исследование проведено на базе анализа устойчивого периодического движения в плоской ограниченной круговой задаче трех тел Земля–Луна–Частица с учетом возмущения от Солнца

в предположении, что орбиты Земли и Луны круговые и лежат в одной плоскости. Уравнения движения пробной материальной частицы P в подвижной системе координат $Oxyz$ в форме уравнений Лагранжа имеют вид [38]:

$$\left.\begin{aligned}
&\ddot{x} = 2\dot{y} + (1-\omega^2)x + 3\omega^2 \cos p\,(x\cos p - y\sin p) - (1-\mu)(x+\mu)r^{-3} + \\
&+ \mu(1-\mu-x)\ell^{-3}, \\
&\ddot{y} = -2\dot{x} + (1-\omega^2)y - 3\omega^2 \sin p\,(x\cos p - y\sin p) - (1-\mu)\,y\,r^{-3} - \\
&- \mu\,y\,\ell^{-3}, \\
&\ddot{z} = -(\omega^2 + (1-\mu)r^{-3} + \mu\,\ell^{-3})\,z,
\end{aligned}\right\} \qquad (1.2)$$

где $r = \sqrt{(x+\mu)^2 + y^2 + z^2}$ – расстояние от частицы до Земли, $\ell = \sqrt{(1-\mu-x)^2 + y^2 + z^2}$ – расстояние от частицы до Луны, ω – абсолютная орбитальная угловая скорость подвижной системы координат, $p = 2\pi(1-\omega)t$ – угол поворота подвижной системы координат относительно неподвижной, t_0 – момент полнолуния, $\dot{x}$ означает производную по времени. Значения для параметров приняты следующие: $\mu = 0.0122$, $\omega = 1/13.36$. Уравнения обладают инвариантной плоскостью $z = 0$. В [38] отсутствует информация об использованных при расчетах численных методах. На рис. 1.1 представлена траектория движения частицы, выпущенной из точки либрации $L_4(x_l = 0.5 - \mu,\, y_l = \sqrt{3}/2)$ с нулевой относительной скоростью, за время 24 синодических месяца. Траектория расчитана на основе решения задачи (1.2) методом Дормана-Принса 5 (4) [35]. Продолжительность синодического месяца – $29^d 12^h 44^m 8^s$, в принятых обозначениях – $2\pi/(1-\omega)$.

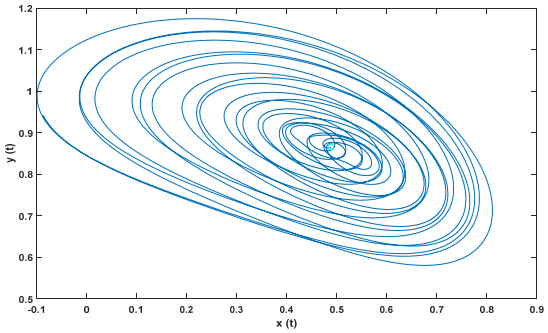

Рис. 1.1. Траектория движения частицы из точки либрации $L_4(0.5-\mu, \sqrt{3}/2)$ при $\dot{x}=0,\ \dot{y}=0$ на отрезке $[0,\ 48\pi/(1-\omega)]$ (метод Дормана-Принса 5 (4))

Вложенный метод Дормана-Принса 5 (4) выбран вследствие его высокой эффективности при невысоких требованиях к точности приближения [33]. Метод основан на вложенных формулах Рунге-Кутты и использует минимизацию остаточных членов погрешности для приближения старшего порядка точности. Кроме того, он не требует дополнительных вычислений функции правой части для интерполяции значений приближения между шагами сетки, что важно для визуализации результатов моделирования(требуемый краткий обзор численных методов решения задачи (1.1) представлен в п.1.2).

На рис. 1.2 траектория исследуемого авторами устойчивого по Ляпунову периодического решения с периодом $T=2\pi/(1-\omega)$ в плоскости $z=0$ воспроизводится с применением означенного метода Дормана-Принса 5-го порядка. Метод дает приближенное решение задачи (1.2) с погрешностью порядка 10^{-7}.

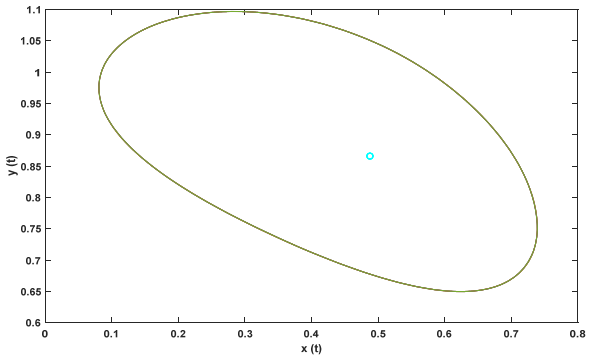

Рис. 1.2. Периодическая траектория задачи (1.2) с начальными условиями $(x_0,\ y_0,\ \dot{x}_0,\ \dot{y}_0)=(0.0848452,\ 0.094826895,\ -0.05135042,\ 0.1739387)$.

Использование высокоточных методов численного интегрирования задач небесной механики, таких как симплектический метод Йошиды [39], интегратор Эверхарта [13] или экстраполяционный метод Булирша-Штера [35, 40], в данном случае нецелесообразно, так как начальные данные представлены с точностью не превышающей 8 десятичных знаков. При численном исследовании авторы использовали метод невысокого порядка вследствие его сравнительно малой трудоемкости, так как многократное решение задачи (1.2) с целью нахождения начальных данных, обеспечивающих устойчивое по Ляпунову периодическое решение, весьма трудоемкий процесс для известных численных методов решения (1.1).

Представляется, что использование более точных методов, характеризующихся малой трудоемкостью, позволило бы уточнить модель и результаты моделирования. Уточненное решение могло бы выявить новые устойчивые периодические решения задачи (1.2) в окрестности точек либрации или получить начальные значения с более высокой точностью, не исключено, что при уточнении численного интегрирования устойчивых решений могло бы и не оказаться.

Таким образом, точность численного метода может напрямую влиять на результат моделирования и на трактовку его физического смысла.

1.1.2. Обработка данных в моделях замкнутых систем гравитирующих небесных тел. Главной проблемой «в решении задач долговременной орбитальной эволюции с использованием численных методов интегрирования является низкое быстродействие и потеря точности: последовательное определение состояний рассматриваемой динамической системы для ее адекватного представления требует малого шага интегрирования, тогда как каждое следующее состояние определяется со все большими ошибками, которые кроме того, могут усиливаться ляпуновской неустойчивостью. Орбитальная модель замкнутой системы гравитирующих небесных тел представляет собой совокупность осциллирующих динамических

подсистем» [37]. При этом, в частности, при использовании методов интегрирования высоких порядков гармоническая составляющая с наибольшей величиной частоты определяет шаг численного метода [41]. Пути повышения быстродействия и точности численного моделирования таких задач (кроме использования более мощных компьютеров) состоят в поисках оптимальных, специальных методик интегрирования, основанных на учете специфики рассматриваемых задач при обеспечении требуемой точности [42]. Другим подходом является преобразование уравнений движения с целью исключения высокочастотных составляющих либо уменьшения их амплитуды с использованием методов вариации параметров (метод Лагранжа) или координат (метод Энке) [37]. При численном моделировании динамики далеких планетных спутников с целью решения проблемы короткопериодических возмущений массы близких спутников включают в массу планеты [43], что приводит к возмущенным уравнениям задачи двух тел с модифицированным гравитационным параметром:

$$\ddot{\mathbf{x}} = -(\mu_S + \mu_P)\frac{\mathbf{x}}{|\mathbf{x}|^3} + P,\ P = \mu_S\left(\frac{\mathbf{x}}{|\mathbf{x}|^3} - \frac{\mathbf{x}-\mathbf{x}_S}{|\mathbf{x}-\mathbf{x}_S|^3}\right) + \mu_P\left(\frac{\mathbf{x}}{|\mathbf{x}|^3} - \frac{\mathbf{x}-\mathbf{x}_P}{|\mathbf{x}-\mathbf{x}_P|^3}\right),$$

где $\mathbf{x}$ – вектор положения спутника, $\mathbf{x}_S$ и $\mathbf{x}_P$ – векторы положения планет, μ_S и μ_P – гравитационные постоянные планет, P – возмущающая сила, вызванная отклонением планет от их барицентра.

Эффективность методов численного интегрирования уравнений движения в аспекте быстродействия и точности приближения обычно сравнивают на невозмущенной задаче двух тел вида: [2, 16]

$$\ddot{\mathbf{x}} = -\frac{\mu}{|\mathbf{x}|^3}\mathbf{x}, \qquad (1.3)$$

где $\mathbf{x} = (x_1,\ x_2)$ – вектор положения, μ – гравитационный параметр. В экспериментах значение гравитационного параметра принимается равным единице ($\mu = 1$), а начальные условия

$$x_1 = 1 - e,\ x_2 = 0,\ \dot{x}_1 = 0,\ \dot{x}_2 = \sqrt{(1+e)/(1-e)}, \qquad (1.4)$$

соответствующими однопараметрическому семейству орбит $(x_1 + e)^2 + \frac{x_2^2}{1-e^2} = 1$ с эксцентриситетом e в качестве параметра и с единичной большой полуосью. При круговой орбите $e = 0$ или элиптической орбите с небольшим значением эксцентриситета $e = 0.1$ наименьшей абсолютной погрешности, порядка 10^{-15} на отрезке $[0, 6\pi]$, среди современных методов численного приближения уравнений движения [16, 44] достигает интегратор Гаусса-Эверхарта – *GAUSS_32* [15], реализованный в среде *Delphi* [45]. Использование типа данных *extended* в *Delphi* дает повышение точности приближения на 2 – 3 десятичных порядка в сравнении с исходной реализаций метода на языке *Fortran* с использованием переменных типа *Real*8*. Сравнительно высокая точность решения задачи (1.3), (1.4) при невысоких значениях эксцентриситета достигается вследствие достаточной гладкости решения (рис. 1.3) и учета специфики именно плоской задачи двух тел в программе *GAUSS_32* [15].

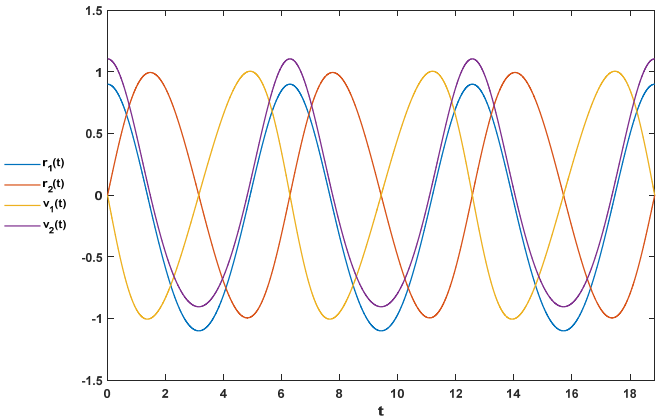

Рис. 1.3. Численное решение задачи (1.3), (1.4) на отрезке $[0, 6\pi]$ при $e = 0.1$

При высоких значениях эксцентриситета $e=0.9$ или $e=0.999$ появляются быстроосциллирующие компоненты решения, вызывающие проблемы для методов численного интегрирования [16] (рис. 1.4). Особо остро в таких случаях стоит вопрос об эффективном управлении величиной шага. Это связано с тем, что в окрестности фокусов шаг должен выбираться достаточно малым, в то время как на других участках его величина может быть достаточно большой.

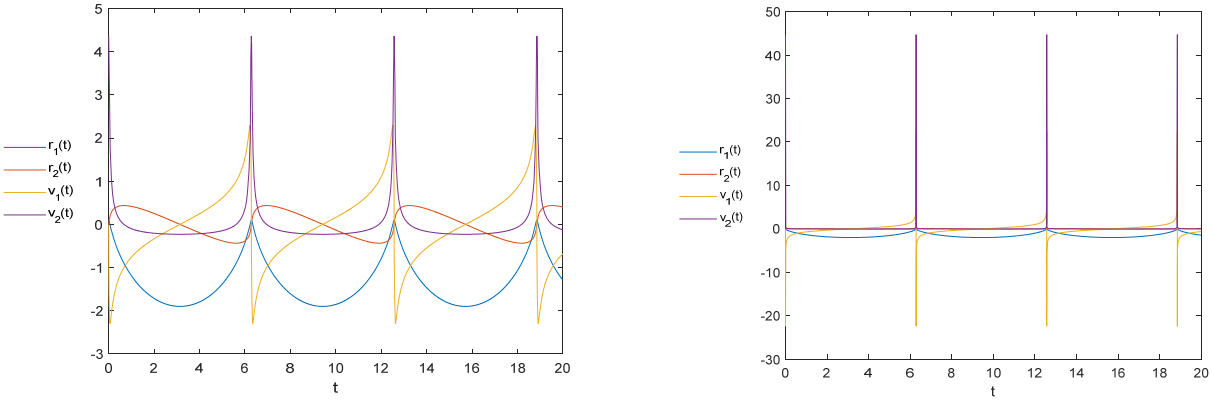

Рис. 1.4. Численное решение задачи (1.3), (1.4) на отрезке $[0, 6\pi]$ при $e=0.9$ (a) и при $e=0.999$ (b)

Некоторым вопросам сравнения эффективности применения традиционных методов Рунге-Кутты и наиболее распространенных при решении задач небесной механики многошаговых методов Адамса и Коуэлла [3, 35], неявного одношагового метода Эверхарта [13, 14] посвящены работы [39, 46].

Проблема повышения эффективности численного моделирования большинства задач небесной мехники обуславливается также их неустойчивостью по Ляпунову, что вызывает повышение ошибок, неизбежно сопровождающих любой численный процесс. Такие задачи принятно называть плохо обусловленными вследствие наличия собственных чисел матрицы Якоби с большими значениями положительной действительной части. Одной из модельных задач для исследования численных методов решения плохо обусловленных задач является задача Аренсторфа [35, 47]. Рассматриваются два тела с массами μ и $1-\mu$, участвующие в совместном движении в некоторой

плоскости, и движущееся в той же плоскости третье тело пренебрежительно малой массы. Уравнения имеют вид

$$\begin{cases} \ddot{x}_1 = x_1 + 2\dot{x}_2 - (1-\mu)\dfrac{x_1+\mu}{D_1} - \mu\dfrac{x_1-1+\mu}{D_2}, \\ \ddot{x}_2 = x_2 - 2\dot{x}_1 - (1-\mu)\dfrac{x_2}{D_1} - \mu\dfrac{x_2}{D_2}, \end{cases} \tag{1.5}$$

где $D_1 = \left(\sqrt{(x_1+\mu)^2 + x_2^2}\right)^3$, $D_2 = \left(\sqrt{(x_1-1+\mu)^2 + x_2^2}\right)^3$, $\mu = 0.012277471$. Начальные условия

$$\begin{aligned} &x_1(0) = 0.994,\ \dot{x}_1(0) = 0,\ x_2(0) = 0, \\ &\dot{x}_2(0) = -2.0015851063\ 7908252240\ 537862224 \end{aligned} \tag{1.6}$$

тщательно подобраны с целью получить замкнутую орбиту с периодом $T = 17.065216560157962558891\,7206249$. Траектория решения, полученного на основе вложенного метода порядков 8 (7), предложенного Принсом и Дорманом в 1981 г., представлена на рис. 1.5. Так как движение по полученной траектории крайне неравномерно, для эффективного численного решения задачи (1.5), (1.6) и многих подобных задач следует использовать методы высоких порядков. Как отмечается в [35], авторы приложили много усилий, «чтобы минимизировать коэффициенты погрешности для аппроксимации 8-го порядка», метод дает «прекрасные численные результаты».

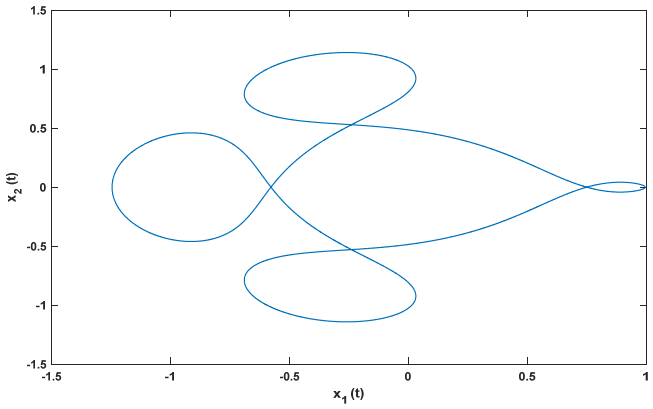

Рис. 1.5. Численное решение задачи (1.5), (1.6) на одном периоде с использованием метода Дормана-Принса 8 (7)

Использование методов невысокого порядка влечет повышение быстродействия, но и быстрое накопление погрешности, например, для решения задачи методом Мерсона 4-го порядка на одном периоде с абсолютной погрешностью 2.1×10^{-3} потребовалось выполнить 1005 вычислений правой части; на двух периодах потребовалось 1910 вычислений правой части, но ошибка в этом случае составила уже 0.336. Если при этом уменьшить шаг с целью повысить точность, то последует снижение быстродействия за счет повторения операций на промежуточных малых шагах, однако, существенного повышения точности приближения в результате можно не достичь в силу изначального построения метода [12, 35].

Следует отметить, что даже с учетом означенных проблем численного моделирования задач небесной механики, развитие численных методов, новых стратегий моделирования хаотических динамических систем и использование суперкомпьютеров позволяют исследователям получать все больше новых теоретических и практических результатов. Относительно задачи трех тел, для которой в течение трехсот лет было известно только три вида периодических орбит, использование численного моделирования и возможностей суперкомпьютера Тяньхэ-2 позволило найти 1349 семейств периодических орбит [48]. Некоторые орбиты имеют красивые и элегантные графические представления (рис. 1.6).

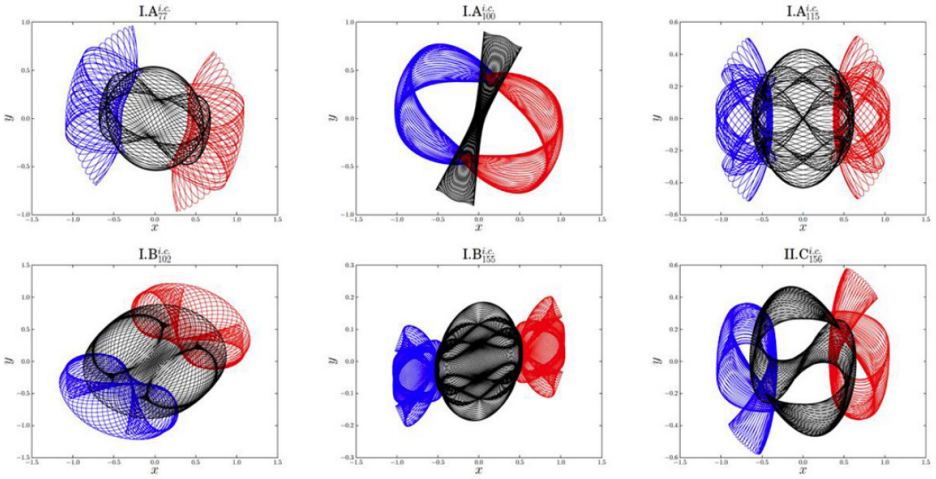

Рис. 1.6. Схематические изображения шести семейств замкнутых орбит задачи трех тел [48]

Численное моделирование производилось авторами с помощью программы, основанной на явном методе Рунге-Кутты с переменным шагом по времени.

1.1.3. Методы обработки данных в моделях управления движением космического аппарата. Отмеченные проблемы повышения эффективности численного моделирования орбит движения небесных тел сохраняются и в задачах баллистико-навигационного обеспечения (БНО) полетов космических аппаратов (КА). В задачах БНО «значительное место занимают определение и уточнение параметров орбит КА (а также осуществленных маневров) по данным траекторных измерений, прогнозирование движения спутников» [49]. При этом обработка измерений и построение орбиты спутника должны проводиться достаточно оперативно, чтобы иметь постоянно обновляемую на основе поступающих данных орбиту КА. Универсальной методикой прогнозирования движения спутника является использование численных методов интегрирования уравнений движения. На практике с целью избежать описанных проблем численного интегрирования точных уравнений движения применяются различные полуаналитические теории (методики) движения спутника [49], которые получаются путем исключения трудно интегрируемых подсистем. Однако они не приемлемы для произвольного состава сил, действующих на спутник, и для произвольного класса орбит КА. Математическая модель движения КА с управлением при помощи «малой тяги» в плоскости орбиты может быть представлена следующей системой уравнений [50, 51]:

$$\begin{cases} \dot{r} = V_r, \quad \dot{V}_r = \dfrac{V_\theta^2}{r} - \dfrac{h^2}{r^2 p} + U_r, \\ \dot{\theta} = \dfrac{V_\theta}{r}, \quad \dot{V}_\theta = -\dfrac{V_r V_\theta}{r} + U_\theta, \end{cases} \tag{1.7}$$

где r, θ – полярные координаты, V_r, V_θ – радиальная и трансверсальная составляющие скорости; U_r, U_θ – составляющие вектора тяги, p – фокальный параметр. Для параметров системы (1.7) имеют место следующие соотношения:

$$a = \frac{p}{1-e^2},\ b = \frac{p}{\sqrt{1-e^2}},\ h = \frac{2\pi a b}{T},\ \frac{h^2}{p} = GM\ ,$$

где a, b – большая и малая полуоси эллипса, e – эксцентриситет эллипса. Управляющие воздействия считаются малыми по сравнению с минимальным значением силы тяготения, т.е. $\sqrt{U_r^2 + U_\theta^2} << \frac{h^2}{p\, r_{\max}^2}$ [51]. При $U_r = U_\theta = 0$ и отрицательной полной энергии $E < 0$ система (1.7) описывает движение КА по эллиптической орбите под действием закона тяготения Ньютона [52]. На основе законов Кеплера с использованием метода АКАР в [52] синтезированы системные законы управления U_r и U_θ, обеспечивающие устойчивое орбитальное движение КА:

$$U_r = \frac{h^2}{p\, r^2} - \frac{(\omega_2 + h)^2}{p\, r^2} - \frac{\Phi}{p}\left(r\,\dot{\omega}_1(t) + e\,\omega_2 \sin\theta\right),\ U_\theta = -\frac{\omega_2}{r}\ \Phi, \qquad (1.8)$$

где $\omega_1 = r(1 + e\cos\theta) - p = 0$, $\omega_2 = r^2\,\dot{\theta}(t) - h = 0$ – законы Кеплера. В качестве Φ может быть выбрана функция $\Phi = \frac{h}{r^2}$ [52]. При $\dot{\omega}_1(t) = \omega_2 = 0$ КА устойчиво движется на периодической орбите. На практике система управления (1.8) может использоваться «при решении важных задач управления КА, навигации Земли с помощью спутниковых систем» [53].

Для управления движением КА необходимо в режиме реального времени находить приближенное решение системы (1.7), (1.8) с высокой точностью. Результаты численного моделирования решения системы (1.7), (1.8) на промежутке $[0,\ 5.5\cdot 10^5]$ для случая $\dot{\omega}_1(t) \neq 0$, $\omega_2(t) \neq 0$ при начальных условиях

$$r(0) = r_0,\ V_r(0) = V_0,\ \theta(0) = 0,\ V_\theta(0) = 0, \qquad (1.9)$$

где $r_0 = \frac{p}{1+e}$ *км*, $p = 36000$ *км*, $e = 0.5$, $V_0 = \frac{eh}{p}$ *км*/*сек*, $T = 24$ *ч*., $h = \frac{2\pi ab}{T}$

представлены на рис. 1.7.

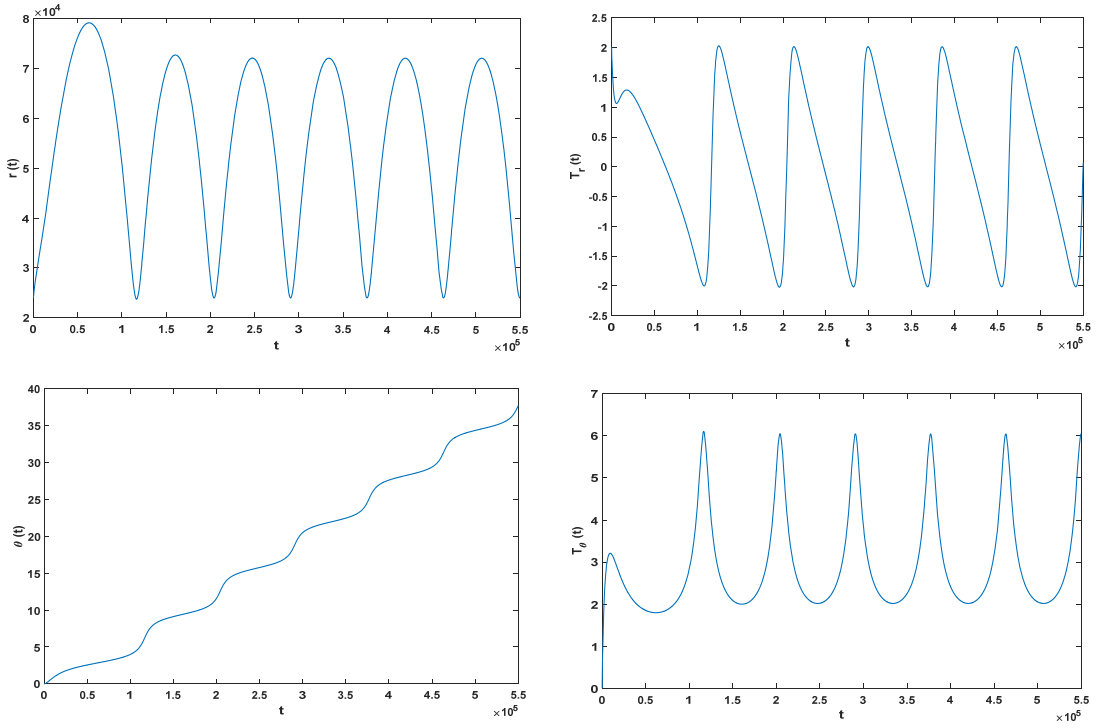

Рис. 1.7. Решение задачи (1.7) – (1.9) методом Дромана-Принса 8 (7)

Численное моделирование решения задачи (1.7) – (1.9) выполнено на основе отмеченного выше метода Дормана-Принса 8 (7), реализованного в среде *Delphi*. Время решения на компьютере *Intel Core i5-2500* с равномерным шагом интегрирования 10^{-1} составило $23s$. На рис. 1.8. представлены графики изменения $r(t)$ в окрестности одной из точек минимума на основе функции *ode45* (*MATLAB*) [54], реализующей кратко охарактеризованный выше метод Дормана-Принса 5-го порядка аппроксимации, и на основе метода Дормана-Принса 8 (7) с шагом интегрирования $h = 10^{-1}$. С целью снижения трудоемкости метод в *MATLAB* реализован с механизмом управления длиной шага интегрирования.

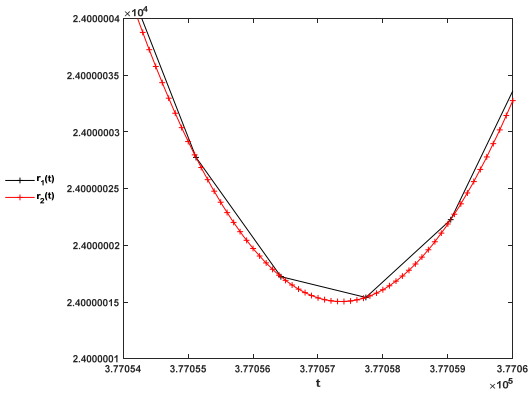

Рис. 1.8. График изменения $r(t)$, задача (1.7) – (1.9), $r_1(t)$ – функция *ode45* (*MATLAB*), $r_2(t)$ – метод Дормана-Принса 8 (7) ($h = 10^{-1}$)

Функция *ode45* позволила получить приближение решения задачи за $3\,s$, однако использование механизма управления величиной шага снизило качество визуализации результатов моделирования (рис. 1.8).

Таким образом, и в данной области актуальна проблема повышения точности методов приближенного решения задачи (1.1). При этом, однако, требуется сохранять возможность использования механизма управления длиной шага интегрирования и возможность выдачи результатов приближения в промежуточных между шагами сетки точках без дополнительных вычислений функции правой части.

1.1.4. Компьютерная обработка данных в моделях периодических химических и биологических процессов. Применение численных методов решения задачи (1.1), обладающих высокой точностью и одновременно малой трудоемкостью, как отмечалось, актуально не только в задачах небесной механики. Сходные проблемы численного моделирования имеют колебательные системы и в других областях. Так, численное моделирование решения задачи Коши (1.1) актуально и при исследовании математических моделей химических и биологических осцилляторов. Модели колебательных систем используются в ферментативном катализе, теории иммунитета [55], в теории трансмембранного ионного переноса, микробиологии и биотехнологии [56 – 58]. Как правило, модели осцилляторов приводят к задаче (1.1), где

$\mathbf{y} = \left(y_1(t), y_2(t), \ldots, y_n(t)\right)$ – вектор концентраций, а вектор $\mathbf{f}$ описывает нелинейную реакционную кинетику или механизм, лежащий в основе колебаний, химических или биологических [59, 60]. При этом большинство задач являются жесткими или плохо обусловленными.

Среди известных колебательных химических реакций одной из важных и часто применяемых для тестирования численных методов является реакция Белоусова-Жаботинского [61]. Ниже рассмотрены ее ключевые стадии и представлены некоторые результаты численного моделирования. Модели реакции Белоусова-Жаботинского используют как прототип реальной системы, поскольку теоретические гипотезы могут быть проверены экспериментально. Полученные при исследовании моделей реакции Белоусова-Жаботинского результаты могут быть использованы также в биологических осцилляторах [62]. Основным в исходной реакции Белоусова [61] является механизм окисления малоновой кислоты в кислом растворе ионами бромата BrO_3^-. Устойчивые периодические колебания возникают вследствие того, что реакция катализируется ионами церия, имеющего два состояния – Ce^{3+} и Ce^{4+}. В течение времени меняется концентрация не только катализатора, но и других участников реакции. На рис. 1.9 показаны экспериментальные кривые концентрации бромида $[Br^-]$ и отношения концентраций ионов церия $[Ce^{4+}] / [Ce^{3+}]$ во времени из работы Филда и Нойеса [63].

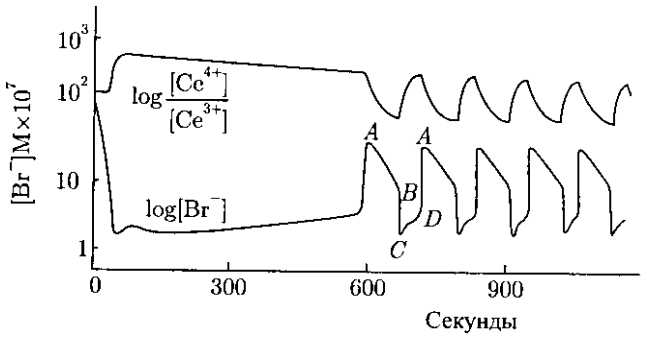

Рис. 1.9. Экспериментально измеренные изменения отношения концентраций ионов церия $[Ce^{4+}]/[Ce^{3+}]$ и концентрации бромида $[Br^-]$ в реакции Белоусова-Жаботинского [63]

Схема автоколебательной реакции может быть качественно описана следующим образом. Реакция разделяется на два процесса, I и II, зависящих от концентрации $[Br^-]$. При высокой $[Br^-]$ преобладает процесс I (область A на рис. 1.9). При этом Br^- потребляется, что соответствует участку AB, и ионы церия, в основном, находятся в состоянии Ce^{3+}. По мере уменьшения $[Br^-]$ проходит критическое значение B и быстро падает до уровня C. С этого момента преобладает процесс II, в течение которого Ce^{3+} переходит в Ce^{4+}. Однако на стадии II Ce^{4+} вступает в реакцию, ведущую к образованию Br^-, возвращаясь в состояние Ce^{3+}. Теперь $[Br^-]$ возрастает, что соответствует участку CDA, и при достаточно высоком ее значении стадия I вновь становится преобладающей. Эта последовательность постоянно повторяется, и, таким образом, реализуются наблюдаемые колебания. Очевидно, что качественное описание не доказывает наличия колебаний. Процессы I и II могут достигнуть устойчивого состояния сосуществования, для получения колебаний необходимы определенные значения параметров. При определении значений параметров помимо экспериментов используются также методы численного моделирования, в частности, методы приближенного решения задачи (1.1). Естественно, что с целью наиболее точного определения значений параметров от численных методов требуется высокая точность приближения. Помимо этого, при численном моделировании реакции, математической моделью которой, как будет означено ниже, является задача (1.1), с уже определенными значениями параметров, незначительные ошибки применяемого численного метода могут привести к неверным результатам моделирования (в результатах моделирования не будет наблюдается устойчивый предельный цикл реакции).

Высокая скорость изменения концентрации на участках BC и DA характерна для релаксационных осцилляторов [59] (осцилляторов, у которых отдельные стадии предельного цикла протекают быстро). Несмотря на большое

количество вовлеченных реакций, для описания основных процессов возможно сокращение их количества до 5 ключевых с известными константами скоростей. Такая модель, предложенная Филдом и Нойесом [63], получила название «орегонатор». Схема реакций имеет вид:

$$A + Y \xrightarrow{k_1} X,\ X + Y \xrightarrow{k_2} P$$
$$B + X \xrightarrow{k_3} 2X + Z,\ 2X \xrightarrow{k_4} Q,\ Z \xrightarrow{k_5} f\,Y$$

где A, B – исходные реагенты, P, Q – продукты реакции, X, Y, Z – промежуточные соединения: бромистая кислота $HBrO_2$, бромид-ион Br^-, Ce^{4+}. Концентрации исходных реагентов полагают в модели неизменными.

При обозначении малыми буквами переменных, соответствующих концентрациям реагентов, уравнения, описывающие изменения концентраций автокатализатора ($x = [HBrO_2]$), бромид-иона ($y = [Br^-]$) и катализатора ($z = [Ce^{4+}]$) во времени в соответствии с законом действующих масс имеет вид:

$$\frac{dx}{dt} = k_1 a y - k_2 x y + k_3 b x - 2k_4 x^2,\ \frac{dy}{dt} = -k_1 a y - k_2 x y + f k_5 z,$$
$$\frac{dz}{dt} = k_3 b x - k_5 z.$$

Значения констант скоростей прямых реакций получены Филдом и Нойесом из экспериментальных данных [63]:

$$[A] = [B] = 0.06\,\text{M},\ k_1 = 1.34\ \text{M/c},\ k_2 = 1.6 \times 10^9\ \text{M/c},$$
$$k_3 = 8 \times 10^3\ \text{M/c},\ k_4 = 4 \times 10^7\ \text{M/c}.$$

Для сопоставления с экспериментальными наблюдениями автоколебаний в [63] использованы следующие дополнительные значения переменных: $k_5 = 1,\ f = 1$. Безразмерная форма записи модели «орегонатор» имеет вид:

$$\frac{d\alpha}{d\tau} = s(\eta - \eta\alpha + \alpha - q\alpha^2),\ \frac{d\eta}{d\tau} = s^{-1}(-\eta - \eta\alpha + f\rho),\ \frac{d\rho}{d\tau} = w(\alpha - \rho),$$

где

$$[HBrO_2] = x = \frac{k_1 a}{k_2} = 5.025\times10^{-11}\alpha,\ [Br] = y = \frac{k_3 b}{k_2}\eta = 3.000\times10^{-7}\eta,$$

$$[Ce^{4+}] = z = \frac{k_1 k_3}{k_2 k_5} a b \rho = 2.412\times10^{-8}\rho,\ Time = t = \frac{\tau}{\sqrt{k_1 k_3 a b}} = 0.1610\tau,$$

$$s = \sqrt{k_3 b / k_1 a} = 77.27,\ w = \frac{k_5}{\sqrt{k_1 k_3 a b}} = 0.1610,\ q = \frac{2k_1 k_4 a}{k_2 k_3 b} = 8.375\times10^{-6}.$$

Математическая модель с данными параметрами (соответствующими устойчивому предельному циклу), представленная в виде

$$\begin{cases} y_1' = 77.27\ (y_2 + y_1(1 - 8.375\times10^{-6}\ y_1 - y_2)), \\ y_2' = \dfrac{1}{77.27}\ (y_3 - y_2(1 + y_1)), \\ y_3' = 0.161\ (y_1 - y_3), \end{cases} \tag{1.10}$$

применялась в [3] при тестировании численных методов для решения жестких задач. Здесь и всюду ниже, если не оговорено иное, y' означает производную по времени. В [35] отмечено, что «данный пример служит серьезным испытанием для программ численного интегрирования». В [3] решение задачи (1.10) приближалось при помощи специализированных для жестких задач программ: *RODAS*, основанной на методе Розенброка, экстраполяционной программой *SEULEX*, многошаговой программой *LSODE*, а также программой *RADAU5*, реализующей неявные методы Рунге-Кутты. На рис. 1.10 представлены результаты численного моделирования системы (1.10) с начальными условиями $y_1(0) = 4,\ y_2(0) = 1.1,\ y_3(0) = 4$ на интервале $t \in [0, 500]$.

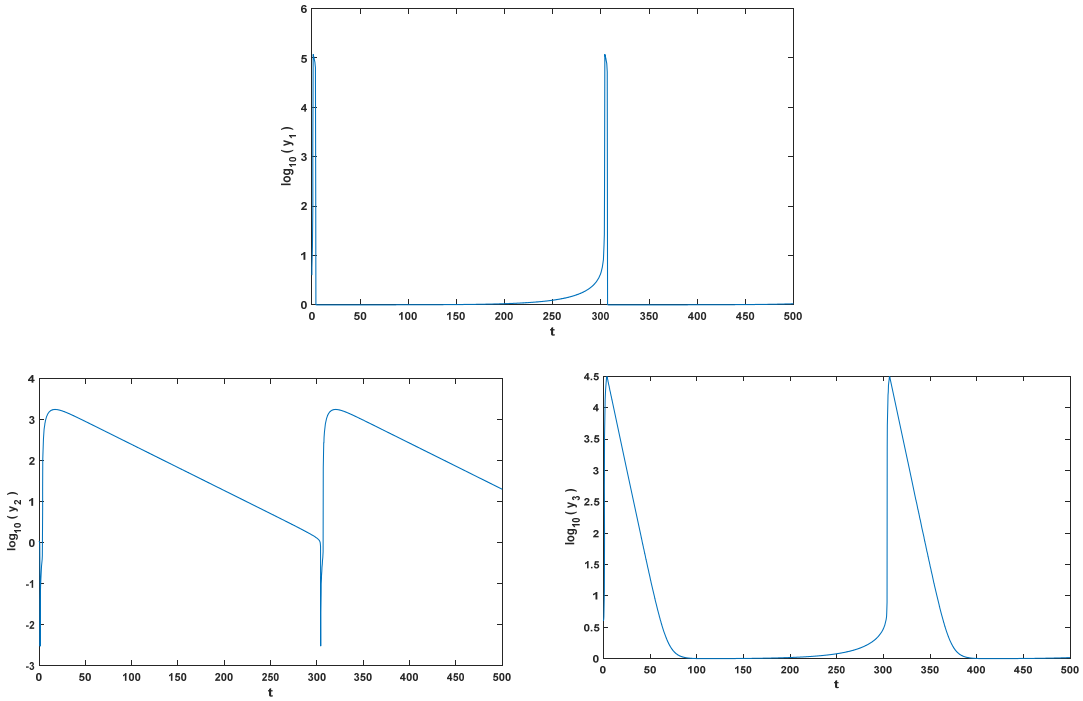

Рис. 1.10. Численное решение системы (1.10) с начальными условиями $y_1(0)=4,\ y_2(0)=1.1,\ y_3(0)=4$

Приближение получено на основе специально созданного для решения жестких задач решателя *ode15s* (*MATLAB*), реализующего метод конечных разностей переменного порядка с переменным шагом в сочетании с методом Гира. Экстраполяционная программа на основе линейно неявного метода Эйлера (*Stiff EULer Extrapolation*), представленная в [3], дает приближение решения задачи (1.10) с погрешностью 10^{-9}. Высокая степень жесткости задачи не позволяет применить явные методы Рунге-Кутты высоких порядков. Решение задачи с высокой точностью явными методами Рунге-Кутты даже на многопроцессорной рабочей станции может занимать несколько часов [3].

При потере точности численного интегрирования результат моделирования может буквально потерять смысл (в результатах моделирования не будет наблюдается устойчивый предельный цикл реакции). Таким образом, в данном классе задач, как и классах рассмотренных выше, применение высокоточных методов решения задачи (1.1) является актуальным. При этом требуется сочетать точность с приемлемой трудоемкостью.

Узкая специализированность известных методов требует неформального анализа для выбора подходящего метода численного интегрирования. Это может повлечь необходимость специального создания нового метода. Все вместе показывает актуальность вопроса о построении инвариантного высокоточного метода решения дифференциальных систем при условии сравнительно низкой трудоемкости.

Поскольку во всех отмеченных процессах принципиальное значение имеет устойчивость по Ляпунову решения дифференциальных систем, следует отметить, что точность численного решения задачи (1.1) может играть существенную роль в компьютерных методах анализа устойчивости, основанных на приближенном решении дифференциальных систем [64, 65].

1.2. Методы обработки данных в ИВС для дифференциальных моделей. Ниже представлена детализация алгоритмов, наиболее распространенных в практических вычислениях, и численных методов решения задачи (1.1) с целью исследованияособенностей и принципов использования методов в соответствии со специфическими качествами систем ОДУ. Конструкция численных методов описывается для случая $n=1$ в (1.1), то есть для задачи вида $\frac{d\,y}{d\,t}=f\,(t,\,y)\,,\quad y\,(t_0)=y_0\,.$ Как уже отмечалось, выбор универсального численного метода фактически исключается противоречивостью требований в аспекте точности и быстродействия. В связи с этим разработан широкий класс численных методов, имеющих, как правило, проблемно ориентированный или специализированный характер. При выборе численного метода принципиальную роль играет тип решаемой задачи Коши, в значительной степени определяемый спектром матрицы Якоби системы (1.1) – $\mathbf{J}\,(t)=\frac{\partial\,\mathbf{f}\,(t,\,\mathbf{y}(t))}{\partial\,\mathbf{y}}$. В зависимости от расположения наибольших по модулю собственных значений матрицы выделяют [47]: жесткие (значения в левой полуплоскости), колебательные (значения вблизи мнимой оси) и плохо

обусловленные задачи (значения в правой полуплоскости) задачи. Задачи, не содержащие указанных проблем, условно называют нежесткими. При решении нежестких задач часто используются явные методы Рунге-Кутты (ЯМРК) [35]:

$$
\begin{aligned}
&k_1 = f(t_0, y_0),\\
&k_2 = f(t_0 + c_2 h, y_0 + h a_{21} k_1),\\
&k_3 = f(t_0 + c_3 h, y_0 + h(a_{31} k_1 + a_{32} k_2)),\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&k_s = f(t_0 + c_s h, y_0 + h(a_{s1} k_1 + \ldots + a_{s,s-1} k_{s-1})),\\
&y_1 = y_0 + h\,(b_1 k_1 + \ldots + b_s k_s),
\end{aligned}
$$

где $s \in Z^+$ – число стадий, $a_{21}, a_{31}, a_{32}, \ldots, a_{s1}, a_{s2}, a_{s,s-1}, \ldots, b_1, \ldots, b_s, c_2, \ldots c_s$ – вещественные коэффициенты. Для краткости метод обычно представляется символически (табл. 1.1).

Таблица 1.1

Символическое представление ЯМРК

0					
c_2	a_{21}				
c_3	a_{31}	a_{32}			
…	…	…			
c_s	a_{s1}	a_{s2}	…	$a_{s,s-1}$	
	b_1	b_2	…	b_{s-1}	b_s

К достоинствам методов Рунге-Кутты (РК) относят одношаговость, отсутствие необходимости в дополнительных вычислениях, сопоставимость с тейлоровскими разложениями [1, 35]. Выбор коэффициентов и числа стадий определяет порядк метода. Программные реализации методов РК представлены в большинстве библиотек программ, предназначенных для выполнения научных и инженерных расчетов. В программном пакете *MATLAB* функция *ode23* реализует ЯМРК 3-го порядка, модифицированный Богацки и Шампайном, для вывода значений приближения между разностными шагами используется кубическая интерполяция Эрмита [35, 36]. Метод рекомендуется применять при умеренной жесткости системы ОДУ и низких требованиях к

точности. Среди методов ЯМРК 4-го порядка вследствие незначительных различий в результатах тестов сравнения чаще используется «классический» метод Рунге-Кутты [35, 66]. В программном пакете *MathCAD* схему реализует функция *rkfixed* [32].

При сравнении ЯМРК невысоких порядков обычно анализируется зависимость максимальной глобальной погрешности от числа обращений к подпрограмме вычисления значения функции. Кроме того, порядок точности методов РК связан с условием сходимости метода: для сходимости метода РК порядка p требуется $p+1$ непрерывная производная от решения [12]. В большинстве задач, моделирующих реальные динамические системы, высокую степень гладкости правой части (1.1) гарантировать невозможно.

Важное значение, в частности, при построении методов с автоматизированным управлением величиной шага, имеет алгоритм практической оценки погрешности приближения. Оценка погрешности значения y_2, полученного в результате выполнения двух шагов длины h по ЯМРК порядка p, обычно производится по правилу Рунге: $y(t_0+2h)-y_2=\frac{y_1-w}{2^p-1}+O(h^{p+1})$, w – значение, полученное в результате выполнения одного шага длины $2h$. Кроме оценки погрешности по правилу Рунге можно получить экстраполированное по Ричардсону значение величины $y(t_0+2h)$ с порядком $p+1$: $\widetilde{y}_2=y_2+\frac{y_2-w}{2^p-1}$. Однако в практических вычислениях вместо экстраполяции Ричардсона используют вложенные формулы (табл. 1.2), которые в своем алгоритме уже содержат выражение для получения приближения решения $\widetilde{y}_1$ более высокого порядка [35].

Таблица 1.2

Символическое представление вложенных формул РК

0					
c_2	a_{21}				
c_3	a_{31}	a_{32}			
…	…	…			
c_s	a_{s1}	a_{s2}	…	$a_{s,s-1}$	
	b_1	b_2	…	b_{s-1}	b_s
	$\tilde{b}_1$	$\tilde{b}_2$	…	$\tilde{b}_{s-1}$	$\tilde{b}_s$

Величина $y_1 = y_0 + h(b_1 k_1 + ... + b_s k_s)$ приближает решение на шаге с порядком p, а $\tilde{y}_1 = y_0 + h(\tilde{b}_1 k_1 + ... + \tilde{b}_s k_s)$ – с более высоким порядком q. Из множества вложенных методов в практических вычислениях наиболее часто используется метод Фельберга 4-го порядка (табл. 1.3) [36]. В скобках в названии вложенных методов приводится порядок оценки погрешности – q.

Таблица 1.3

Фельберг 4 (5)

0						
1 / 4	1 / 4					
3 / 8	3 / 32	9 / 32				
12 / 13	1932 / 2197	7296 / 2197				
1	439 / 216	−8	3680 / 513	−845 / 4104		
1 / 2	−8/27	2	−3544 / 2565	1859 / 4104	−11 / 40	
y_1	25 / 216	0	1408 / 2565	2197 / 4104	−1 / 5	0
$\tilde{y}_1$	16 / 135	0	6656 / 12825	28561/ 56430	−9/50	2 / 55

Метод Фельберга 4 (5) реализован функцией *rkf45* в *Maple* [33], функцией *ode45* в ранних версиях *MATLAB.* Впоследствии ввиду высокой эффективности в функции *ode45* был реализован метод Дормана-Принса 5 (4), который использует подход, заключающийся в минимизации членов погрешности для результата старшего порядка [36, 67]. Кроме того, этот метод не требует дополнительных вычислений функции правой части для интерполяции значений приближения между шагами сетки, что важно для визуализации

результатов моделирования в *MATLAB*. В программном пакете *MathCAD* ЯМРК с автоматизированным управлением величиной шага реализует функция *Rkadapt*.

Преимущества методов приближения систем ОДУ с адаптацией величины шага наглядно проявляются при приближении решения классической модели автоколебаний в системе химических реакций – «брюсселятор»:

$$\begin{cases} y_1' = 1 + y_1^2 y_2 - 4 y_1, \\ y_2' = 3 y_1 - y_1^2 y_2 \end{cases} \qquad (1.11)$$

с начальными условиями $y_1(0) = 1.01$, $y_2(0) = 3$, близкими к особой точке (рис. 1.11) [35].

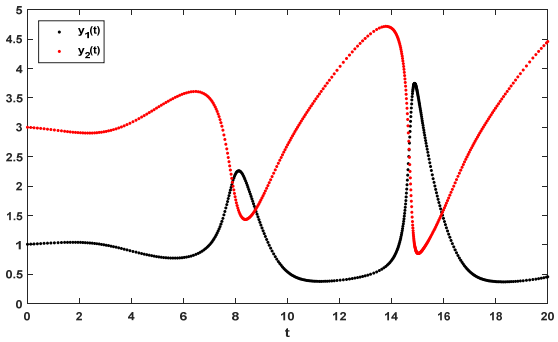

Рис. 1.11. Решение задачи (1.11) с помощью ЯМРК с адаптацией величины шага

К недостаткам методов с автоматическим управлением длиной шага относят невозможность «плотной» выдачи результатов приближения. Одним из способов реализации «плотной» выдачи является уменьшение шага интегрирования, но это влечет повышение трудоемкости, то есть, снижается эффективность методов управления величиной шага, реализуемых с целью повышения быстродействия.

Для обеспечения вывода результатов приближения в промежуточных между шагами сетки точках для ЯМРК невысокого порядка сконструированы непрерывные методы. Их применение позволяет получать приближение решения во всех точках $t^* = t_0 + \theta h$, $0 < \theta \le 1$, без дополнительных вычислений функции правой части. Неплохим инструментом для плотной выдачи

результатов и графического представления решения является непрерывное расширение 4-го порядка для метода Дормана-Принса 5 (4). Решение переходит в решение y_1 пятого порядка точности при $\theta = 1$ и определяется следующими формулами [35]:

$$b_1(\theta) = \theta(1 + \theta(-1337/480 + \theta(1039/360 + \theta(-1163/1152)))),$$
$$b_2(\theta) = 0,$$
$$b_3(\theta) = 100\theta^2(1054/9275 + \theta(-4682/27825 + \theta(379/5565)))/3,$$
$$b_4(\theta) = -5\theta^2(27/40 + \theta(-9/5 + \theta(88/96)))/2,$$
$$b_5(\theta) = 18225\,\theta^2(-3/250 + \theta(22/375 + \theta(-37/600)))/848,$$
$$b_6(\theta) = -22\,\theta^2(-3/10 + \theta(29/30 + \theta(-17/24)))/7,$$
$$y(t_0 + \theta h) \approx y_0 + h\sum_{j=1}^{6} b_j(\theta) k_j .$$

При этом в промежуточных точках происходит незначительное снижение точности численного решения. Универсальным подходом для обеспечения вывода значений решения в промежуточных точках является интерполяция полученных численных данных. В функции *ode23* (*MATLAB*), как отмечалось выше, используется кубическая интерполяция Эрмита, с которой, как известно [68], на каждом частичном отрезке $[t_{i-1}, t_i]$ совпадает наиболее распространенный локальный кубический интерполяционный сплайн

$$S_3(t) = P_{3,i}(t) = \frac{(t-t_i)^2(2(t-t_{i-1})+h_i)}{h_i^3} y_{i-1} + \frac{(t-t_{i-1})^2(2(t_i-t)+h_i)}{h_i^3} y_i + \\ + \frac{(t-t_i)^2(t-t_{i-1})}{h_i^2} y'_{i-1} + \frac{(t-t_{i-1})^2(t-t_i)}{h_i^2} y'_i, \qquad (1.12)$$

где $h_i = t_i - t_{i-1}$, обеспечивающий непрерывность функции и ее первой производной. Оценка погрешности такой интерполяции [68]:

$$\max_{[a,\,b]} \left| y(t) - S_3(t) \right| \le \frac{M_4}{384} h_{\max}^4 ,$$

где $h_{\max}$ – максимальная из длин частичных отрезков. При этом следует учитывать, что погрешность приближения значений производных, используемых в качестве коэффициентов в (1.12), в случае интерполяции результатов численного решения (1.1), как правило, больше погрешности приближения решения [12]. Тем самым оценка погрешности интерполяции снижается при учете дополнительной погрешности приближения значений производных интерполируемой функции. Использование отличных от описанного способов выбора наклонов сплайна, приводящих к другим локальным сплайнам (кубический многочлен Бесселя, метод Акимы и др. [69]) не приводит к существенным различиям в оценках точности и трудоемкости [68]. Отметим, что с учетом низкой точности приближений при использовании численных методов невысокого порядка применение кубической интерполяции, в частности, многочлена Эрмита вполне оправдано.

Проблема снижения погрешности интерполяции имеет принципиальное значение при численном моделировании решения задачи (1.1) с использованием методов высоких порядков. Среди ЯМРК высокого порядка в практических вычислениях используются: метод Бутчера 6-го порядка (табл. 1.4), 17-стадийный метод Хайрера 10-го порядка (при требованиях к точности выше порядка 10^{-15}) [35]. Коэффициенты метода Хайрера построены на базе квадратурной формулы Лобатто 10-го порядка и с 21 десятичным знаком приведены в работе [70].

Таблица 1.4

Метод Бутчера 6-го порядка

0							
1/2	1/2						
2/3	2/9	4/9					
1/3	7/36	2/9	−1/12				
5/6	−35/144	−55/36	35/48	15/8			
1/6	−1/360	−11/36	−1/8	1/2	1/10		
1	−41/260	22/13	13/156	−118/39	32/195	80/39	
	13/200	0	11/40	11/40	4/25	4/25	13/200

Среди вложенных формул высших порядков наибольшее распространение получил кратко охарактеризованный выше метод Дормана-Принса 8 (7) [71].

Сравнительно высокий порядок точности при решении нежестких задач также позволяют получить многошаговые методы, среди которых наиболее употребительны конечно-разностные методы [72]

$$\sum_{i=0}^{k} a_{-i}\, y_{n-i} - h \sum_{i=0}^{k} b_{-i}\, f_i(t_{n-i},\, y_{n-i}) = 0. \qquad (1.13)$$

В вычислительной практике применяют как экстраполяционные формулы (со значениями $a_0 \neq 0,\ b_0 = 0$) так и интерполяционные ($a_0 \neq 0,\ b_0 \neq 0$). При высоких требованиях к точности приближения предпочтительнее применять экстраполяционные программы. В программном пакете *MathCAD* на методе Булирша-Штера с использованием рациональной экстраполяции основана функция *bulstoer*, которая используется при приближении нежестких систем ОДУ при высокой степени гладкости правой части системы [32]. В *MATLAB* решатель *ode113* реализует многошаговый метод Адамса-Башворта-Мултона переменного порядка (от 1 до 13). Экстраполяционный метод с комбинированным управлением длиной шага и порядком метода эффективно реализован в [35] с помощью программы *ODEX*.

В [35] на серии тестовых задач представлено сравнение вычислительных качеств наиболее эффективных методов высокоточного решения нежестких задач (1.1): метода Дормана-Принса 8 (7) и экстраполяционной программы *ODEX*. Типичные результаты дает тестовая задача двух тел (здесь она представлена в форме, отличной от (1.3), что обусловлено стандартными обозначениями для данной предметной области):

$$\begin{aligned} &y_1' = y_3,\ \ y_2' = y_4,\ \ y_3' = -\frac{y_1}{(y_1^2 + y_2^2)^{3/2}},\ \ y_4' = -\frac{y_2}{(y_1^2 + y_2^2)^{3/2}}, \\ &y_1(0) = 0.5,\ \ y_2(0) = 0,\ y_3(0) = 0,\ \ y_4(0) = \sqrt{3}. \end{aligned} \qquad (1.14)$$

На отрезке $[0, 6\pi]$ метод Дормана-Принса 8-го порядка характеризуется границей абсолютной погрешности порядка 10^{-14} при ≈ 39000 обращений к функции правой части, экстраполяционная программа *ODEX* на том же интервале достигает границы погрешности порядка 10^{-13} при ≈ 36000 обращений к функции правой части. Метод Булирша-Штера, реализованный в *MathCAD*, не превышает погрешности порядка 10^{-9}, графики решения представлены на рис. 1.12.

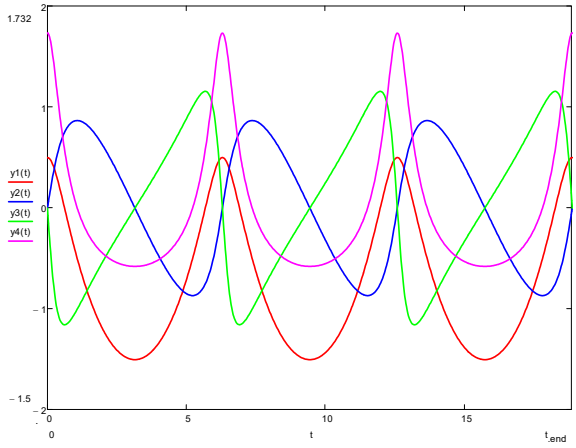

Рис. 1.12. Решение задачи (1.14) методом Булирша-Штера

Сравнительно высокая точность приближения при использовании данных методов приводит, при необходимости получения непрерывного приближения, к проблеме снижения погрешности интерполяции. Одним из способов снижения погрешности полиномиальной интерполяции может являться повышение степени интерполяционных многочленов, однако, это приводит к существенному повышению сложности вычислений и к быстрому накоплению ошибок округления при их вычислении [72]. Другим способом может являться увеличение числа частичных отрезков для сплайн-интерполяции. При глобальном способе построения сплайнов увеличение числа частичных отрезков влечет рост размерности линейной системы алгебраических уравнений, решение которой необходимо для склеивания значений в точках «стыка» многочленов [68]. Проблемы, возникающие при использовании локального кубического интерполяционного сплайна, отмечены выше. В

основном применение сплайн-интерполяции связано с необходимостью обеспечения достаточной степени гладкости аппроксимирующих функций, что актуально, например, при аппроксимации обводов конструктивных элементов сложных поверхностей, при этом, как правило, не требуется достижение высокой точности приближения.

В данном аспекте можно отметить, что высокую точность интерполяции при минимальной временной сложности можно получить при использовании варьируемого кусочно-полиномиального метода вычисления действительной функции одной действительной переменной. Детальное описание и математическое обоснование этого метода приводится в главе 2. Здесь предварительно отметим отдельные особенности. Метод основан на аппроксимации интерполяционными полиномами Ньютона, приведенными к виду алгебраических полиномов с числовыми коэффициентами в форме с явными значениями числовых коэффициентов [73]. Преобразование опирается на видоизменение формул Виета для восстановления коэффициентов полинома по его корням [74, 75]. Однако, его непосредственное применение требует наличия высокоточного приближения решения задачи (1.1) с достаточной для обеспечения точности интерполяции малостью шага интегрирования, что влечет повышение трудоемкости. В главе 3 на базе данного подхода конструируется метод варьируемого кусочно-интерполяционного решения задачи (1.1) с итерационным уточнением [76]. Метод позволяет получить высокоточное непрерывное и непрерывно дифференцируемое приближение решения задачи (1.1) при малой трудоемкости. Для сравнения кусочно-интерполяционное решение задачи (1.14) восстанавливается с границей абсолютной погрешности порядка 10^{-17} при 68409 обращениях к вектор-функции правой части, с границей погрешности порядка 10^{-16} – при 33696 обращениях.

В отличие от описанных выше методов приближения решения нежестких задач конструируемое кусочно-интерполяционное решение с итерационным

уточнением применимо и для приближения решения жестких задач. Понятие жестких задач ввели в 1952 г. Куртис и Хиршфельдер. Жесткой называют систему, содержащую как быстрозатухающие, так и медленно меняющиеся компоненты решения [77]. Специально жестким системам посвящены [3, 77, 78]. Общепринятого формального определения жесткости нет. В монографии [79] к классу жестких относят системы, для которых справедливо неравенство

$$\mu(\mathbf{J}) = \frac{\max\limits_i \left|\mathrm{Re}\ \lambda_i\right|}{\min\limits_i \left|\mathrm{Re}\ \lambda_i\right|} >> 1,\ \mathrm{Re}\lambda_i < 0,\ i = 1,...,n,$$

где λ_i – собственные числа матрицы Якоби. Однако данное определение не охватывает всего класса жестких задач, применимо только к устойчивым системам и игнорирует длину промежутка наблюдения решения [78]. К жестким относят задачи химической кинетики, нестационарные процессы в сложных радиоцепях, системы, возникающие при решении уравнений теплопроводности и диффузии методом прямых, и многие другие. Одним из подходов к решению таких задач являются ЯМРК с расширенными областями устойчивости [80]. Однако наиболее широкое использование получил подход, заключающийся в применении неявных абсолютно устойчивых методов РК с их адаптацией к решению жестких задач [3, 81]. Среди безитерационных наиболее популярным при приближении задачи Коши для нелинейных автономных систем вида

$$u' = f(u),\ u(0) = u_0, \tag{1.15}$$

являются многостадийные схемы семейства Розенброка [82]:

$$y_{n+1} = y_n + h \sum_{m=1}^{s} b_m w_m,$$

$$\left[E - h a_{mm} f_u\left(y_n + h\sum_{k=1}^{m-1} c_{mk} w_k,\ t_n + h c_m\right)\right] w_m = f\left(y_n + h\sum_{k=1}^{m-1} a_{mk} w_k,\ t_n + h a_m\right),$$

где E – единичная матрица, f_u – матрица Якоби для системы (1.15), s – число стадий метода, $a_{mk}, a_m, b_m, c_{mk}, c_m$ – заданные коэффициенты. Методы Розенброка относятся к классу линейно неявных методов Рунге-Кутты, основная идея которых заключается в замене нелинейных систем последовательностью линейных. На практике, как правило, используются методы Розенброка четвертого порядка, реализованные, например, в программах *ROS4*, *RODAS*. В программном пакете *MathCAD* для приближения жестких задач реализованы функции *stiffr* на основе метода Розенброка и *stiffb* – на основе метода Булирша-Штера. В *MATLAB* функция *ode23s* реализует одношаговый метод, использующий модифицированную формулу Розенброка 2-го порядка. При этом решатели, основанные на методе Рознеброка, оценивают якобиан на каждом шаге интегрирования, поэтому решающим значением для их надежности и эффективности является наличие матрицы Якоби.

Среди итерационных схем лучшими являются неявные s-стадийные методы Рунге-Кутты [81]. Для автономной системы (1.14) они имеют следующий вид

$$y_{n+1} = y_n + h \sum_{m=1}^{s} b_m w_m \,,\; w_m = f\left(y_n + h \sum_{j=1}^{s} \alpha_{mj} w_j\right),\; m = \overline{1,\, s}\,.$$

Если матрица коэффициентов α_{mj} поддиагональна, то схема явная и не пригодна для жестких задач, если суммирование идет до m, то схема называется диагонально-неявной (*DIRK*) [83]. Такие схемы для нахождения каждого k_m требуют решения системы нелинейных уравнений, размерность которой равна размерности вектора y (таким образом, решение искомой задачи снижения трудоемкости приводит к соизмеримой по трудоемкости задаче решения системы нелинейных уравнений). При суммировании до s для нахождения k_m приходится решать систему нелинейных уравнений в s раз большей размерности, что является достаточно трудоемкой задачей при

большой размерности системы. Поэтому на практике ограничиваются диагонально-неявными методами. В [3] приводятся коды программ *RADAU5*, *SDIRK4*, реализующих неявные методы Рунге-Кутты. Помимо того, для приближения жестких задач применяются экстраполяционные методы. Широко распространены программы *SEULEX* и *SODEX*.

Как правило, эффективность методов решения жеских задач сравнивают на тестовых задачах, одной из таковых является описанная выше задача (1.10), моделирующая периодическую реакцию Белоусова-Жаботинского. Как уже отмечалось, экстраполяционная программа *SEULEX* дает приближение решения задачи (1.10) с погрешностью 10^{-9} [3]. Конструируемый в главе 2 кусочно-интерполяционный метод с итерационным уточнением дает приближение решения этой же задачи с погрешностью 10^{-13}.

В области небесной механики, как отмечалось выше, эффективным является подход, позволяющий на каждом шаге интегрирования строить алгебраический многочлен, аппроксимирующий правую часть дифференциальных уравнений, и получать решение задачи также в виде многочлена. Идея излагается в работе Эверхарта [5], где строится неявный одношаговый метод специально для решения астрономических задач. В дальнейшем Эверхарт дал обобщение метода для численного решения любых ОДУ первого и второго порядка [13, 14]. Предложенный в этих работах интегратор широко используется в практических расчетах при решении задач небесной механики. Близкие идеи приводятся в работе [15], где выполнена теоретическая разработка метода численного интегрирования ОДУ на основе аппроксимации правой части ДУ на шаге интегрирования алгебраическим многочленом и последующего его интегрирования. Алгоритм получения коэффициентов этого многочлена такой же, как в методе Эверхарта, однако в работе дается подробный вывод и оригинальный подход к решению этой проблемы. В частности, выбор разбиения шага интегрирования предлагается выполнять с помощью узлов квадратурной формулы Маркова. Новая

программная реализация интегратора Гаусса-Эверхарта с некоторыми видоизменениями алгоритма предлагается в [16]. К идеям этих работ примыкает работа [17, 18], в которой авторы предлагают реализацию подхода на основе приближения решения и его производной частичными суммами смещенных рядов Чебышева, при этом, коэффициенты рядов определяются с помощью итерационного процесса с применением квадратурной формулы Маркова. Преимущество в точности приближения задач небесной механики методом Эверхарта дополнительно подтверждает эксперимент по приближению решения тестовой задачи (1.14). Применение интегратора Гаусса-Эверхарта 19-го порядка, *GAUSS_32* [16], программно реализованного в среде *Delphi*, позволило получить приближение решения на отрезке $[0, 6\pi]$ с границей абсолютной погрешности порядка 10^{-16}. При этом следует отметить, что метод адаптирован специально для решения задач небесной мехнаники, в частности, механизм выбора величины шага интегрирования осуществлен с учетом специфики именно плоской задачи двух тел [16]. Для других задач, в частности, описанных выше, метод не дает такой эффективности. Как уже отмечалось, конструируемый в диссертационной работе метод дает решение этой же задачи с погрешностью, не превышающей порядок 10^{-17}. При этом метод не специализируется на решении задач небесной механики, а является инвариантным для различных классов задач, включая жесткие.

Естественно, мы лишь коснулись множества проблем, связанных с точностью численного моделирования на основе решения задачи Коши для систем ОДУ. Аналогичную проблему, примыкающую к проблеме повышения точности решения задачи Коши для систем ОДУ, охарактеризуем для дифференциальных моделей, описываемых уравнениями в частных производных. При этом основное внимание будет уделено численным методам решения задачи Коши для этих уравнений.

Данный выбор анализа исследований обусловлен тем, что в диссертационной работе основной результат вначале излагается для кусочно-

интерполяционного приближения функции одной переменной и для систем ОДУ, а затем результат обобщается на случай функции многих переменных и переносится на ДУ в частных производных.

1.3. Моделирование в ИВС периодических процессов на основе уравнений в частных производных. Математические модели многих классов физических явлений в природе, технике и технологии описываются с помощью ДУ в частных производных. Из разнообразия возможных примеров отметим некоторые уравнения гиперболического типа: нестационарные уравнения течения несжимаемой жидкости в поле силы тяжести в приближении мелкой воды, нестационарные уравнения течения несжимаемой жидкости в упруго деформируемых трубах, динамические уравнения переноса энергии в электроэнергетических и радиотехнических системах, интенсивных информационных потоков в телекоммуникационных и компьютерных сетях и многие другие [84]. Основным инструментом исследования таких моделей, как и в случае ОДУ, является численное моделирование [85 – 87]. Ниже приводятся актуальные процессы из различных областей науки, при моделировании которых необходимы методы приближенного решения начальных и начально-краевых задач для систем ДУ в частных производных, отличающиеся высокой точностью и одновременно сравнительно малой трудоемкостью.

1.3.1. Методы обработки данных в моделях распространения длинных волн. Традиционной задачей механики жидкости является динамика нелинейных волн на поверхности однородной и несжимаемой жидкости [88, 89]. Как отмечено в [90] «практическая важность таких исследований обусловлена опасным характером больших морских волн. Особую опасность имеют так называемые длинные волны (их длина превышает глубину бассейна) из-за их воздействия на прибрежные населенные пункты, портовые и береговые сооружения».

Процесс распространения длинных волн принято моделировать с использованием «уравнений мелкой воды, записанных в форме дифференциальных законов сохранения» [90, 91]. Закон сохранения массы [92]:

$$\frac{\partial H}{\partial t} + \frac{\partial Hu}{\partial x} = 0, \tag{1.16}$$

где $H(x,t)$ – полная глубина бассейна (сумма невозмущенной глубины бассейна и смещения), $u(x,t)$ – горизонтальная скорость волны, Hu – расход, определяющий поток воды через сечение. Закон сохранения количества движения:

$$\frac{\partial Hu}{\partial t} + \frac{\partial}{\partial x}(Hu^2 + 0.5gH^2) = gH\frac{dh}{dx}, \tag{1.17}$$

где $h(x)$ – невозмущенная глубина бассейна, g – ускорение свободного падения. Численное моделирование (1.16) и (1.17) «используется как для анализа гладких волновых процессов, так и для описания ударных волн (гидравлических прыжков)» [84]. При численном моделировании в данной области широко применяются различные системы компьютерной математики, в частности *Maple* и *MATLAB*, а также специализированные программные пакеты, например, *CLAWPACK*. В программном пакете *CLAWPACK* приближенное решение систем уравнений мелкой воды рассчитывается на основе метода конечных объемов, реализованного по схеме Роу [90]. Сравнение результатов численного решения системы (1.16), (1.17) в программном пакете *CLAWPACK* с линейной теории мелкой воды представлено на рис. 1.13 [92]. Рисунки даны для следующих значений параметров: амплитуда начальной волны – 1 м, длина волны равна 5 км в основании.

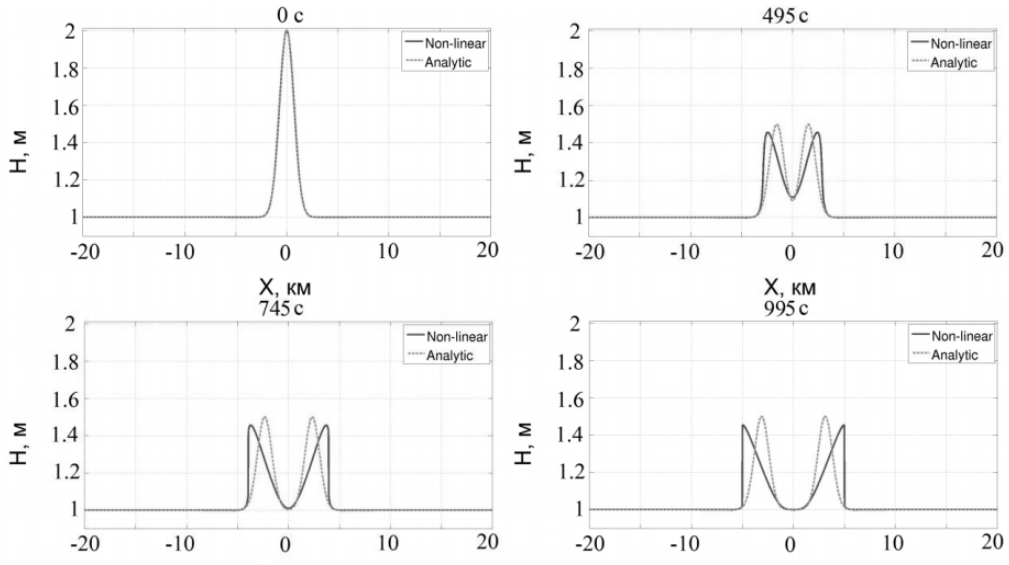

Рис. 1.13. Сравнение результатов численного решения системы (1.16), (1.17) с линейной теорией

Отличие в результатах моделирования составляет образование ударных фронтов при численном решении системы нелинейных уравнений (рис. 1.13). При этом, точки оснований волн в линейном и нелинейном случае совпадают, что согласуется с результатами теоретических исследований [93].

Узкая специализированность большинства алгоритмов численного решения систем ДУ в частных производных требует постоянных изменений алгоритмов и их программных реализаций. Так, в программной пакете *CLAWPACK 4.3* не удается моделировать движение воды по сухому берегу в связи с тем, что основным требованием к системе уравнений для их решения в данном программном пакете является ее гиперболичность. То есть, матрица A при записи системы уравнений в неконсервативной форме,

$$q_t + A(q,x,t)\,q_x = \psi(x,t),$$

должна иметь действительные и различные собственные значения. А в случае $H=0$, как показано в [92], для системы (1.16), (1.17) характеристики сливаются. В связи с этим в версии *CLAWPACK 4.6.2*, выпущенной в 2012 году, изменены алгоритмы вычисления решения уравнений мелкой воды, их подробное описание представлено в [94]. В новой версии помимо метода конечных объемов в программном пакете также реализован метод конечных

разностей Лакса-Вендроффа второго порядка [95]. Для аппроксимации системы нелинейных уравнений мелкой воды в задачах гидродинамики также часто используются схемы Мак-Кормака, Кранка-Николсон и Аракавы [96].

Достаточно полно отражает особенности моделирования распространения длинных волн уравнение Бюргерса. В рассматриваемом ниже частном случае уравнение обычно называют уравнением Ван-Хопфа

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0. \qquad (1.18)$$

Наличие точных решений задачи Коши для уравнения (1.18), описывающих формирование ударной волны, слияние и распад разрывов, периодические волны и другие физические эффекты [97, 98], позволяет эффективно тестировать численные методы решения ДУ в частных производных, описывающих данные явления [99, 100]. На рис. 1.14 представлены результаты моделирования процесса формирования ударной волны, которая описывается уравнением (1.18) и образуется из уединенной волны с гладким начальным профилем $u(x,0) = ch^{-2}(x-2)+1$.

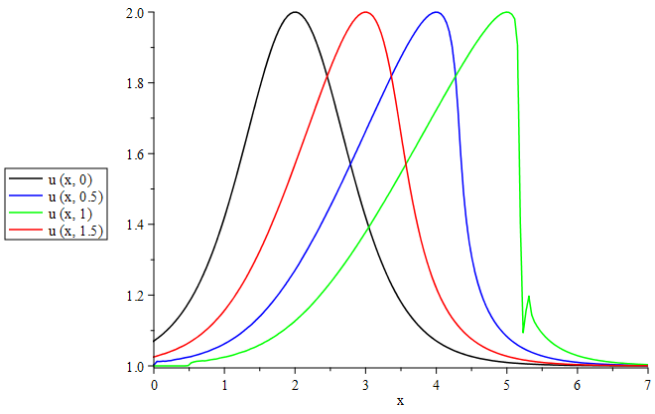

Рис. 1.14. Численное моделирование процесса формирования ударной волны с применением центрированной по времени неявной схемы (метода Кранка-Николсон)

Вследствие наличия разрыва при $t = 1.5$ на графике численного решения наблюдаются небольшие осцилляции. Положение точки разрыва для уравнения (1.18) может быть определено из системы [101]

$$s(t) = \xi_1 + \varphi(\xi_1)t,$$
$$s(t) = \xi_2 + \varphi(\xi_2)t,$$
$$\frac{\varphi(\xi_1) + \varphi(\xi_2)}{2} = \frac{1}{\xi_2 - \xi_1}\int_{\xi_1}^{\xi_2} \varphi(\xi)\, d\xi,$$

где $x = s(t)$ – линия разрыва функции $u(x,t)$, $\xi_1 = \xi_1(t)$, $\xi_2 = \xi_2(t)$, $\varphi(\xi)$ – начальный профиль волны. Применение рекурсивной 5-точечной разностной схемы сместило область осцилляции численного решения. В [99] уравнение (1.18) использовалось для тестирования численных методов решения систем ОДУ с применением метода прямых. Ввиду тесной связи с соответственным разделом диссертации отметим некоторые особенности метода прямых. В отличие от метода сеток в методе прямых только часть независимых переменных подвергается «квантованию», то есть взамен непрерывных областей их изменения вводятся дискретные множества значений, а производные по соответствующим направлениям заменяются приближающими их выражениями [102]. Другая часть аргументов и отвечающие им производные сохраняются, и исходная дифференциальная задача сводится, таким образом, к другой, также дифференциальной задаче, но меньшей размерности (в методе сеток получается система алгебраических уравнений). Впервые эту идею использовал Э. Роте [103] в 1930 г. для уравнений параболического типа. В дальнейшем применимость метода прямых (как и метода сеток, предельным случаем которого он является) была обоснована для широкого класса уравнений, в том числе не классических типов [12, 102]. Для уравнения (1.18) в методе прямых вводят пространственную сетку $\{x_n, 0 \le n \le N\}$ и заменяют производные по пространству разностными отношениями, сохраняя производную только по времени. При этом получается система большого числа ОДУ

$$\frac{dv}{dt} = f(v), \quad v = \{v_n, 1 \le n \le N\}, \; f = \{f_n, 1 \le n \le N\},$$

решение которой $v_n(t)$ аппроксимирует $u(x_n,t)$ из (1.18). Возникающие при применении метода прямых системы ОДУ зачастую оказываются жесткими, поэтому для их решения используют неяные методы РК, кратко описанные выше.

Таким образом, при моделировании распространения длинных волн, в частности, их трансформации в прибрежной зоне и наката на берег, актуальна разработка эффективных численных методов решения задачи Коши для ДУ в частных производных, описывающих данные явления. Кроме того, целесообразными могут оказаться методы решения ОДУ, сохраняющие высокую точность при решении жестких задач.

1.3.2. Методы обработки данных в ИВС для моделей переходных и турбулентных течений. Схожие проблемы вычислительной гидродинамики возникают при моделировании переходных и турбулентных течений в аэрокосмической промышленности. Наиболее перспективным методом моделирования данных процессов в настоящее время является метод крупных вихрей (*LES*, *large eddy simulation*) [104]. Впервые этот подход был использован в 1970 году Дирдорфом [105] для моделирования турбулентного течения в канале, а затем для моделирования атмосферного пограничного слоя [106]. Метод требует больших вычислительных затрат и при этом часто не достигается требуемая при моделировании реальных процессов точность и надежность вычислений [104]. С целью преодоления данных проблем специалисты в области вычислительной математики разрабатывают новые методы численного решения уравнений в частных производных. Так, в [104] предлагается серия методов и схем для повышения эффективности, точности и надежности моделирования методом крупных вихрей. В частности, в работе представлен новый класс схем дискретизации высокого порядка для уравнений Эйлера и Навье-Стокса, называемых энтропийно-устойчивыми гибридизированными разрывными методами Галеркина, вводится «немодальная» теория анализа, характеризующая численное рассеяние

предлагаемых схем дискретизации высокого порядка. Схема анализа поясняется на примере задачи Коши для линейного уравнения конвекции-диффузии с постоянными коэффициентами в одномерной области $G = (-\infty,\ \infty)$:

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2},\quad u(x,0) = u_0, \tag{1.19}$$

где a – скорость переноса, $\nu \geq 0$ – коэффициент диффузии, u_0 – дважды непрерывно дифференцируемая функция начального условия. С целью пространственной дискретизации гибридизированным разрывным методом Галеркина задача (1.19) записывается в форме

$$q = \frac{\partial u}{\partial x},\quad \frac{\partial u}{\partial t} + \frac{\partial F(u)}{\partial x} + \frac{\partial G(q)}{\partial x} = 0,\quad u(x,0) = u_0, \tag{1.20}$$

где q – вспомогательная градиентная переменная, $F(u) = a\,u$ и $G(q) = -\nu\,q$ – конвективная и диффузионная составляющие. После разбиения области G на однородные неперекрывающиеся элементы одинакого размера численное решение и его градиент в каждом элементе разбиения аппроксимируются полиномиальными разложениями.

Представленный в [104] численный эксперимент показывает целесообразность применения аппроксимации полиномами высоких степеней при доминировании диффузионной составляющей. Результаты проведенного анализа распространяются на нелинейную постановку: представлено его применение к уравнениям Бюргерса, Эйлера и Навье-Стокса.

В зависимости от специфики решаемой задачи целесообразно применение и других подходов для повышения точности моделирования турбулентных течений. В частности, в [107] при моделировании крупных вихрей в дозвуковой изотермической турбулентной струе «дискретизация основных уравнений проводится с помощью метода контрольного объема и разностных схем повышенной разрешающей способности по времени и пространству. Для дискретизации по времени используется метод Рунге-Кутты

третьего порядка. Вектор потока расщепляется на невязкую и вязкую составляющие. Для дискретизации невязких потоков применяются метод кусочно-параболической реконструкции и схема Чакраварти-Ошера, а для дискретизации вязких потоков – центральные конечно-разностные формулы второго порядка» [107].

Таким образом, в данной предметной области, равно как и в описанных ранее, актуальна разработка эффективных численных методов решения задачи Коши для ДУ в частных производных. При этом целесообразными могут оказаться методы, реализующие аппроксимации полиномами высоких порядков.

1.3.3. Моделирование в ИВС распространения гемодинамических импульсов. Проблемы разработки эффективных численных методов существуют и при моделировании движения жидкости в эластичных сосудах. Моделирование этих процессов имеет широкое научное и практическое распространение. Медико-физиологическое значение моделирования данного процесса связано с тем, что такие модели описывают течение крови в кровеносных сосудах. Симптомами ряда заболеваний кровеносной системы служат нарушения в распространении волны давления (в биомеханике ее называют пульсовой волной) [108, 109]. Особое внимание уделяется математическому моделированию динамики распространения пульсовой волны. Уравнения гемодинамики в квазиодномерном приближении представляют собой систему двух ДУ в частных производных (выражающих законы сохранения), которая замыкается одним алгебраическим соотношением, связывающим площадь поперечного сечения сосуда с трансмуральным давлением (разницей давлений внутри и снаружи сосуда) в сосуде. ДУ, выражающее уравнение неразрывности, имеет вид [110]

$$\frac{\partial S}{d t}+\frac{\partial u S}{d x}=0\,. \qquad (1.21)$$

Закон изменения количества движения приводит к ДУ

$$\rho \frac{\partial u S}{d t} + \rho \frac{\partial u^2 S}{d x} + \frac{\partial p S}{d x} - p \frac{\partial S}{d x} = q_f S,$$

которое преобразуется к виду

$$\frac{\partial u}{d t} + u \frac{\partial u}{d x} + \frac{1}{\rho} \frac{\partial p}{d x} = \frac{q_f}{\rho}, \tag{1.22}$$

где q_f – объемная плотность внешних сил. В качестве пространственной переменной x в (1.21) и (1.22) выбрана длина дуги, проходящей через центры круговых поперечных сечений сосуда, $S(x,t)$ – площадь попереченого сечения сосуда, $u(x,t)$ – скорость движения крови, $p(x,t)$ – давление внутри крови, ρ – плотность крови, принятая в качестве постоянной величины. Для построения уравнения состояния, замыкающего уравнения (1.21), (1.22) используется дополнительное соотношение

$$S = S(p), \tag{1.23}$$

являющееся экспериментально устанавливаемой зависимостью между площадью поперечного сечения сосуда и давлением внутри сосуда. Простейший вид модельного уравнения состояния определяется формулой

$$S(p) = \begin{cases} S_{\min} + \dfrac{S_{\max} - S_{\min}}{p_{\max} - p_{\min}} (p - p_{\min}), & p_{\min} < p < p_{\max}, \\ S_{\min}, \ p \le p_{\min}, \\ S_{\max}, \ p \ge p_{\max}, \end{cases}$$

где $S_{\min}$, $S_{\max}$, $p_{\min}$, $p_{\max}$ являются характеристиками конкретного сосуда. Уравнения (1.21) – (1.23) в совокупности представляют собой систему уравнений гемодинамики в квазиодномерном приближении. В нормальных условиях гемодинамика кровотока в сосуде удовлетворительно описывается уравнениями в линеаризованной постановке [111]:

$$\begin{cases} \dfrac{\partial p}{d t} + \bar{u}\dfrac{\partial p}{d x} + \rho\bar{c}^2\dfrac{\partial u}{d x} = 0, \\ \dfrac{\partial u}{d t} + \bar{u}\dfrac{\partial u}{d x} + \dfrac{1}{\rho}\dfrac{\partial p}{d x} = 0, \end{cases} \tag{1.24}$$

где p и u – малые отклонения от постоянного течения со среднефоновыми значениями параметров $\bar{p}$ и $\bar{u}$, $\bar{c} = \sqrt{\bar{S}/\rho\bar{\chi}}$ – скорость распространения малых возмущений, $\bar{\chi} = \dfrac{dS}{dp}$ – числовой параметр, характеризующий эластичность сосуда. Использование линейного приближения допустипо, когда отклонения давления скорости крови от среднефоновых значений, оказываются ниже 20 – 30 %.

В зависимости от постановки задачи в процессе исследования гемодинамики в кровеносном сосуде возникает необходимость численного решения уравнений переноса в широком диапазоне значений шага дискретизации по времени с применением как явных, так и неявных разностных схем [112]. В [113] обсуждается гибридная квазилинейная схема второго порядка аппроксимации на основе использования локальных линейных сплайнов. Представленные в работе результаты расчетов демонстрируют возможность применения данной схемы для численного моделирования распространения гемодинамических импульсов. Схема описана для линейного уравнения переноса, однако отмечается, что она может быть обобщена и на случай нелинейного уравнения переноса. Другие подходы к построению численных методов приближенного решения уравнения переноса представлены в [26, 27, 114, 115].

Отметим близкую к тематике диссертации работу [116], в которой также рассматривается одномерное уравнение переноса и предлагается разностный метод с адаптивным изменением величины шага по времени и пространству. При этом область решения предварительно разбита на блоки, в которых разностная сетка уточняется или укрупняется на основе оценки локальной

ошибки, интегрирование по времени выполняется с использованием метода Рунге-Кутты-Фельберга. На примере различных модельных задач Коши для уравнений в частных производных показана возможность их приближенного решения с абсолютной погрешностью не превышающей порядка $10^{-4} - 10^{-3}$.

На основании изложенного можно сделать вывод о том, что при исследовании широкого класса актуальных процессов, в частности, в области гидродинамики используются линейные и квазилинейные модели переноса.

Таким образом, выполненный анализ подтвердил актуальность разработки и исследования высокоточных методов обработки данных, которые бы характеризовались сравнительно малой временной сложностью при моделировании периодических процессов, в частности, в области химической кинетики, планетной астрономии, астрофизики и периодического возмущенного движения КА. Анализ также показал, что в исследуемых областях существует проблема построения гладких аналитических приближений данных дифференциальных моделей. Для моделей процессов в перечисленных областях важно получение данных как в реальном времени, так и в произвольной точке области приближения.

1.4. Выводы

1. Выполнен анализ современного состояния проблем обработки данных моделей периодических процессов, представляющих собой объекты актуальных научно-технических исследований. Основное внимание уделено процессам и методам обработки данных, к которым предъявляются требования высокой точности и одновременно малой трудоемкости. Отмечены особенности обработки данных моделей периодических процессов химической кинетики, астрофизики, планетной астрономии, возмущённого движения КА, затрудняющие выполнение высокоточного быстродействующего моделирования на основе известных методов и как следствие приводящие к необходимости создания специализированных методов.

2. Представлена детализация наиболее распространенных алгоритмов обработки данных в ИВС моделей периодических процессов, характеризующихся высокой степенью жесткости, в частности, в области химической кинетики, при моделировании автоколебательных химических реакций. Детализированы высокоточные алгоритмы решения нежестких задач с целью исследования особенностей их применения для обработки данных с качествами гладкости в дифференциальных моделях.

3. Выполнен анализ проблем обработки данных в ИВС для моделей с частными производными. Представлены примеры задач в области гидродинамики, при моделировании которых актуальны проблемы повышения точности обработки данных, в частности, для модели переноса.

4. Показано, что при исследовании периодических процессов актуальна проблема построения гладких аналитических приближений для обработки данных дифференциальных моделей. При этом для моделей процессов в рассмотренных областях актуально требование получения данных в реальном времени при условии высокой точности.

ГЛАВА 2. МЕТОД ВАРЬИРУЕМОЙ РАЗНОСТНО-ПОЛИНОМИАЛЬНОЙ ОБРАБОТКИ ДАННЫХ В ИВС С АВТОМАТИЧЕСКИМ ВЫБОРОМ ПАРАМЕТРОВ ДЛЯ МОДЕЛЕЙ ПЕРИОДИЧЕСКИХ ПРОЦЕССОВ

В главе излагается метод разностно-полиномиальной обработки данных в ИВС с программным выбором варьируемых параметров для моделей периодических процессов. Метод отличается от аналогов обработкой данных на временных подынтервалах интерполяционными полиномами Ньютона, программно преобразуемыми в форму алгебраических полиномов с числовыми коэффициентами. В качестве значений зависимой переменной при вычислении значений функции в узлах интерполяции используются разностные приближения решения. Обработанные данные итерационно уточняются (по аналогии с методом Пикара) [117]. На каждом временном интервале выполняется автоматизированный программный выбор параметров метода, что позволяет повысить точность и уменьшить время обработки данных относительно известных методов, а также улучшить качество моделирования исследуемых процессов.

Выполнено исследование сходимости предложенного метода, даны оценки скорости сходимости. Выполнена алгоритмизация и программная реализация метода, представлены результаты численного эксперимента для моделей жестких и нежестких задач. Реализация метода выполнена на персональном компьютере в среде *Delphi*.

Непосредственно ниже излагается базовая конструкция для излагаемого подхода: быстродействующее компьютерное приближение функций с вариацией параметров для обработки данных в ИВС, которое отличается повышенной точностью, непрерывностью и непрерывной дифференцируемостью, сходимостью со скоростью геометрической прогрессии [74].

Материал главы соответствует содержанию публикаций автора [118 – 123, 129 – 132].

2.1. Быстродействующее компьютерное приближение функций с вариацией параметров для обработки данных в ИВС с повышенной точностью. Ставится задача компьютерного вычисления действительной функции одной действительной переменной с априори заданной границей погрешности при одновременной минимизации временной сложности. Решение строится на основе кусочно-полиномиальной аппроксимации, на подынтервале выбирается интерполяционный полином Ньютона, который преобразуется к форме полинома с числовыми коэффициентами. В результате композиция стандартных функций вычисляется за время $O(1)$ с точностью до 10^{-19} – 10^{-18}. При данной компьютерной аппроксимации подынтегральных функций повышается точность вычисления интеграла.

Аппроксимация действительной функции $u = u(x)$ от одной действительной переменной на произвольном отрезке $[\alpha, \beta]$ выполняется следующим образом [73, 74]. Выбирается система подынтервалов равной длины, объединение которых покрывает отрезок $[\alpha, \beta]$:

$$[\alpha, \beta] = \bigcup_{i=0}^{P-1} [x_i,\ x_{i+1}], \tag{2.1}$$

для определенности полагается $P = 2^k,\ k \in \{0, 1, \ldots\}$. Пусть априори задана граница ε абсолютной погрешности аппроксимации данной функции. При каждом i из (2.1) на i-м подынтервале строится интерполяционный полином Ньютона $\Psi_{in}(t)$ с равноотстоящими узлами, где $t = (x - x_i)/h,\ h = (x_{i+1} - x_i)/n$ – расстояние между узлами. Степень полинома n выбирается одинаковой для всех подынтервалов и минимальной при условии:

$$|u(x) - \Psi_{in}(t)| \le \varepsilon,\quad x \in [x_i, x_{i+1}],\quad i = \overline{0, P-1}. \tag{2.2}$$

Интерполяционный полином Ньютона преобразуется к виду:

$$\Psi_{in}(t) = a_{i0} + a_{i1} t + a_{i2} t^2 + \ldots + a_{in} t^n,\ i = \overline{0, P-1}, \tag{2.3}$$

где $x \in [x_i, x_{i+1}]$, $t = \dfrac{x - x_i}{h}$. Для этого в его исходной записи

$$\Psi_{in}(t) = u(x_{i0}) + \sum_{j=1}^{n} \frac{\Delta^j u_{i0}}{j!} \prod_{k=0}^{j-1} (t-k),\ t = \frac{x - x_{i0}}{h}, \tag{2.4}$$

где $x_{ij} = x_i + jh$, $j = \overline{0, n}$, – узлы интерполяции, $\Delta^j u_{i0}$ – конечная разность j-го порядка в узле $x_{i0} = x_i$, вычисляются конечные разности $\Delta u_{i0} = u(x_{i1}) - u(x_{i0})$, $\Delta^r u_{i0} = \Delta^{r-1} u_{i1} - \Delta^{r-1} u_{i0}$, $r = \overline{2, n}$. В обозначениях $b_{ij} = \Delta^j u_{i0}$, $j = \overline{1, n}$,

$$\Psi_{in}(t) = u(x_{i0}) + \sum_{j=1}^{n} \frac{b_{ij}}{j!} \prod_{k=0}^{j-1} (t-k). \tag{2.5}$$

Произведение $P_j(t) = \prod_{k=0}^{j-1} (t-k)$ является полиномом, заданным разложением на множители, где $k = \overline{0, j-1}$ – его нули, по которым можно восстановить коэффициенты:

$$P_j(t) = d_{j0} + d_{j1} t + d_{j2} t^2 + \ldots + d_{jj} t^j. \tag{2.6}$$

В обозначении $z_\ell = \ell$, $\ell = \overline{0, j-1}$, применяется отличная от формул Виета схема выражения коэффициентов полинома через его нули [75], которая строится на основе реккурентного увеличения степени полинома

$$P_k(t) = d_{k0} + d_{k1} t + \ldots + d_{kk} t^k = (t - z_0)(t - z_1) \ldots (t - z_{k-1}),$$

где d_{kr} – найденные на k-м шаге коэффициенты, $r = \overline{0, k}$; $k = \overline{1, j}$. При $k = j$ определятся все искомые коэффициенты полинома (2.6). Пусть вычислено $P_1(t) = t - z_0$ или $P_1(t) = d_{11} t + d_{10}$, где $d_{10} = -z_0, d_{11} = 1$. Тогда в аналогичных обозначениях можно найти $P_2(t) = (t - z_0)(t - z_1) = P_1(t)(t - z_1)$ по схеме $P_2(t) = d_{22} t^2 + d_{21} t + d_{20}$, где $d_{22} = d_{11}$, $d_{21} = d_{10} - d_{11} z_1$, $d_{20} = -d_{10} z_1$.

Если уже вычислены коэффициенты полинома $(k-1)$-й степени

$$P_{k-1}(t) = d_{(k-1)(k-1)} t^{k-1} + d_{(k-1)(k-2)} t^{k-2} + \ldots + d_{(k-1)1} t + d_{(k-1)0},$$

коэффициенты полинома k-й степени

$$P_k(t) = \prod_{r=0}^{k-1} (t - r) = P_{k-1}(t)\,(t - z_{k-1}),$$

определяются по аналогичной схеме

$$\left.\begin{aligned}
&d_{kk} = d_{(k-1)(k-1)},\\
&d_{k(k-1)} = d_{(k-1)(k-2)} - d_{(k-1)(k-1)}\, z_{k-1},\\
&d_{k(k-2)} = d_{(k-1)(k-3)} - d_{(k-1)(k-2)}\, z_{k-1},\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&d_{k(k-\ell)} = d_{(k-1)(k-\ell-1)} - d_{(k-1)(k-\ell)}\, z_{k-1},\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&d_{k0} = -d_{(k-1)0}\, z_{k-1},
\end{aligned}\right\} \qquad (2.7)$$

где $d_{kk} = d_{(k-1)(k-1)} = \ldots = d_{11} = 1$. В (2.7) $\ell = \overline{1, k-1}$ при изменении значений $k = \overline{2, j}$. В программной реализации алгоритм (2.7) представляется в виде двух циклов: внутренний реализует вычисление коэффициентов в строках при $\ell = \overline{1, k-1}$, а внешний цикл изменяет $k = \overline{2, j}$. При $k = j$ левые части (2.7) совпадут с искомыми значениями коэффициентов полинома (2.6). Данный алгоритм имеет матричную запись [76]:

$$\begin{pmatrix} d_{kk} \\ d_{k(k-1)} \\ d_{k(k-2)} \\ \cdots \\ d_{k1} \\ d_{k0} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ -z_{k-1} & 1 & 0 & \cdots & 0 & 0 \\ 0 & -z_{k-1} & 1 & \cdots & \cdots & \cdots \\ 0 & 0 & -z_{k-1} & \cdots & 0 & 0 \\ \cdots & \cdots & 0 & \cdots & 1 & 0 \\ \cdots & \cdots & \cdots & \cdots & -z_{k-1} & 1 \\ 0 & 0 & 0 & \cdots & 0 & -z_{k-1} \end{pmatrix}}_{k} \times \begin{pmatrix} d_{(k-1)(k-1)} \\ d_{(k-1)(k-2)} \\ \cdots \\ d_{(k-1)1} \\ d_{(k-1)0} \end{pmatrix} \qquad (2.8)$$

Замена в (2.8) k на $k-1$ позволит в аналогичной форме представить вектор из правой части. Рекуррентный процесс таких замен и соответственных подстановок влечет явное выражение при $k=j$ коэффициентов полинома (2.6) через все его j нулей $z_\ell=\ell,\ \ell=\overline{0,\ j-1}$:

$$\begin{pmatrix} d_{jj} \\ d_{j(j-1)} \\ \cdots \\ d_{j0} \end{pmatrix} = \left(\underbrace{\begin{matrix} 1 & 0 \\ -z_{j-1} & 1 \\ 0 & -z_{j-1} \\ 0 & 0 \\ \cdots & \cdots \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{matrix}}_{j+1} \begin{matrix} 0 & 0 \\ 0 & 0 \\ \cdots & \cdots \\ \cdots & \cdots \\ 0 & 0 \\ 1 & 0 \\ -z_{j-1} & 1 \\ 0 & -z_{j-1} \end{matrix} \right)_{j} \times \left(\underbrace{\begin{matrix} 1 & 0 \\ -z_{j-2} & 1 \\ 0 & -z_{j-2} \\ 0 & 0 \\ \cdots & \cdots \\ 0 & 0 \\ 0 & 0 \end{matrix}}_{j} \begin{matrix} 0 & 0 \\ 0 & 0 \\ \cdots & \cdots \\ 0 & 0 \\ 1 & 0 \\ -z_{j-2} & 1 \\ 0 & -z_{j-2} \end{matrix} \right)_{j-1} \times$$

$$\times \begin{pmatrix} 1 & 0 & 0 \\ -z_2 & 1 & 0 \\ 0 & -z_2 & 1 \\ 0 & 0 & -z_2 \end{pmatrix} \times \begin{pmatrix} 1 & 0 \\ -z_1 & 1 \\ 0 & -z_1 \end{pmatrix} \times \begin{pmatrix} 1 \\ -z_0 \end{pmatrix}.$$

Данное выражение можно представить в свернутом виде:

$$\begin{pmatrix} d_{jj} \\ d_{j(j-1)} \\ \cdots \\ d_{j0} \end{pmatrix} = \prod_{\ell=1}^{j} \left(\underbrace{\begin{matrix} 1 & 0 & \ldots & 0 & 0 \\ -z_{j-\ell} & 1 & \ldots & 0 & 0 \\ 0 & -z_{j-\ell} & \ldots & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & -z_{j-\ell} & 1 \\ 0 & 0 & \ldots & 0 & -z_{j-\ell} \end{matrix}}_{j-\ell+1} \right). \qquad (2.9)$$

Каждый нуль полинома повторяется на поддиагонали одной и только одной матрицы из (2.9), диагональ всегда состоит из единиц. Число элементов

диагонали и поддиагонали совпадают, поскольку число строк матрицы на единицу больше числа столбцов. Размерность на единицу возрастает справа налево.

Легко убедиться, что последовательное справа налево умножение матриц (2.9) даст инвариантный относительно номера искомого коэффициента алгоритм (2.7). В листинге 2.1 приведена стандартная процедура *Viet*, реализующая описанный алгоритм (степень полинома обозначена переменной nn).

Листинг 2.1

```
procedure Viet;
var k,i:byte; q:array[0..nn] of extended; dd:array[0..nn,0..nn] of extended;
begin for k:=0 to nn-1 do q[k]:=k; dd[1,1]:=1; dd[1,0]:=-q[0];
for k:=2 to nn do begin dd[k,0]:=-dd[k-1,0]*q[k-1];
for i:=1 to k-1 do dd[k,k-i]:=dd[k-1,k-i-1]-dd[k-1,k-i]*q[k-1];
dd[k,k]:=dd[k-1,k-1] end;
for k:=1 to n do for i:=0 to k do d[i,k]:=dd[k,i] end;
```

Доказательство (2.9) и программная реализация для случая полинома с комплексными коэффициентами приводятся в работе [75]. Полученные с помощью процедуры *Viet* коэффициенты $d_{j\ell}$, $\ell = \overline{0,\ j}$, можно хранить в памяти компьютера для любой конечной степени j, что предполагается ниже. Предполагается также, что априори вычислены факториалы из (2.5) и значения $d_{j\ell}/j!$, $\ell = \overline{0,\ j}$, которые также хранятся в памяти. После данных преобразований полином (2.5) переводится в форму (2.3) по дистрибутивности с приведением подобных. В результате

$$\Psi_{in}(t) = a_{i0} + \sum_{\ell=1}^{n} a_{i\ell}\, t^{\ell}, \tag{2.10}$$

где

$$a_{i0} = u(x_{i0}),\ a_{i\ell} = \sum_{j=\ell}^{n} b_{ij}\, \frac{d_{j\ell}}{j!}. \tag{2.11}$$

Для (2.10) проверка точности приближения (2.2) на каждом подынтервале проводится дискретно в проверочных точках x_{ℓ}, расстояние между которыми

составляет величину h/γ, где γ – параметр, в программной реализации $\gamma \geq 3$. Минимальность степени n полинома (2.10) при условии (2.2) обеспечивается по следующейсхеме. Построение полинома и проверка (2.2) начинается с $n=1$при $k=0$ в (2.2), где $P=2^k$. При нарушении неравенства (2.2) хотя бы в одной проверочной точке значение k увеличивается на единицу. Этот процесс продолжается до нарушения априори заданной границы k, $k \leq k_0$. Если в результате заданная точность приближения не достигается, то снова полагается $k=0$, при этом степень полинома увеличивается на единицу и так до момента нарушения априори заданной границы значения n, $n \leq n_0$. В качестве минимального фиксируется наименьшее n, при котором неравенство (2.2) выполняется одновременно во всех проверочных точках всех 2^k подынтервалов, в соответствии с этим фиксируется текущее k.

При аппроксимации полиномом (2.10) функции $u(x)$, $x \in [\alpha, \beta]$, выполняется дешифрация номера подынтервала $i = \left[(x-\alpha)/\rho\right]$, $[\ ]$ – целая часть числа, $\rho = x_{i+1} - x_i$, $i = \overline{0, P-1}$, $x \in [x_i\ ,\ x_{i+1})$. Значение i служит адресом выборки соответственных подынтервалу коэффициентов (2.10).

Всюду ниже $n \leq n_0$, $n_0 = \text{const}$, степень n по изложенной схеме можно сделать «сколь угодно» малой; отсюда при условии запоминания коэффициентов (2.10) время вычисления функции путем данного приближения составляет $T = O(1)$. В частности, таким образом минимизируется время вычисления часто встречающихся или стандартных функций. При фиксированном n выполнение неравенства (2.2) обеспечивается за счет варьируемого увеличения количества подынтервалов $P=2^k$ и соответственного сокращения их длины. Например, вычисление $u(x) = \sqrt[3]{\text{arctg}\left(e^{\cos\sqrt[5]{(1/x)}}\right)}$, $x \in [0.5,\ 1]$, выполняется за время двух умножений со сложением с точностью приближения до 10^{-19} при количестве подынтервалов

$P = 2^{17}$. В приложении к главе 2 (п. 1), представлены листинг программы и результаты ее работы, в главе 5, п. 5.3, даны другие примеры, где функции вычисляются с точностью до $10^{-20} - 10^{-19}$.

Объяснение сравнительно высокой точности предложенного метода подробно описано в [178] и заключается в следующем. «Интерполяционные полиномы переводятся в форму алгебраических полиномов с числовыми коэффициентами на основе целочисленных преобразований их основных компонентов, соотношения (2.4) – (2.9). Эти компоненты становятся полиномами с целыми коэффициентами. Для компьютерной арифметики с плавающей точкой целочисленные операции наименее подвержены накоплению погрешности. Кроме того, переход к приближению функций на подынтервалах дает возможность минимизировать модуль остаточного члена интерполяции» [178]. Доказательство равномерной сходимости метода и оценки скорости сходимости даны непосредственно ниже.

«Арифметика с плавающей точкой искажает оценки погрешности многих классических методов, исторически и математически для такой арифметики не предназначавшихся. Эти методы строились над полем вещественных (комплексных) чисел в абстрактном понимании числа, без ограничения разрядности, при естественной записи разрядов с фиксированной точкой. В компьютерной реализации арифметики с плавающей точкой для выравнивания порядков выполняется сдвиг мантиссы в ограниченной разрядной сетке с неизбежной потерей значащих цифр числа. Это происходит при выполнении практически каждой арифметической операции. С алгебраическим полем вещественных (комплексных) чисел такая арифметика имеет значительные отличия. Классические алгоритмы не эквивалентны их компьютерной реализации и искажаются ею. Так, расположение узлов интерполяции в полиноме Лагранжа по схеме Чебышева [124] для наименьшего отклонения от нуля осточного члена интерполяции сохраняет вещественный тип узлов, которые предварительно вычисляются через тригонометрические функции. В

результате, применение этой схемы в компьютере для кусочной интерполяции элементарных функций не позволяет превысить точность приближения порядка 10^{-12}. Использование других классических полиномов [124] для кусочной интерполяции тех же функций даст точность приближения не выше 10^{-15} [125] (*Delphi*, *C++*)» [178]. Поэтому в дальнейшем такие реализации метода не рассматриваются.

2.1.1. Сходимость и скорость сходимости предложенного метода обработки данных. Следующие оценки показывают равномерную сходимость метода в зависимости от числа подынтервалов. Пусть на время рассуждения произвольно зафиксировано $n = \text{const}$, а функция $u = u(x)$ определена, непрерывна и имеет $n+1$ непрерывную производную на отрезке $[\alpha, \beta]$ из (2.1). На границах отрезка все рассматриваемые производные считаются односторонними. Полагая $P = 1,\ i = 0,\ [x_0, x_1] = [\alpha, \beta]$, погрешность приближения рассматриваемой функции на $[\alpha, \beta]$ интерполяционным полиномом Ньютона $\Psi_{0n}(t)$ можно оценить из неравенства [124]:

$$|u(x) - \Psi_{0n}(t)| \le \frac{\max\limits_{\xi \in [x_0, x_1]} |u^{(n+1)}(\xi)|\, h^{n+1}}{(n+1)!} \left| \prod_{\ell=0}^{n} (t - \ell) \right|,\ x \in [x_0, x_1], \qquad (2.12)$$

где $x_{0j} = x_0 + j\,h,\ j = \overline{0, n}$, поэтому $t = \dfrac{x - x_{00}}{h}$, $h = \dfrac{x_{0n} - x_{00}}{n}$. С учетом $P = 2^k$ это соответствует $k = 0$ в (2.1) и интерпретируется как нулевой этап разбиения $[\alpha, \beta]$. Оценку (2.12) можно преобразовать следующим образом. Пусть j – номер узла интерполяции, такой что $x \in [x_{0j}, x_{0j+1}]$. Поскольку $t - \ell = \dfrac{x - x_{0\ell}}{h},\ \ell = \overline{0, n}$, то из (2.12) следует неравенство

$$|u(x) - \Psi_{0n}(t)| \le \frac{c(n)\, h^{n+1}}{(n+1)!} \left| \prod_{\ell=0}^{j} \frac{x - x_{0\ell}}{h} \prod_{\ell=j+1}^{n} \frac{x - x_{0\ell}}{h} \right|,$$

где $c(n) = \max\limits_{\xi \in [x_0, x_1]} |u^{(n+1)}(\xi)|$. При данном выборе j выполняется $x_{0j} \le x \le x_{0j+1}$, поэтому

$$|u(x) - \Psi_{0n}(t)| \le \frac{c(n)\, h^{n+1}}{(n+1)!} \prod_{\ell=0}^{j} \frac{|x_{0j+1} - x_{0\ell}|}{h} \prod_{\ell=j+1}^{n} \frac{|x_{0j} - x_{0\ell}|}{h}.$$

С учетом того, что на k-м подынтервале $x_{0j+\ell} = x_{00} + (j+\ell)\, h$ и $x_{0j+\ell} = x_{0k} + (j+\ell-k)h$, $k = \overline{0,\, j+\ell}$, $\ell = \overline{0,\, n-j}$, последнее неравенство примет вид:

$$|u(x) - \Psi_{0n}(t)| \le \frac{c(n)\, h^{n+1}}{(n+1)!} (j+1)!\,(n-j)!.$$

Очевидно, $(j+1)!\,(n-j)! \le (n+1)!$; следовательно,

$$|u(x) - \Psi_{0n}(t)| \le c(n)\, h^{n+1},\ P = 1,\ [\alpha, \beta] = [x_0, x_1], \tag{2.13}$$

при любом $j = \overline{0,\, n-1}$. Пусть теперь $P = 2$ и на каждом подынтервале $[x_0, x_1]$, $[x_1, x_2]$ разбиения (2.1) выполняется интерполяция (2.10) с тем же $n = \text{const}$. Тогда на каждом из них в силу равенства числа узлов будет выполняться аналог неравенства (2.13) с вдвое меньшим h (узлы отстоят на $h/2$): $|u(x) - \Psi_{in}(t)| \le c(n) \left(\frac{h}{2}\right)^{n+1}$, $i = 0, 1$, или,

$$|u(x) - \Psi_{in}(t)| \le \frac{c(n)}{2^{n+1}} h^{n+1},\ P = 2,\ [\alpha, \beta] = \bigcup_{i=0}^{1} [x_i, x_{i+1}], \tag{2.14}$$

где $c(n)$ – постоянная с таким же значением, как и в (2.13), $t = \frac{x - x_i}{h/2}$ при каждом $i = 0, 1$. При этом в (2.14) и непосредственно ниже шаг h относится к узлам интерполяционного полинома на начальном отрезке $[\alpha, \beta]$ и не меняется при его разбиении на $P = 2^k$ подынтервалов, $k = 1, 2, \ldots$ В силу индукции [126]

удвоение числа подынтервалов разбиения (2.1) уменьшает дробь в правой части рассматриваемых оценок в 2^{n+1}, и при $k = \log_2 P$ справедливо неравенство:

$$|u(x) - \Psi_{in}(t)| \le c(n)\, 2^{-k(n+1)}\, h^{n+1},\ [\alpha, \beta] = \bigcup_{i=0}^{P-1} [x_i, x_{i+1}],\ P = 2^k. \qquad (2.15)$$

Из изложенного вытекает лемма.

Лемма 2.1. Пусть для произвольного $n = \text{const}$ функция $u = u(x)$ определена, непрерывна и непрерывно дифференцируема $n+1$ раз на отрезке $[\alpha, \beta]$, на концах которого подразумеваются соответственные односторонние производные. Тогда, каким бы ни было $n \ge 1$, последовательность полиномов $\Psi_{in}(t)$ из (2.10) равномерно сходится к функции $u(x)$ на данном отрезке при $k \to \infty$, где $k = \log_2 P$, P – число подынтервалов из (2.1). Скорость сходимости оценивается из (2.15), где $c(n) = \text{const}$, h – шаг интерполирования полинома $\Psi_{0n}(t)$ на $[\alpha, \beta]$ при $k = 0$.

С учетом $n \le n_0$, $n_0 = \text{const}$, имеет место следствие.

Следствие 2.1. В условиях леммы 2.1, а также в предположении $h < 1$ для любого i из (2.1) выполняется неравенство

$$|u(x) - \Psi_{in}(t)| \le \frac{c}{2^{2k}}\, h^2, \qquad (2.16)$$

где $c = \max\limits_{n \le n_0}(c(n))$, при этом $c = \text{const}$ не зависит от выбора степени полинома $n \le n_0$.

Доказательство следует из (2.15) заменой $c(n)$ на $\max\limits_{n \le n_0}(c(n))$ и n в $\dfrac{h^{n+1}}{2^{k(n+1)}}$ на минимальное значение с учетом $h < 1$.

Замечание 2.1. В условиях следствия 2.1 произвольно заданная граница абсолютной погрешности приближения функции достигается путем сужения подынтервалов независимо от порядка гладкости и выбора степени полинома –

$\forall\, \varepsilon > 0 \;\wedge\; \forall\, n\colon\; 1 \le n \le n_0$, для всех i из (2.1) согласно (2.16) выполняется неравенство $|u(x) - \Psi_{in}(t)| \le \varepsilon$, как только $0.5 \log_2\left(\frac{c}{\varepsilon}\right) \le k$.

Равномерная сходимость полиномов $\Psi_{in}(t)$ в виде (2.10) на $[\alpha, \beta]$ независимо от порядка гладкости функции $u(x)$, начиная с двукратной непрерывной дифференцируемости, соответствует результатам численных экспериментов [74, 127].

2.1.2. Временная сложность кусочно-полиномиального приближения функций в ИВС с параллельной архитектурой. При вычислении функций общего вида минимизация временной сложности может быть достигнута за счет преобразования набора алгоритмов к параллельной форме. Преобразование выполняется с использованием модели неветвящихся параллельных программ без учета обмена; временная сложность (время) выполнения алгоритма измеряется числом последовательных шагов и обозначается $T(R)$, где R – количество процессоров. Рассмотрим $u = u(x)$ на системе подынтервалов (2.1) в условиях (2.2). Полином вида (2.4) можно построить по максимально параллельной схеме с логарифмической оценкой временной сложности [74]. Именно, коэффициенты полинома (2.6) можно считать априори вычисленными согласно (2.9) для произвольно фиксированного n, аналогично можно считать вычисленными $q_{j\ell} = d_{j\ell} / j!$ для всех j из (2.11) при всех соответственных $\ell = \overline{1, n}$. В выражении разности j-го порядка

$$\Delta^j u_{i0} = u(x_{ij}) - j\, u(x_{i(j-1)}) + \frac{j(j-1)}{2} u(x_{i(j-2)}) + \ldots + (-1)^j u(x_{i0}) \qquad (2.17)$$

биноминальные коэффициенты C_ℓ^j также можно считать известными и хранимыми в памяти для всех требуемых j и n. Отсюда вычисление (2.17)

сводится к синхронному умножению C_ℓ^j на значения функции $u(x_{i\ell})$, $\ell = \overline{0,\ j}$, сложению произведений по схеме сдваивания одновременно для всех $\Delta^j u_{i0}$, $j = \overline{1,\ n}$, что потребует $R \sim n^2/2$ процессоров. В результате время вычисления всех b_{ij} из (2.11) составит $T_0\left(\frac{n^2}{2}\right) = t_y + \lceil \log_2(j+1) \rceil t_c$, где t_y, t_c – время бинарного умножения и сложения соответственно. Вычисление $a_{i\ell}$ сводится к синхронному умножению n пар b_{ij} на $q_{j\ell}$ и сложению по схеме сдваивания за время $\lceil \log_2(n-\ell) \rceil t_c$. Такие вычисления выполняются синхронно для всех ℓ из (2.10) с оценкой $T_1(R) = t_y + \lceil \log_2 n \rceil t_c$. В результате, $T(R) \sim 2\,(t_y + \lceil \log_2 n \rceil t_c)$ или

$$T\left(\frac{n^2}{2}\right) \sim 2\,(t_y + \log_2 n\, t_c) = O(\log_2 n). \qquad (2.18)$$

Данное построение воспроизводится параллельно по всем подынтервалам из (2.1), что приводит к пропорциональному росту числа процессоров [126]: $R = 2^{k-1} n^2$.

2.1.3. Приближение производных и интегралов в ИВС для обработки числовых данных в режиме реального времени. Из полинома вида (2.10) вытекает табличный вид производной и первообразной. Их можно использовать для аппроксимации производной $u'(x) \approx \frac{1}{h} \sum_{\ell=1}^{n} \ell\, a_{i\ell}\, t^{\ell-1}$, $t = \frac{x - x_i}{h}$, и определенного интеграла от функции по соотношению $\int_\alpha^\beta u(x)\,dx \approx h \sum_{i=0}^{P-1} \int_0^n \Psi_{in}(t)\,dt$ [74] или

$$\int_\alpha^\beta u(x)\,dx \approx h \sum_{i=0}^{P-1} \sum_{\ell=0}^{n} \frac{a_{i\ell}\, n^{\ell+1}}{\ell+1}. \qquad (2.19)$$

где $a_{i\ell}$ вычисляется согласно (2.11). На практике граница погрешности такого приближения производной имеет порядок 10^{-15}, определенного интеграла – 10^{-19} [74, 125]. Для приближения интеграла (2.19) этот результат согласуется с (2.15). Действительно, пусть в условиях леммы 2.1 выбрано произвольное $\varepsilon > 0$. Поскольку $\int_{\alpha}^{\beta} u(x)\,dx = \sum_{i=0}^{P-1} \int_{x_i}^{x_{i+1}} u(x)\,dx$, то

$$\left| \sum_{i=0}^{P-1} \int_{x_i}^{x_{i+1}} u(x)\,dx - \sum_{i=0}^{P-1} \int_{x_i}^{x_{i+1}} \Psi_{in}(t)\,dx \right| \le \sum_{i=0}^{P-1} \int_{x_i}^{x_{i+1}} |\,u(x) - \Psi_{in}(t)\,|\,dx .$$

Для любого $\varepsilon_1 > 0$, начиная с некоторого k_0, при всех $k \ge k_0$ выполняется неравенство $|\,u(x) - \Psi_{in}(t)\,| \le \varepsilon_1$; следовательно,

$$\sum_{i=0}^{P-1} \int_{x_i}^{x_{i+1}} |\,u(x) - \Psi_{in}(t)\,|\,dx \le \sum_{i=0}^{P-1} \int_{x_i}^{x_{i+1}} \varepsilon_1\,dx .$$

С учетом $\sum_{i=0}^{P-1} \int_{x_i}^{x_{i+1}} \varepsilon_1\,dx = (\beta - \alpha)\,\varepsilon_1$ при выборе $\varepsilon_1 = \dfrac{\varepsilon}{\beta - \alpha}$, получим неравенство

$$\left| \int_{\alpha}^{\beta} u(x)\,dx - \sum_{i=0}^{P-1} \int_{x_i}^{x_{i+1}} \Psi_{in}\left(\frac{x - x_i}{h} \right) dx \right| \le \varepsilon \quad \forall\ k \ge k_0,\ P = 2^k .$$

Например, функция $u = \dfrac{1}{1 + e^{2x}}$ на отрезке $[0, 1]$ изложенным методом вычисляется с точностью до 2.71×10^{-20}, граница погрешности ее производной 1.89×10^{-15}, определенный интеграл от нее на $[0, 1]$ вычисляется с точностью до 2.17×10^{-19}. По формуле Симпсона этот интеграл вычисляется с абсолютной погрешностью 1.74×10^{-14}. Сравнительная точность предложенной схемы реализуется в результате априорной минимизации погрешности подынтегральной функции посредством компьютерной вариации числа подынтервалов и степени полинома. Такая вариация непосредственно к методу

Симпсона не относится, отличительным [75] является также аналитическое выражение (2.19).

Ниже кусочно-полиномиальная схема видоизменяется для аппроксимации функций в правых частях ОДУ и для приближения решения с помощью первообразной от аппроксимирующего полинома. Аналогом (2.2) является условие минимальности отклонения полинома (2.10) от функции правой части. При фиксированном n уточнение решения будет достигаться увеличением числа подынтервалов (сужением их длины) на текущем интервале постоянной длины. За счет вариации n при варьируемом сужении подынтервалов достигается более высокая точность приближения решения в узлах интерполяции (и между ними) по сравнению с исходными значениями, которые априори задаются разностным методом.

2.2. Обработка разностных данных для дифференциальных моделей. Первоначально отрезок $[a, b]$ приближенного решения задачи Коши для системы ОДУ делится на интервалы равной длины. Начальное значение решения на текущем интервале полагается равным значению на конце предшествующего интервала. Аналогично (2.1) текущий интервал разбивается на 2^k подынтервалов равной длины, выбираются $n+1$ равноотстоящих узлов, строится интерполяционный полином Ньютона для аппроксимации функции правой части. В качестве значений зависимой переменной при вычислении значений функции в узлах интерполяции используются разностные приближения решения. Выполняются преобразования (2.4) – (2.9), в результате которых интерполяционный полином на подынтервале принимает вид (2.10). На этом же подынтервале определяется первообразная

$$\int \psi_{jn}(t)\,dt = C + h \sum_{\ell=0}^{n} \frac{a_{j\ell}}{\ell+1}\, t^{\ell+1}, \qquad (2.20)$$

для которой сохраняется переменным верхний предел и фиксируется нижний. За нижний предел принимается рассматриваемое приближение решения на

правой границе предыдущего подынтервала (на начальном подынтервале используются начальные данные задачи Коши). Первообразная (2.20) с определенной таким образом константой принимается за приближение искомого решения на рассматриваемом подынтервале. Ниже эта схема излагается с модификацией, заключающейся в программной вариации степени полинома и числа подынтервалов, которая обеспечивает наименьшую погрешность приближенного решения.

Пусть в евклидовом пространстве R_2 на произвольно фиксированном отрезке $[a^{(0)}, b^{(0)}]$ требуется приближенно решить задачу Коши для ОДУ первого порядка

$$y'(x) = f(x, y), \;\; y(x_0) = y_0. \tag{2.21}$$

Строится расширение $[a, b]$ этого отрезка: $a^{(0)} = a, \; b^{(0)} \le b$, где разность $b - a$ кратна величине ∇, равной длине текущего интервала. Для нежестких задач целесообразно выбрать $\nabla \approx 1$ (целесообразность выбора подтверждают результаты численного эксперимента, приведенные в [127, 128]). Предполагается, что на $[a, b]$ выполнены все условия существования и единственности, а также все условия сходимости рассматриваемого разностного метода (в данном случае метода Эйлера). Выполняется разбиение:

$$[a, b] = \bigcup_{i=0}^{R-1} [a_i, b_i], \; b_i - a_i = \nabla, \; i = \overline{0, R-1}. \tag{2.22}$$

Приближение решения на $[a, b]$ сводится к последовательному приближению на интервалах $[a_i, b_i]$, при каждом $i \ge 1$ полагается $y(a_i) = y(b_{i-1})$ и $y(a_0) = y_0$.

Для каждого i из (2.22) аналогично (2.1) задается система подынтервалов:

$$[a_i, b_i] = \bigcup_{j=0}^{P-1} [x_j, x_{j+1}],\ P = 2^k,\ k = \{0, 1, \ldots\}. \qquad (2.23)$$

Аналогично (2.2) – (2.11), на каждом подынтервале из (2.23) строится кусочно-полиномиальное приближение функции правой части (2.21). Количество подынтервалов $P = 2^k$ и степень интерполяционного полинома n выбираются так, чтобы было минимальным значение

$$\delta_{ij}(x) = |\ \psi_{jn}(x) - f(x, P_j(x))\ |,\quad x \in [x_j, x_{j+1}],\quad j = \overline{0, P-1}, \qquad (2.24)$$

где $\psi_{jn}(x) \approx f(x, y)$ из (2.21), $P_j(x) = y_j + \int_{x_j}^{x} \psi_{jn}(x)\,dx$ – полином с числовыми коэффициентами вида (2.20), приближающий искомое решение $P_j(x) \approx y(x)$. При этом узлы интерполяции на каждом подынтервале априори вычисляются по методу Эйлера с излагаемыми ниже особенностями.

Практически метод реализуется со следующим уточнением. Фиксируется начальное n, и при $k = 0$ на подынтервале $[x_0, x_1]$ из (2.23) в равноотстоящих точках с шагом, меньшим шага интерполяции, вычисляется $\delta_{i0}(x)$ из (2.24). Максимальное из этих значений, обозначаемое $\tilde{\delta}_i^{(k,n)}$, сохраняется в памяти компьютера с соответственными индексами i, k, n. Затем k увеличивается на единицу и построение воспроизводится при том же n на каждом подынтервале. При этом в памяти вместе с индексами i, k, n сохраняется максимальное значение $\tilde{\delta}_i^{(k,n)} = \max_j(\delta_{ij}(x))$ по всем $j = \overline{0, P-1}$ среди соответствующих j-му подынтервалу значений $\delta_{ij}(x)$. Процесс продолжается до нарушения допустимых (априори заданных) границ k. При их нарушении делается переход к минимальному k, но уже при увеличении n на единицу, процесс продолжается до нарушения априори заданных границ n. Среди всех сохраняемых значений выбирается $\min_{i=\text{const}} \tilde{\delta}_i^{(k,n)}$, соответствующие k и n фиксируются для i-го интервала из (2.22) как целевые. Именно при этих

значениях решение задачи (2.21) приближается на $[a_i, b_i]$ по данной кусочно-полиномиальной схеме. Согласно эксперименту [129, 130] для k и n целесообразны границы $k \le 10$, $n \le 15$ (часть эксперимента представлена в [127]), при этом достигается точность приближения решения $10^{-16} - 10^{-19}$ и выше (ниже даны примеры).

Коэффициенты кусочно-полиномиальной аппроксимации вычисляются следующим образом. При каждом j подынтервал $[x_j, x_{j+1}]$ из (2.23) разбивается на n равноотстоящих узлов с шагом h:

$$x_{jp} = x_j + ph, \; p = \overline{0, n}, \; h = \frac{x_{j+1} - x_j}{n}. \tag{2.25}$$

В каждом узле (2.25) вычисляются значения $f(x_{jp}, \bar{y}_{jp})$, где $\bar{y}_{jp}$ определяется по методу Эйлера:

$$\bar{y}_{jp} = \bar{y}_{j(p-1)} + h \cdot f(x_{j(p-1)}, \bar{y}_{j(p-1)}), \quad p = \overline{1, n}, \tag{2.26}$$

при этом в качестве начального значения $\bar{y}_{j0}$ берется значение на правой границе из окончательного приближения на предыдущем подынтервале: $\bar{y}_{j0} = \bar{y}_{(j-1)n}$, для начального подынтервала из (2.21) $\bar{y}_{00} = y_0$. Ниже, в п. 2.4, исследуются варианты метода с вычислением узловых значений на основе явных методов Рунге-Кутты высших порядков. Значения $f(x_{jp}, \bar{y}_{jp})$ принимаются в качестве значений в узлах интерполяции:

$$\bar{f}_{jp} = f(x_{jp}, \bar{y}_{jp}), \quad p = \overline{0, n}. \tag{2.27}$$

Аналогично, всюду ниже $\bar{y}$ будет обозначать вычисляемое приближение точного решения y, $\bar{f}$ – приближение функции правой части (2.21). Согласно условиям интерполяции (2.27) строится интерполяционный полином Ньютона степени n, который по схеме (2.4) – (2.11) приводится к виду:

$$\psi_{jn}(x) = a_{j0} + \sum_{\ell=1}^{n} a_{j\ell} \left(\frac{x - x_{j0}}{h} \right)^{\ell}, \tag{2.28}$$

где $a_{j0} = \bar{f}_{j0}$, $a_{j\ell} = \sum_{k=\ell}^{n} \frac{b_{jk}\, d_{k\ell}}{k!}$, $b_{jk} = \Delta^k \bar{f}_{j0}$.

Полином (2.28) приближает производную решения задачи (2.21). Приближение самого решения строится как первообразная от (2.28) с постоянной, принимающей значение $\bar{y}_{j0}$. Семейство первообразных от полинома $\psi_{jn}(x)$ на j-м подынтервале имеет вид (2.20). Фиксирование в правой части (2.20) значения нижнего предела и замена константы C на $\bar{y}_{j0}$ определяет функцию $P_j(x) = \bar{y}_{j0} + \int_{x_{j0}}^{x} \psi_{jn}(x)\, dx$ или

$$P_j(x) = \bar{y}_{j0} + h \sum_{\ell=0}^{n} \frac{a_{j\ell}}{\ell+1} \left(\frac{x - x_{j0}}{h} \right)^{\ell+1}. \tag{2.29}$$

Полином (2.29) принимается в качестве приближения решения $y(x)$ на j-м подынтервале:

$$y(x) \approx P_j(x), \quad x \in [x_j, x_{j+1}]. \tag{2.30}$$

Вычисление значений полинома (2.29) производится по схеме Горнера при $t = \frac{x - x_{j0}}{h}$:

$$P_j(t) = \bar{y}_{j0} + h \left(\ldots \left(\left(\frac{a_{jn}}{n+1}\, t + \frac{a_{j(n-1)}}{n} \right) t + \frac{a_{j(n-2)}}{n-1} \right) t + \ldots + a_{j0} \right) t. \tag{2.31}$$

Аналогичное приближение строится на следующем подынтервале и т. д. до исчерпания интервала $[a_i, b_i]$.

Замечание 2.2. В силу интерполяции приближение (2.29) является аналитическим внутри каждого подынтервала. Поскольку согласно построению

$P_j(x_{j0}) = P_{j-1}(x_{(j-1)n})$, оно непрерывно на границах каждого подынтервала. Поэтому при каждом i из (2.22) данное приближение непрерывно на $[a_i, b_i]$. С учетом того, что $\bar{y}(a_i) = \bar{y}(b_{i-1})$, оно непрерывно на всем отрезке $[a, b]$. Приближение производной решения также непрерывно на всем $[a, b]$. Действительно, поскольку $x_{j0} = x_{(j-1)n}$, для решения выполнено равенство $\bar{y}_{j0} = \bar{y}_{(j-1)n}$; следовательно, $f(x_{j0}, \bar{y}_{j0}) = f(x_{(j-1)n}, \bar{y}_{(j-1)n})$. Из (2.27), (2.28) имеем $\psi_{jn}(x_{j0}) = \psi_{(j-1)n}(x_{(j-1)n})$; таким образом, рассматриваемое приближение непрерывно при каждом i из (2.22) на $[a_i, b_i]$. Равенство $\bar{y}(a_i) = \bar{y}(b_{i-1})$ обуславливает $f(a_i, \bar{y}(a_i)) = f(b_{i-1}, \bar{y}(b_{i-1}))$; следовательно, приближение правой части (2.21) непрерывно на всем $[a, b]$.

Значения $\bar{y}_{jp} = P_j(x_{jp})$, $p = \overline{1, n}$, из (2.29) в процессе компьютерной реализации оказываются более точными приближениями решения, чем получаемые непосредственно разностным методом. Эти значения целесообразно принять как новые уточненные значения в интерполяционных узлах для последующего интерполирования. Уточнение строится посредством следующего рекуррентного процесса. За результаты первой итерации на j-м подынтервале принимаются $\bar{y}_{jp} = P_j(x_{jp})$ из (2.31): $\bar{y}_{jp}^{(1)} = \bar{y}_{jp}$, $p = \overline{1, n}$. В этом обозначении они используются в качестве подстановки второго аргумента функции $f(x_{jp}, \bar{y}_{jp}^{(1)})$, $p = \overline{1, n}$. Далее описанный процесс интерполирования повторяется один раз. В свою очередь, значения, полученные таким образом в результате второй итерации, обозначаются $\bar{y}_{jp}^{(2)}$, $p = \overline{1, n}$. Они повторно используются для новых уточнений на текущем подынтервале с узловыми значениями $f(x_{jp}, \bar{y}_{jp}^{(2)})$, $p = \overline{1, n}$, и описанный выше процесс повторяется. В результате ℓ повторений приближение производной $f(x_{jp}, \bar{y}_{jp}^{(\ell)})$, $p = \overline{1, n}$, оказывается существенно более точным, чем начальное, соответственно

уточняется приближение решения. Для перехода от ℓ-й к $(\ell+1)$-й итерации индексируются значения $\psi_{jn}^{(\ell)}(x_{jp}) = f(x_{jp}, \bar{y}_{jp}^{(\ell)})$ из (2.28) для всех $p = \overline{1, n}$ из (2.25) и значения $\bar{y}_{jp}^{(\ell+1)} = P_j^{(\ell)}(x_{jp})$, вычисляемые из вновь построенного полинома с коэффициентами вида (2.29), которые используются в качестве $\bar{y}_{jp}$ для (2.27) при выполнении $(\ell+1)$-й итерации. После завершения ℓ итераций на j-м подынтервале выполняется переход к $(j+1)$-му подынтервалу, где в качестве начального значения приближения решения берется значение окончательного приближения на конце предыдущего подынтервала: $\bar{y}_{(j+1)0} = \bar{y}_{jn}^{(\ell)}$. На $(j+1)$-м подынтервале снова выполняются ℓ итераций, процесс воспроизводится от $j = 0$ до $j = P-1$ из (2.23). Число итераций ограничивается: $\ell \le \widetilde{L}$, $\widetilde{L} = \text{const}$, на практике $\widetilde{L} \le 10$. Все описанные итерации выполняются на каждом интервале $[a_i, b_i]$.

Листинг программы *DiffPol*, реализующей разностно-полиномиальный метод решения задачи (2.21), с подробными комментариями приводится непосредственно ниже.

Листинг 2.2

```
program DiffPol; {$APPTYPE CONSOLE} uses SysUtils;
const Anach=0; Bkonech=10; //границы отрезка интегрирования
      nn=10;     //граница степени интерполяционного полинома
type  matr=array[0..nn,0..nn] of extended; matr_C=array[-3..nn+1] of extended;
      matric=array[0..20200] of matr_C; vect=array[0..nn] of extended;
var mm,p7,r,pod:longint; m,i,j,l,n,Nmin,Kmin,k_,iter:byte;
  pp,p,ss,t,xpr,tpr: extended;  bb,X,Y,A,fy: vect; dy,d: matr; Ck:matric;
  a00,b00,h,h0,h3,x3,y3,lk,a0,b0,max,min,Velpod,ypr:extended;
Описание процедуры Viet
function f(x,y:extended):extended; //функция правой части
begin f:=cos(x+y); end;
function fun(x:extended):extended; //аналитическое решение, используемое
begin fun:=-x+2*arctan(x); end;       //для вывода погрешности приближения
//вычисление значений конечных разностей для аппроксимации функции правой части
procedure Konech_Raznost(var dy:matr); var i,j:byte;
begin for j:=0 to n-1 do dy[1,j]:=fy[j+1]-fy[j];
 for i:=2 to n do for j:=0 to n-i do dy[i,j]:=dy[i-1,j+1]-dy[i-1,j]; end;
//вычисление полинома по схеме Горнера на фиксированном подынтервале
function fapr(Koef:matr_C; xu:extended):extended;
var   ll: Shortint;  Si,tpr: extended;
begin tpr:=(xu-Koef[-1])/h; Si:=Koef[n+1];
 for ll:=n downto 0 do Si:=tpr*Si+Koef[ll];
fapr:=Si end;
begin
```

```
Viet; Velpod:=1; {величина интервала} iter:=10; {количество итераций}
a0:=Anach; b0:=Anach+Velpod; lk:=0; {начальноеусловие}
//непосредственное решение задачи по интервалам
while a0 <Bkonech do begin
//программный выбор числа подынтервалов и степени полинома для данного интервала
Min:=10; k_:=6; pod:=0;
repeat n:=4; //фиксирование начального значения степени полинома
repeat max:=0;
h0:=(b0-a0)/exp(k_*ln(2)); // длина подынтервала
a00:=a0; b00:=a00+h0; y[0]:=lk;
 x[0]:=a0; mm:=0; pod:=0;
   while a00<=b0-h0 do begin
    h:=(b00-a00)/n;
    for j:=1 to n do begin mm:=mm+1; x[j]:=a0+mm*h; end;
    for i:=1 to n do y[i]:=y[i-1]+h*f(x[i-1],y[i-1]);
    for m:=1 to iter do begin  for i:=0 to n do fy[i]:=f(x[i],y[i]);
    Konech_Raznost(dy); p7:=1;
    for j:=1 to n do begin p7:=p7*j; bb[j]:=dy[j,0]/p7; end;
    a[0]:=fy[0];
for l:=1 to n do begin
     ss:=0; for j:=l to n do ss:=ss+d[l,j]*bb[j]; a[l]:=ss; end;
     for i:=1 to n do begin
      t:=(x[i]-x[0])/h; pp:=a[n]/(n+1);
      for r:=n-1 downto 0 do pp:=pp*t+a[r]/(r+1); y[i]:=pp*h*t+y[0];
     end; end;
    h3:=h/7; x3:=a00;
    while x3<=b00 do begin
     t:=(x3-x[0])/h; pp:=a[n]/(n+1);
     for r:=n-1 downto 0 do pp:=pp*t+a[r]/(r+1);
     y3:=pp*h*t+y[0]; {значение решения(интеграл)} p:=a[n];
for r:=n-1 downto 0 do p:=p*t+a[r]; //значение производной
     if abs(p-f(x3,y3))>max then max:=abs(p-f(x3,y3)); x3:=x3+h3; end;
    Y[0]:=Y[n]; x[0]:=x[n]; pod:=pod+1; a00:=a00+h0; b00:=a00+h0;
   end; //переход к следующим n значениям
if max<Min then begin Min:=max; Nmin:=n; Kmin:=k_; end;
n:=n+1;
  until n>8;
k_:=k_+1;
 untilk_>9;
//приближение решения с программно определенными значениями N и K
n:=Nmin; k_:=Kmin; h0:=(b0-a0)/exp(k_*ln(2)); a00:=a0; b00:=a00+h0;
 y[0]:=lk; x[0]:=a0; mm:=0; pod:=0;
 while a00<=b0-h0 do begin h:=(b00-a00)/n;
  for j:=1 to n do begin  mm:=mm+1; x[j]:=a0+mm*h; end;
  for i:=1 to n do y[i]:=y[i-1]+h*f(x[i-1],y[i-1]);
  for m:=1 to iter do begin
   for i:=0 to n do fy[i]:=f(x[i],y[i]);
 Konech_Raznost(dy);p7:=1;
 for j:=1 to n do begin p7:=p7*j; bb[j]:=dy[j,0]/p7; end;
   a[0]:=fy[0];
   for l:=1 to n do begin ss:=0;
    for j:=l to n do ss:=ss+d[l,j]*bb[j]; a[l]:=ss; end;
   for i:=1 to n do begin
t:=(x[i]-x[0])/h; pp:=a[n]/(n+1);
   for r:=n-1 downto 0 do pp:=pp*t+a[r]/(r+1);
 y[i]:=pp*h*t+y[0]; end; end;
//Сохранение коэффициентов интерполяционного полинома на текущем подынтервале
Ck[pod,0]:=y[0]; Ck[pod,-1]:=x[0]; Ck[pod,-2]:=n; Ck[pod,-3]:=h;
  for i:=1 to n+1 do Ck[pod,i]:=a[i-1]*h/i;
  Y[0]:=Y[n]; x[0]:=x[n]; pod:=pod+1; a00:=a00+h0; b00:=a00+h0; end;
 xpr:=a0*1.03; pod:=trunc((xpr-a0)/(h*n)); ypr:=fapr(Ck[pod],xpr);
 writeln(xpr,'  ',ypr,'  ',abs(ypr-fun(xpr))); //Вывод приближения
```

```
 pod:=trunc((b0-a0)/(h*n))-1; lk:=fapr(Ck[pod],b0);
 a0:=a0+Velpod; b0:=b0+Velpod;
end; {переход к следующему интервалу} readln; end.
```

Описание процедуры *Viet* приведено в листинге 2.1, процедура вычисляет и сохраняет в качестве значений элементов массива коэффициенты $d_{j\ell}$, $\ell = \overline{0,\ j}$, которые далее в программе используются для построения интерполяционного полинома в виде (2.28). Вычисление конечных разностей $b_{jk} = \Delta^k \bar{f}_{j0}$, $k = \overline{\ell,\ n}$, $\ell = \overline{1,\ n}$ из (2.28) производится с помощью процедуры *Konech_Raznost*. Функция *fapr* производит вычисление значений полинома Ньютона на соответствующем подынтервале при известных коэффициентах по схеме (2.31).

Представленная в листинге 2.2 программа дает непрерывное приближение решения задачи Коши $y' = \cos(x+y)$, $y(0) = 0$, на интервале $x \in [0, 10]$ описанным варьируемым разностно-полиномиальным методом. Результат работы программы (в 1-й колонке – аргумент, во 2-й – значение приближенного решения, в 3-й – абсолютная погрешность решения):

```
0.00000000000000E+0000   0.00000000000000E+0000   0.00000000000000E+0000
1.03000000000000E+0000   5.70350825609881E-0001   2.71050543121376E-0019
2.06000000000000E+0000   1.77733862792969E-0001   1.21972744404619E-0019
3.09000000000000E+0000  -5.74382126830208E-0001   1.62630325872826E-0019
4.12000000000000E+0000  -1.45463491123023E+0000   4.33680868994202E-0019
5.15000000000000E+0000  -2.39198363038362E+0000   0.00000000000000E+0000
6.18000000000000E+0000  -3.35925099013609E+0000   0.00000000000000E+0000
7.21000000000000E+0000  -4.34404140763963E+0000   4.33680868994202E-0019
8.24000000000000E+0000  -5.33994462268117E+0000   8.67361737988404E-0019
9.27000000000000E+0000  -6.34332597969473E+0000   0.00000000000000E+0000
```

Для вывода погрешности приближения использовано аналитическое решение задачи $y = -x + 2\,arctg(x)$. В табл. 2.1 представлено сравнение полученных значений погрешности приближения с абсолютными погрешностями решения данной задачи явными методами Рунге-Кутты различных порядков.

Таблица 2.1

Абсолютная погрешность приближения решения задачи $y' = \cos(x + y),\ y(0) = 0$

x	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*DiffPol*
	$h = 1.03 \times 10^{-3}$	$h = 1.03 \times 10^{-3}$	$h = 1.03 \times 10^{-2}$	
1.03	1.188**e-14**	5.963**e-19**	7.589**e-19**	2.711**e-19**
2.06	6.487**e-15**	1.166**e-18**	1.220**e-19**	1.220**e-19**
…	…	…	…	…
8.24	5.330**e-16**	1.735**e-17**	9.107**e-18**	8.674**e-19**
9.27	4.276**e-16**	8.764**e-19**	5.638**e-18**	0.000**e+00**

Названия столбцов таблицы соответствуют названиям методов, применяемых для приближения решения: *Runge-Kutta_4* – метод Рунге-Кутты 4-го порядка [72], *Butcher_6* – метод Бутчера 6-го порядка [35], *Dormand-Prince_8* – метод Дормана-Принса 8-го порядка аппроксимации [35], *DiffPol* – метод варьируемого разностно-полиномиального приближения (листинг 2.2). Шаг наилучшего компьютерного приближения разностным методом обозначен h. Методу 4-го порядка соответствует погрешность 10^{-16}, затем с повышением порядка разностного метода она убывает, достигая 10^{-17} для метода 8-го порядка. Краткий обзор разностных методов представлен в п. 1.2, их программные реализации на языке *Delphi* даны в приложении к главе 2 (п. 2). Граница погрешности варьируемого разностно-полиномиального решения не превышает порядка 10^{-19} на всем интервале приближения. Значения параметров метода: $\nabla = 1$ – длина интервала из (2.22), $\widetilde{L} = 6$ – число итераций, значения степени полинома n и числа подынтервалов $P = 2^k$ из (2.23) варьировались программой в следующих границах: $4 \le n \le 8,\ 6 \le k \le 9$. Время работы всех программ менее одной секунды.

Аналогичное соотношение погрешностей приближения наблюдается в экспериментах с другими нежесткими задачами [129, 131]. Как правило, погрешность разностно-полиномиального приближения порядка 10^{-18} не превышается на всем отрезке (2.22) (часть эксперимента приводится в п. 2.5 и в приложении к главе 2).

Непосредственно ниже доказывается равномерная сходимость метода, приводятся оценки скорости сходимости.

2.3. Сходимость и скорость сходимости разностно-полиномиального решения дифференциальной системы. Первоначально оценка выполняется без учета итерационного уточнения, при этом для некоторого $d = \text{const}$ в области $\widetilde{R} = \{\, a \le x \le b\,, |y - y_0| \le d\,\}$ предполагаются выполненными все условия существования и единственности решения задачи (2.21); в частности, функция $f(x, y)$ существует и непрерывна во всей $\widetilde{R}$; кроме того, выполнено условие Липшица $\forall x, y, \widetilde{y} : (x, y) \in \widetilde{R} \wedge (x, \widetilde{y}) \in \widetilde{R}$:

$$| f(x, y) - f(x, \widetilde{y}) | \le L | y - \widetilde{y} |, L = \text{const}. \qquad (2.32)$$

Первоначально предполагается, что $f(x, y(x))$ имеет $n+1$ непрерывную производную на всем отрезке $[a, b]$ из (2.22), на границах отрезка производные понимаются как соответственно односторонние. Пусть для произвольного i зафиксирован интервал $[a_i\,, b_i]$. На этом и всех других интервалах из (2.22) зафиксирована степень n интерполирующих правую часть (2.21) полиномов $\psi_{jn}(x)$ из (2.28), $n = \text{const}$, $1 \le n$; $j = \overline{0, P-1}$. Согласно (2.25) – (2.28) для вычисления значений в узлах интерполяции применяется метод Эйлера в предположении, что разностные значения, а также кусочно-полиномиальные приближения решения не выводят из области $\widetilde{R}$, для них сохраняется условие (2.32), предполагается существование и непрерывность $n+1$ производной функции $f(x, \bar{y}(x))$, где $\bar{y}(x)$ – рассматриваемое приближение решения $y(x)$.

Начальный этап разбиения выбранного интервала на подынтервалы считается нулевым, последующие этапы нумеруются соответственно показателю степени k из (2.23).

На нулевом этапе полагается $[x_0, x_1] = [a_i, b_i]$. Аналогично (2.12), (2.13), погрешность интерполирования правой части (2.21) на данном подынтервале полиномом $\psi_{0n}(x)$ с узловыми значениями $f(x_{0p}, \bar{y}_{0p})$ из (2.27) при $j = 0$ можно оценить из неравенства:

$$| f(x, \bar{y}) - \psi_{0n}(x) | \le \widetilde{c}\, h^{n+1},\ \widetilde{c} = \widetilde{c}(f, n, \widetilde{R}),\ P = 1,\ [a_i, b_i] = [x_0, x_1], \quad (2.33)$$

где $\widetilde{c}(f, n, \widetilde{R}) = \max\limits_{\widetilde{R}} | f^{(n+1)}(x, \bar{y}(x)) |$, $\widetilde{c} = \text{const}$, $\widetilde{c} < \infty$ в силу замкнутости $\widetilde{R}$.

Аналогично (2.14), (2.15) применительно к функции $f(x, \bar{y}(x))$ на k-м этапе, $1 \le k$, интервал $[a_i, b_i]$ разбивается на $P = 2^k$ подынтервалов равной длины, на каждом из них интерполируется правая часть (2.21) полиномом Ньютона $\psi_{jn}(x)$, $j = \overline{0, P-1}$. При этом входные значения в узлах интерполяции $f(x_{jp}, \bar{y}_{jp})$ на каждом подынтервале с номером $j = \overline{0, P-1}$ данного этапа разбиения вычисляются согласно (2.26), (2.27). Начальным значением $\bar{y}_{j0}$ при данном и всех последующих k берется значение на правой границе из окончательного приближения на предыдущем подынтервале: $\bar{y}_{j0} = \bar{y}_{(j-1)n}$. При $j = 0$ значение $\bar{y}_{00}$ полагается равным значению приближения в правой границе предыдущего интервала с номером $i-1$, для начального интервала ($i = 0$) – $\bar{y}_{00} = y_0$ из (2.21). Шаг метода Эйлера при вычислении узловых значений на каждом подынтервале полагается равным расстоянию между узлами интерполяции, которое составляет $h / 2^k$. В результате по аналогии с (2.13) – (2.15) из (2.33) вытекает оценка

$$| f(x, \bar{y}) - \psi_{k j_k n}(x) | \le \frac{\widetilde{c}}{2^{k(n+1)}} h^{n+1},\ P = 2^k,\ [a_i, b_i] = \bigcup_{j=0}^{2^k-1} [x_j, x_{j+1}], \quad (2.34)$$

где $\psi_{k j_k n}(x) = \psi_{j n}(x)$ – интерполяционный полином Ньютона степени n из (2.28) для j_k-го подынтервала; шаг h аналогично (2.14), (2.15) относится к интерполяционному полиному на начальном интервале $[a_i, b_i]$ и не изменяется при его разбиении на $P = 2^k$ частей.

Далее, пусть $\psi_{k j_k n} = \psi_{k j_k n}(x)$. Очевидно,

$$|\psi_{k j_k n} - \psi_{(k-1) j_{k-1} n}| \le |f(x, \bar{y}) - \psi_{k j_k n}| + |f(x, \bar{y}) - \psi_{(k-1) j_{k-1} n}|, \qquad (2.35)$$

где $x \in [x_{j_k}, x_{j_k+1}]$. На $(k-1)$-м этапе – при четном j_k имеем $x \in [x_{j_{k-1}}, (x_{j_{k-1}+1} - x_{j_{k-1}}) / 2]$, при нечетном j_k имеем $x \in [(x_{j_{k-1}+1} - x_{j_{k-1}}) / 2,\ x_{j_{k-1}+1}]$. Из (2.34), (2.35) следует $|\psi_{k j_k n} - \psi_{(k-1) j_{k-1} n}| \le \tilde{c}\, h^{n+1}\left(\frac{1}{2^{k(n+1)}} + \frac{1}{2^{(k-1)(n+1)}}\right)$ или

$$|\psi_{k j_k n} - \psi_{(k-1) j_{k-1} n}| \le \frac{\tilde{C}\, h^{n+1}}{2^{k(n+1)}},\ \tilde{C} = \tilde{c}(1 + 2^{n+1}). \qquad (2.36)$$

Неравенство $|\psi_{(k+m) j_{k+m} n} - \psi_{k j_k n}| \le \sigma_{j_{k+1}(k+1)} + \sigma_{j_{k+2}(k+2)} + \ldots + \sigma_{j_{k+m}(k+m)}$, где $\sigma_{j_{k+\ell}(k+\ell)} = |\psi_{(k+\ell) j_{k+\ell} n} - \psi_{(k+\ell-1) j_{k+\ell-1} n}|$, $\ell = \overline{1, m}$, справедливо для произвольного $m \in N$. Отсюда и из (2.36) следует $|\psi_{(k+m) j_{k+m} n} - \psi_{k j_k n}| \le \tilde{C} h^{n+1} \sum_{r=1}^{m} \frac{1}{2^{(k+r)(n+1)}}$ или

$$|\psi_{(k+m) j_{k+m} n} - \psi_{k j_k n}| \le \tilde{C} h^{n+1} \frac{1}{2^{(k+1)(n+1)}} \sum_{r=0}^{m-1} \frac{1}{2^{r(n+1)}}. \qquad (2.37)$$

При замене конечного ряда слагаемых в (2.37) на $\sum_{r=0}^{\infty} \frac{1}{2^{r(n+1)}}$ получаем

$$|\psi_{(k+m) j_{k+m} n} - \psi_{k j_k n}| < \frac{\tilde{C} h^{n+1}}{2^{(k+1)(n+1)}} \frac{1}{1-q}, \qquad (2.38)$$

где $q = \frac{1}{2^{n+1}}$. Из (2.38) следует, что для любого $\varepsilon > 0$ существует $N = N(\varepsilon)$ такое, что

$$|\psi_{(k+m)j_{k+m}n} - \psi_{kj_kn}| \le \varepsilon \ \forall x \in [a_i, b_i],$$

если $k \ge N$, где $N = \text{int}\left((n+1)\log_2\left(\frac{\widetilde{C}\,h^{n+1}}{\varepsilon(1-q)}\right)\right) - 1$. Оценка (2.38) не зависит от выбора m, поэтому на основании леммы Больцано-Коши справедлива следующая лемма.

Лемма 2.2. Последовательность $\psi_{kj_kn} = \psi_{kj_kn}(x)$ равномерно сходится к некоторой функции $\widetilde{f}(x,\ \bar{y})$ на$[a_i, b_i]$:

$$\psi_{kj_kn} \rightrightarrows \widetilde{f}(x,\ \bar{y}),\ k \to \infty. \tag{2.39}$$

Переход к пределу в неравенстве (2.38) при $m \to \infty$ обуславливает следствие.

Следствие 2.2. Скорость сходимости (2.39) оценивается из неравенства

$$|\widetilde{f}(x,\ \bar{y}) - \psi_{kj_kn}| \le \frac{\widetilde{C}\,h^{n+1}}{2^{(k+1)(n+1)}}\,\frac{1}{1-q}. \tag{2.40}$$

Функция $\widetilde{f}(x, \bar{y})$ из (2.39), (2.40) совпадает с $f(x, y)$ из (2.21) вследствие того, что последовательность ψ_{kj_kn} включает сходящуюся к $f(x, y)$ подпоследовательность. Последняя определяется последовательностью разностных приближений решения по методу Эйлера, при этом учитывается, что вариации начальных значений на подынтервале вносят в этот метод погрешность на шаге высшего порядка малости.

Действительно, для интерполяционного полинома ψ_{kj_kn} при каждом k в качестве узловых значений принимаются разностные значения, вычисленные

по методу Эйлера с шагом $h / 2^k$. Для нулевого подынтервала на интервале с номером $i=0$ начальное данное для метода Эйлера заимствуется из (2.21). На всех последующих подынтервалах начальное значение для этого метода выбирается из интерполяционного приближения на предшествующем подынтервале с порядком погрешности меньшим порядка погрешности для метода Эйлерана шаге. На j_k-м подынтервале погрешность метода Эйлера на шаге можно оценить [117] из формулы Тейлора:

$$y_{p+1} = y_p + f(x_p, y_p)\frac{h}{2^k} + f'(\zeta_p, y(\zeta_p))\left(\frac{h}{2^k}\right)^2,\ x_p < \zeta_p < x_{p+1},$$

откуда следует неравенство $|\bar{y}_{p+1} - y_{p+1}| \le c_0\left(\frac{h}{2^k}\right)^2$, где $c_0 = \max\limits_{\tilde{R}} |f'(x, y)|$, $\bar{y}_{p+1}$ – приближение по методу Эйлера. Погрешность интерполирования функции $f(x, \bar{y})$ оценивается из (2.34): $|f(x, \bar{y}) - \psi_{k j_k n}(x)| \le \frac{\tilde{c}}{2^{k(n+1)}} h^{n+1}$. Пусть $\varepsilon_k(x) = f(x, \bar{y}) - \psi_{k j_k n}(x)$, тогда $\bar{y}_{p+1} = \bar{y}_p + (\varepsilon_k(x_p) + \psi_{k j_k n}(x_p))\frac{h}{2^k}$, погрешность метода Эйлера на одном шаге от собственно интерполирования на предыдущем подынтервале составит $\varepsilon_k(x_p)\frac{h}{2^k} = \frac{\tilde{c}\, h^{n+2}}{2^{k(n+2)}}$, а это при всех $n \ge 1$ на порядок ниже погрешности на шаге непосредственно метода Эйлера. Отсюда метод Эйлера с использованием кусочного интерполирования сходится к той же функции, что и этот же метод с шагом $\frac{h}{2^k}$. Более точно: $\bar{y}_{j_k p} \to y(x)$ на $[a_i, x]$ в силу $\frac{h}{2^k} \to 0$ при каждом $x \in [a_i, b_i]$, а подпоследовательность узловых значений $f(x_{j_k p}, \bar{y}_{j_k p})$, где p – номер узла интерполяции, сходится при тех же x к правой части (2.21) при $k \to \infty$. Отсюда $f(x, y) = \tilde{f}(x, \bar{y})\ \forall x \in [a_i, b_i]$. Таким образом, имеет место следствие.

Следствие 2.3. В условиях леммы 2.2 $\psi_{k j_k n} \rightrightarrows f(x,\ y)\ \ \forall x \in [a_i, b_i]$ и на k-м этапе

$$| f(x,\ y) - \psi_{k j_k n} | \le \frac{\widetilde{C} h^{n+1}}{2^{(k+1)(n+1)}} \frac{1}{1-q}. \tag{2.41}$$

Остается оценить погрешность приближения непосредственно самого решения. На k-м этапе разбиения согласно (2.29), (2.30) для подынтервала с номером $j = j_k$ погрешность приближения решения определяется из равенства

$$| P_{j_k k}(x) - y(x) | = \left| \bar{y}_{j_k 0} + \int_{x_{j_k 0}}^{x} \psi_{k j_k n}\, dx - y(x_{j_k 0}) - \int_{x_{j_k 0}}^{x} f(x, y)\, dx \right|,\ x \in [x_{j_k}, x_{j_k+1}].$$

Отсюда

$$| P_{j_k k}(x) - y(x) | \le \max_{x_{j_k} \le x \le x_{j_k+1}} | x - x_{j_k 0} |\, | \psi_{k j_k n} - f(x, y) | + | \bar{y}_{j_k 0} - y(x_{j_k 0}) |.$$

С учетом (2.41) и того, что $| x - x_{j_k 0} | \le \frac{\nabla}{2^k}$, где ∇ следует из (2.22), последняя оценка обуславливает неравенство

$$| P_{j_k k}(x) - y(x) | \le \frac{\nabla}{2^k} \frac{\widetilde{C} h^{n+1}}{2^{(k+1)(n+1)}} \frac{1}{1-q} + | \bar{y}_{j_k 0} - y(x_{j_k 0}) |, \tag{2.42}$$

где $\widetilde{C}$ следует из (2.36). Значение $\bar{y}_{j_k 0}$ из (2.42) равно значению полиномиального приближения на конце подынтервала с номером $j_k - 1$, поэтому $\bar{y}_{j_k 0} = P_{(j_k-1) k}(x_{(j_k-1) n})$. Отсюда

$$\max_{x_{j_k} \le x \le x_{j_k+1}} | P_{j_k k}(x) - y(x) | \le \frac{\nabla}{2^k} \frac{\widetilde{C} h^{n+1}}{2^{(k+1)(n+1)}} \frac{1}{1-q} + \max_{x_{j_k} \le x \le x_{j_k+1}} | P_{(j_k-1) k}(x) - y(x) |. \tag{2.43}$$

В обозначении

$$\varepsilon_{j_k} = \max_{x_{j_k} \le x \le x_{j_k+1}} |P_{j_k k}(x) - y(x)|,\ \tau_k = \frac{\nabla}{2^k} \frac{\widetilde{C} h^{n+1}}{2^{(k+1)(n+1)}} \frac{1}{1-q}$$

неравенство (2.43) примет вид: $\varepsilon_{j_k} \le \tau_k + \varepsilon_{j_k-1}$. По рекуррентности $\varepsilon_{j_k} \le \tau_k + \tau_k + \ldots + \tau_k + \varepsilon_0$, где количество слагаемых τ_k равно числу подынтервалов на отрезке $[a_i, b_i]$, умноженному на число интервалов $i+1$, ε_0 – ошибка ввода начального данного, предполагаемая равной нулю. Таким образом, $\varepsilon_{j_k} \le 2^k (i+1)\, \tau_k$. Поскольку $i+1 = \frac{b_i - a}{b_i - a_i}$ и $b_i \le b$, то $\varepsilon_{j_k} \le 2^k (b-a) \nabla\, \tau_k \ \ \forall x \in [a, b_i]$. В силу произвольности выбора i оценка верна $\forall x \in [a, b]$. Таким образом,

$$|P_{j_k k}(x) - y(x)| \le \frac{\widetilde{C}(b-a)\, h^{n+1}}{2^{(k+1)(n+1)}} \frac{1}{1-q} \ \ \forall x \in [a, b]. \qquad (2.44)$$

Правая часть (2.44) не зависит от x; следовательно, выполняется равномерная сходимость $P_{j_k k}(x) \rightrightarrows y(x) \ \ \forall x \in [a, b]$ при $k \to \infty$. Остается учесть, что $P_{j_k k}(x) = P_j(x)$ из (2.29) при $j = j_k$ на k-м этапе разбиения.

Из изложенного вытекает теорема.

Теорема 2.1. При любом выборе n, $1 \le n \le n_0$, $n_0 = \text{const}$, последовательность полиномов $P_{j_k k}(x)$, где k определяется из (2.23), равномерно на $[a, b]$ сходится к решению $y(x)$ задачи (2.21) при $k \to \infty$. Скорость сходимости оценивается из (2.44), где $\widetilde{C} = \text{const}$, h – шаг интерполирования полинома $P_{j_0 0}(x)$ на $[a_i, b_i]$ из (2.23), выбранный в соответствии $k = 0$, его значение не изменяется с ростом k.

Следствие 2.4. В условиях теоремы 2.1 для $n=1$ выполняется равномерная сходимость $P_{j_k k}(x) \rightrightarrows y(x) \ \ \forall x \in [a, b]$, если $k \to \infty$, при этом

$$| P_{j_k k}(x) - y(x) | \le \frac{\widetilde{c}_0 (b-a)\, h^2}{2^{2(k+1)}}, \ \ \widetilde{c}_0 = \text{const}, \tag{2.45}$$

где $\widetilde{c}_0 = \dfrac{\widetilde{C}}{1-q}$, $\widetilde{C}$ определяется из (2.36), q – из (2.38) при $n=1$.

Следствие 2.5. В рассматриваемых условиях теоремы 2.1 и при $h < 1$ на k-м этапе разбиения одновременно всех интервалов (2.22) для $\forall x \in [a, b]$ выполняется неравенство

$$| P_{j_k k}(x) - y(x) | \le \frac{\widetilde{C}_0 (b-a)}{2^{2(k+1)}}, \widetilde{C}_0 = \text{const}, \tag{2.46}$$

где константа $\widetilde{C}_0$ не зависит от n.

Доказательство следует из справедливости (2.44) для любого n, $1 \le n \le n_0$, при $q = \dfrac{1}{2^{n+1}}$, $\widetilde{C} = \widetilde{c}(f, n, \widetilde{R})\,(1+2^{n+1})$. Замена $\widetilde{c}(f, n, \widetilde{R})$ на $\max\limits_{1 \le n \le n_0} \widetilde{c}(f, n, \widetilde{R})$, множителя $1+2^{n+1}$ на $1+2^{n_0+1}$, дроби $q = \dfrac{1}{2^{n+1}}$ на $q = 1/4$, значения h на единицу приводит к (2.46).

Замечание 2.3. Из следствий 2.4, 2.5 вытекает независимость оценок (2.45), (2.46) от порядка гладкости $n+1$ правой части (2.21). Начиная с двукратной непрерывной дифференцируемости функции $f(x, y)$, порядок гладкости не является критичным для сходимости кусочно-полиномиального приближения. Малость правых частей (2.45), (2.46) достигается сужением длины подынтервалов в геометрической прогрессии. Кроме того, рост порядка гладкости снижает величины констант в мажорирующих оценках (2.41), (2.44). Эти утверждения согласуются срезультатами численных экспериментов,

которые приводятся в [127, 131, 132], часть результатов приводится ниже, в п. 2.5.

Замечание 2.4. Оценка (2.46) является мажорантной для n и относится, в частности, к выбору варьируемых n для минимизации погрешности (2.24) в границах $1 \le n \le n_0$.

Для применения метода Эйлера в рассматриваемом кусочно-полиномиальном приближении достаточно однократной дифференцируемости правой части (2.21) и выполнения условия Липшица (2.32). Отсюда с учетом замечаний 2.3, 2.4 метод в целом применим при невысоком порядке гладкости решения задачи (2.21), тогда как методы высоких порядков для своей сходимости требуют существования высших производных [12].

2.4. Обработка данных в дифференциальных моделях ИВС на основе высоких порядков приближений узловых значений. В качестве исходного разностного метода можно использовать метод высокого порядка, если это допускает ограничение на правую часть (2.21). Ниже в процессе описываемого исследования рассматриваются следующие методы. Метод Эйлера-Коши на подынтервале $[x_j, x_{j+1}]$ применяется в виде эквивалентном классической записи [117]

$$\widetilde{y}_{jp} = y_{j(p-1)} + h \cdot f(x_{j(p-1)}, y_{j(p-1)}),$$
$$y_{jp} = y_{j(p-1)} + \frac{h}{2}\left(f(x_{j(p-1)}, y_{j(p-1)}) + f(x_{jp}, \widetilde{y}_{jp}) \right), \; p = \overline{1, n},$$

где x_{jp}, h определяются из (2.25), n – степень интерполяционного полинома из (2.28), y_{jp} принимаются в качестве интерполируемых значений аналогично значениям (2.26) для полинома (2.28). По аналогичной схеме применяются и другие разностные методы. Метод Бутчера 6-го порядка записывается в виде [35]:

$$k_{1p} = h f(x_{j(p-1)} + c_1 h,\ y_{j(p-1)}),\ k_{2p} = h f(x_{j(p-1)} + c_2 h,\ y_{j(p-1)} + a_{21} k_{1p}),$$
$$k_{3p} = h f(x_{j(p-1)} + c_3 h,\ y_{j(p-1)} + a_{31} k_{1p} + a_{32} k_{2p}),$$
.....
$$k_{7p} = h f(x_{j(p-1)} + c_7 h,\ y_{j(p-1)} + a_{71} k_{1p} + a_{72} k_{2p} + a_{73} k_{3p} + a_{74} k_{4p} + a_{75} k_{5p} + a_{76} k_{6p}),$$
$$y_{jp} = y_{j(p-1)} + b_1 k_{1p} + b_2 k_{2p} + b_3 k_{3p} + b_4 k_{4p} + b_5 k_{5p} + b_6 k_{6p} + b_7 k_{7p},\ p = \overline{1, n},$$

коэффициенты b_k, c_k, $a_{\ell m}$, $k = \overline{1, 7}$, $\ell = \overline{2, 7}$, $m = \overline{1, 6}$, приведены в обзорной главе (табл. 1.4). Метод Дормана-Принса 8-го порядка аппроксимации [35] записывается аналогично. Коэффициенты метода Дормана-Принса приведены в [127].

При использовании одного из данных методов с его помощью вычисляются входные значения y_{jp} в узлах интерполяции, остальные преобразования без изменений соответствуют описанным для случая применения метода Эйлера (2.26).

При решении задачи Коши для системы ОДУ вида (1.1) в предположении существования и единственности решения, схема приближенного решения на отрезке $[a, b]$ строится аналогично кусочно-полиномиальному приближению решения задачи (2.21) с таким изменением, когда все преобразования проводятся независимо для каждого компонента вектор-функции правой части (1.1).

Скорость сходимости кусочно-полиномиального приближения решения системы (1.1) оценивается аналогично случаю одного уравнения с переходом к канонической норме вектора погрешности.

Замечание 2.5. Объем вычислений на каждом интервале $[a_i, b_i]$ из (2.22) конечен, одинаков для всех i и зависит от порядка разностного метода (в качестве которого можно выбрать наименьший), а также линейно зависит от числа уравнений в (1.1), но не зависит от длины $[a, b]$.

Листинг программы *DP_S2*, реализующей варьируемое кусочно-полиномиальное решение задачи (1.1) на основе интерполяционного полинома

Ньютона с использованием узловых значений интерполяции, получаемых по методам Эйлера, Эйлера-Коши, Рунге-Кутты 4-го порядка, Бутчера 6-го порядка, а также Дормана-Принса 8-го порядка аппроксимации приводится в приложении к главе 2 (П2.3).

2.5. Экспериментальная проверка исследуемого метода обработки данных. В эксперимент включены системы ОДУ различной сложности, имеющие аналитически представимые решения. Погрешность их приближения оценивается в канонической норме вектора абсолютных прогрешностейкомпонентов. Решения рассматриваемых систем приближаются разностными методами, включая методы высоких порядков, а также варьирумыми кусочно-полиномиальными приближениями на их основе. Приводятся таблицы сравнения прогрешностей приближения.

Пример 2.1. Задача Коши

$$y_1' = x + 2x^{-1}y_1 - \sqrt{y_2},\ y_2' = 2\sqrt{y_2},\ y_1(1) = 2,\ y_2(1) = 4 \qquad (2.47)$$

имеет аналитическое решение [133] $y_1 = x + x^2,\ y_2 = (x+1)^2$. Решение задачи неустойчиво в смысле Ляпунова. Абсолютная погрешность его приближения разностными методами на отрезке $x \in [1, 10]$ приводится в табл. 2.2.

Таблица 2.2

Погрешность решения задачи (2.47) разностными методами

x	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*
	$h = 1.03\times10^{-9}$	$h = 1.03\times10^{-9}$	$h = 1.03\times10^{-5}$	$h = 1.03\times10^{-3}$	$h = 1.03\times10^{-2}$
2.03	1.967**e-09**	5.204**e-18**	8.674**e-19**	9.107**e-18**	6.072**e-18**
3.06	5.630**e-09**	1.041**e-17**	0.000**e+00**	3.123**e-17**	1.908**e-17**
…	…	…	…	…	…
8.21	4.736**e-08**	6.245**e-17**	2.082**e-17**	3.331**e-16**	1.457**e-16**
9.24	6.023**e-08**	9.714**e-17**	2.776**e-17**	4.302**e-16**	1.388**e-16**

Обозначения в табл. 2.2 аналогичны обозначениям в табл. 2.1. Граница погрешности 10^{-8} соответствует методу 1-го порядка (*Euler*), Методам 6-го (*Butcher_6*) и 8-го (*Dormand-Prince_8*) порядков соответствует граница

погрешности 10^{-16}. Значения шагов разностных методов соответствуют наиболее точному приближению решения задачи (уменьшение шага интегрирования не приводит к снижению погрешности). Граница абсолютной погрешности решения задачи (2.47) методом Рунге-Кутты 4-го порядка с шагом $h = 1.03 \times 10^{-5}$ не превышает порядка 10^{-17} (время выполнения программы: $202 ms$). Решение той же системы варьируемым разностно-полиномиальным методом в большинстве проверочных точек дает значения «нулевых» погрешностей в формате вывода данных (табл. 2.3).

Таблица 2.3

Погрешность варьируемого разностно-полиномиального решения задачи (2.47)

x	1	2	3	4	5
2.03	4.337**e-19**	4.337**e-19**	4.337**e-19**	4.337**e-19**	4.337**e-19**
3.06	8.674**e-19**	8.674**e-19**	0.000**e+00**	8.674**e-19**	8.674**e-19**
…	…	…	…	…	…
8.21	0.000**e+00**	0.000**e+00**	0.000**e+00**	0.000**e+00**	0.000**e+00**
9.24	0.000**e+00**	0.000**e+00**	0.000**e+00**	0.000**e+00**	0.000**e+00**

В табл. 2.3 и ниже цифры 1 – 5 обозначают варьируемые кусочно-полиномиальные приближения, значения в узлах интерполяции для которых вычислены соответственно методами Эйлера, Эйлера-Коши, Рунге-Кутты 4-го порядка, Бутчера 6-го и Дормана-Принса 8-го порядков. Кусочно-полиномиальный метод на основе разностных значений метода Эйлера применялся со следующими параметрами: $\nabla = 1$ из (2.22), количество итераций – $\widetilde{L} = 7$, значения степени полинома n и числа подынтервалов $P = 2^k$ варьировались программой (полученные значения параметров – $n = 1, 2, 3, 4$, $k = 2, 3, 4, 5$). Время выполнения программы: $28 ms$. Набор параметров в зависимости от номера интервала i распределен программой следующим образом:

Таблица 2.4

Параметры кусочно-полиномиального решения задачи (2.47) на основе метода Эйлера

i	0	1	2	3	4	5	6	7	8
n	2	1	1	4	1	4	1	3	2
k	5	5	4	3	4	2	4	2	2

В табл. 2.4 и ниже используются следующие обозначения: k – параметр, определяющий число подынтервалов $P=2^k$ из (2.23), n – степень интерполяционных полиномов, аппроксимирующих компоненты вектор-функции правой части (1.1), i – номер интервала из разбиения отрезка интегрирования (2.22) (длина интервала фиксирована и определяется значением параметра ∇). Согласно данным, впредсатвленным в табл. 2.4, значение расстояния между узлами интерполяции изменялось от $h \approx 1.563 \times 10^{-2}$ до $h = 1.25 \times 10^{-1}$.

Замечение 2.6. Повышение точности приближения на основе варьируемого разностно-полиномиального метода достигается при относительно больших значениях шага интегрирования, что существенно для задач, неустойчивых к возмущениям начальных условий. Такая неустойчивость характерна, в частности, для уравнений движения в небесной механике (глава 1, п. 1.1.2). В работе [134] для решения задачи Коши (1.1) разработано семейство численных методов на основе эрмитовых сплайнов и отмечается, что одним из путей преодоления проблемы численного приближения неустойчивых по Ляпунову решений является выбор или создание методов, сохраняющих высокую точность при достаточно больших величинах шагов интегрирования.

Пример 2.2. Задача Коши

$$y_1' = 2y_1 + 4y_2 + \cos(x), \quad y_2' = -y_1 - 2y_2 + \sin(x), \quad y_1(0) = -4, \quad y_2(0) = 1 \qquad (2.48)$$

имеет аналитическое решение $y_1 = -3\sin(x) - 2\cos(x) - 2$, $y_2 = 2\sin(x) + 1$. Погрешность его приближения разностными методами на отрезке $x \in [0, 10]$ приводится в табл. 2.5, варьируемым разностно-полиномиальным методом – в табл. 2.6.

Таблица 2.5

Погрешность решения задачи (2.48) разностными методами

x	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*
	$h = 1.03 \times 10^{-9}$	$h = 1.03 \times 10^{-9}$	$h = 1.03 \times 10^{-5}$	$h = 1.03 \times 10^{-2}$	$h = 1.03 \times 10^{-2}$
1.03	2.454**e-09**	1.575**e-15**	9.845**e-17**	3.903**e-18**	1.735**e-18**
2.06	4.996**e-09**	7.252**e-15**	2.301**e-16**	2.385**e-18**	7.372**e-18**
…	…	…	…	…	…
8.24	1.617**e-09**	6.844**e-14**	7.750**e-16**	3.253**e-17**	6.765**e-17**
9.27	2.064**e-09**	1.039**e-13**	1.001**e-15**	3.686**e-17**	9.452**e-17**

Методу 1-го порядка соответствует погрешность 10^{-9}, для методов 6-го и 8-го порядков аппроксимации абсолютная погрешность приближения достигает порядка 10^{-17}. Варьируемый разностно-полиномиальный метод приближает решение той же задачи с погрешностью порядка 10^{-18}:

Таблица 2.6

Погрешность варьируемого разностно-полиномиального решения задачи (2.48)

x	1	2	3	4	5
1.03	0.000**e+00**	0.000**e+00**	0.000**e+00**	4.337**e-19**	8.764**e-19**
2.06	4.337**e-19**	6.505**e-19**	2.168**e-19**	8.764**e-19**	3.036**e-18**
…	…	…	…	…	…
8.24	3.469**e-18**	4.770**e-18**	2.168**e-18**	2.240**e-18**	7.373**e-18**
9.27	5.773**e-18**	4.960**e-18**	1.355**e-18**	9.758**e-18**	8.457**e-18**

Параметры кусочно-полиномиального метода на основе разностных значений метода Эйлера: $\nabla = 1$, $\widetilde{L} = 4$, $n = 10, 12$, $k = 1, 2, 3$. Программное распределение набора параметров представлено в табл. 2.7.

Таблица 2.7

Параметры кусочно-полиномиального решения задачи (2.48) на основе метода Эйлера

i	1	2	3	4	5	6	7	8	9	10
n	10	12	10	10	10	10	10	12	10	10
k	2	1	3	2	2	2	2	1	3	3

Отсюда расстояние между узлами интерполяции при приближении решения задачи (2.48) разностно-полиномиальным методом изменялось от $h \approx 4.17 \times 10^{-2}$ до $h = 1.25 \times 10^{-2}$. Как следствие, время выполнения программы $< 1 ms$.

Аналогичные соотношения границ погрешности наблюдаются для различных нежестких систем ОДУ [127, 131], но минимум погрешности достигается не всегда в зависимости от вида (1.1).

Пример 2.3. Точное решение задачи Коши

$$\left.\begin{array}{l} y_1' = 2\,x\,y_4\,y_1,\ y_2' = 10\,x\,y_4\,y_1^5,\ y_3' = 2\,x\,y_4,\ y_4' = -2\,x\,(y_3 - 1), \\ y_i(0) = 1,\ i = 1,\ 2,\ 3,\ 4 \end{array}\right\} \qquad (2.49)$$

имеет вид [35]: $y_1 = e^{\sin(x^2)}$, $y_2 = e^{5\sin(x^2)}$, $y_3 = \sin(x^2) + 1$, $y_4 = \cos(x^2)$. Его разностные приближения на отрезке $[0, 10]$ представлены в табл. 2.8, варьируемые разностно-полиномиальные – в табл. 2.9:

Таблица 2.8

Погрешность решения задачи (2.49) разностными методами

x	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*
	$h = 1.03\times10^{-9}$	$h = 1.03\times10^{-9}$	$h = 1.03\times10^{-5}$	$h = 1.03\times10^{-4}$	$h = 1.03\times10^{-4}$
1.03	3.158**e-07**	2.880**e-08**	6.628**e-15**	3.518**e-15**	2.061**e-15**
2.06	7.599**e-07**	9.493**e-07**	6.096**e-16**	4.891**e-16**	1.449**e-15**
…	…	…	…	…	…
8.24	3.723**e-05**	4.650**e-05**	5.063**e-15**	1.324**e-15**	3.563**e-16**
9.27	5.382**e-05**	6.722**e-05**	5.519**e-15**	3.100**e-15**	1.904**e-15**

Наибольшую точность приближения порядка 10^{-15} дают методы 4-го, 6-го и 8-го порядка. Погрешность варьируемого разностно-полиномиального приближения не выше порядка 10^{-16}:

Таблица 2.9

Погрешность варьируемого разностно-полиномиального решения задачи (2.49)

x	1	2	3	4	5
1.03	6.245**e-17**	6.245**e-17**	6.245**e-17**	6.245**e-17**	6.245**e-17**
2.06	2.812**e-17**	2.812**e-17**	2.812**e-17**	2.812**e-17**	2.812**e-17**
…	…	…	…	…	…
8.24	4.424**e-17**	8.413**e-17**	8.066**e-17**	8.762**e-17**	6.894**e-17**
9.27	1.882**e-16**	1.963**e-17**	1.963**e-17**	1.963**e-17**	4.717**e-17**

Кусочно-полиномиальный метод на основе разностных значений метода Эйлера применялся со следующими параметрами: $\nabla = 1$, $\widetilde{L} = 10$, $n = 9, 10$, $k = 6, 7, 8$. Набор параметров в зависимости от номера интервала i:

Таблица 2.10

Параметры кусочно-полиномиального решения задачи (2.49) на основе метода Эйлера

i	1	2	3	4	5	6	7	8	9	10
n	9	9	10	10	10	10	10	10	10	10
k	6	8	7	7	8	8	8	8	8	8

Расстояние между узлами интерполяции согласно табл. 2.10 изменялось от $h \approx 3.9 \times 10^{-4}$ до $h \approx 1.74 \times 10^{-3}$. Время работы программы $609\,ms$.

Из представленной выше части эксперимента видно, что разница в точности при применении в качестве основы для вычисления значений в узлах интерполяции разностных методов высокого порядка по сравнению с результатами на основе метода Эйлера несущественна (табл. 2.3, табл. 2.6, табл. 2.9). Аналогичные результаты сохраняются для различных систем ОДУ [131]. Это означает, что уточнение достигается только за счет кусочно-полиномиальной модификации исходного метода. Такой результат численного эксперимента соответствует аналитическим оценкам погрешности, данным в п. 2.3.

Кусочно-полиномиальная схема на основе метода Эйлера не требует ограничений, кроме тех, которые обеспечивают существование и единственность решения, использование методов высших порядков требует наличия соответственных порядков производных [12]. В приводимых ниже примерах исследуется только кусочно-полиномиальное приближение на основе разностных значений метода Эйлера.

Как отмечалось выше, разностно-полиномиальный метод с итерационным уточнением применим и для решения жестких задач. Ниже приводятся результаты эксперимента с некоторыми модельными жесткими задачами, имеющими аналитически представимое решение.

Пример 2.4. Задача Коши $y' = y, y(0) = 1$, приближается на интервале $x \in [0, 30]$. Высокие порядки значений решения задачи и ее производной (в окрестности точки $x = 30$ порядок 10^{12}) затрудняют численное приближение решения. В табл. 2.11 приведены значения абсолютных погрешностей решения данной задачи, соответствующие наилучшим приближениям, полученным на основе разностных методов и с использованием рассматриваемого разностно-полиномиального метода. Названия столбцов таблицы соответствуют названиям функций или методов, применяемых для приближения решения:

- *Bulstoer* (*MathCAD*) – метод Булирша-Штера (используется для решения систем ОДУ первого порядка с гладкими правыми частями [32, 35]),
- *ode45* (*MATLAB*)– метод Дормана-Принса 5 (4) [54, 135],
- *Butcher_6*– метод Бутчера 6-го порядка,
- *Dormand-Prince_8*– метод Дормана-Принса 8-го порядка,
- *Radau* (*SciPy*)–функция, реализующая метод Радо IIA 5-го порядка [3] (неявный метод Рунге-Кутты). Функция реализована в *SciPy* – библиотеке для языка программирования *Python* с открытым исходным кодом [136], предназначенной для выполнения научных и инженерных расчетов.
- *DiffPol* – кусочно-полиномиальный метод на основе разностных значений метода Эйлера.

Таблица 2.11

Абсолютная погрешность приближения решения задачи $y' = y,\ y(0) = 1$

x	*Bulstoer* (*MathCAD*)	*ode45* (*MATLAB*)	*Radau* (*SciPy*)	*Butcher_6*	*Dormand-Prince_8*	*DiffPol*
0.7	3.553**e-15**	3.553**e-15**	3.109**e-15**	2.602**e-18**	2.385**e-18**	2.168**e-19**
6.7	2.512**e-11**	2.444**e-11**	5.116**e-12**	3.386**e-15**	1.055**e-15**	3.331**e-16**
…	…	…	…	…	…	…
14.7	2.366**e-07**	4.410**e-07**	3.027**e-08**	3.865**e-12**	2.342**e-11**	0.000**e+00**
19.7	7.749**e-07**	1.360**e-07**	3.994**e-06**	6.112**e-10**	7.421**e-09**	2.037**e-10**
24.7	7.153**e-03**	3.593**e-03**	3.738**e-04**	2.608**e-08**	8.084**e-07**	5.215**e-08**
29.7	2.100**e+00**	7.197**e-01**	7.715**e-02**	1.576**e-05**	2.093**e-04**	9.537**e-07**

Среди разностных методов наименьшая погрешность 10^{-5} соответствует методу 6-го подрядка (*Butcher_6*) с шагом $h = 10^{-3}$. Методам *Bulstoer*, *ode45* и

Radau соответствует наименьшее значение границы абсолютной погрешности решения данной задачи среди методов решения систем ОДУ, представленных в соответствующих программных пакетах, включая методы решения жестких задач. Варьируемый разностно-полиномиальный метод дает приближение решения, с точностью до коэффициента, с погрешностью 10^{-7}. Значения параметров метода: $k=4$, $n=8$, $\nabla=1$, $\widetilde{L}=10$, что соответствует расстоянию между узлами интерполяции $h=7.8125\times10^{-3}$.

Пример 2.5. Жесткая задача Коши [134]:

$$\left.\begin{aligned} &y_1'=-400\,(y_2-1)-2\,y_1\,x,\ y_2'=5(1-0.003\,y_1),\\ &y_1(0)=0,\ y_2(0)=1 \end{aligned}\right\} \qquad (2.50)$$

имеет аналитическое решение $y_1=-1000\,x^2$, $y_2=5\,x(1+x^2)+1$. Коэффициент жесткости, определенный (согласно [78]) как отношение максимального абсолютного значения собственных чисел матрицы Якоби к минимальному, для данной задачи характеризуется порядком 10^4. В табл. 2.12 приведены канонические нормы значений погрешностей его приближения на отрезке $x\in[0,10]$, соответствующие наилучшим приближениям, полученным на основе разностных методов и с использованием рассматриваемого метода.

Таблица 2.12

Абсолютная погрешность приближения решения задачи (2.50)

x	*Radau* (*MathCAD*)	*ode45* (*MATLAB*)	*Radau* (*SciPy*)	*Butcher_6*	*Dormand-Prince_8*	*DiffPol*
1.03	2.274**e-12**	2.728**e-12**	3.553**e-15**	1.554**e-15**	7.772**e-16**	1.110**e-16**
2.06	1.182**e-11**	9.095**e-13**	3.638**e-12**	1.732**e-14**	1.776**e-15**	4.441**e-16**
3.09	1.273**e-11**	3.456**e-11**	9.095**e-12**	5.773**e-14**	9.948**e-14**	0.000**e+00**
…	…	…	…	…	…	…
8.24	1.455**e-11**	1.019**e-10**	8.731**e-11**	7.816**e-14**	9.521**e-13**	7.105**e-15**
9.27	1.455**e-11**	5.239**e-10**	1.164**e-10**	2.629**e-13**	7.745**e-13**	7.105**e-15**

Для методов 6-го и 8-го порядков аппроксимации абсолютная погрешность приближения достигает порядка 10^{-13}. Варьируемый разностно-

полиномиальный метод приближает решение с погрешностью 10^{-15}. Параметры метода: $\nabla = 1,\ \widetilde{L} = 7,\ n = 2, 4,\ k = 6, 7, 8$.

Пример 2.6. Решение задачи [18]:

$$y_1' = \frac{y_1^2}{y_2 - x},\ y_2' = y_1 + 1,\ y_1(0) = 1,\ y_2(0) = 1/3 \qquad (2.51)$$

характеризуется высоким порядком значения производной, так как содержит быстрорастущую составляющую e^{3x}: $y_1(x) = e^{3x},\ y_2(x) = x + e^{3x}/3$. В табл. 2.13 приведены значения погрешностей приближенного решения задачи (2.51) на отрезке $x \in [0, 6]$, соответствующие наилучшим приближениям, полученным на основе разностных методов и с использованием рассматриваемого метода.

Таблица 2.13

Абсолютная погрешность приближения решения задачи (2.51)

x	*Rkadapt* (*MathCAD*)	*ode45* (*MATLAB*)	*Radau* (*SciPy*)	*Butcher_6*	*Dormand-Prince_8*	*DiffPol*
1.0	1.528**e-13**	1.421**e-13**	2.416**e-13**	8.674**e-17**	1.839**e-16**	1.735**e-18**
2.0	7.162**e-12**	7.219**e-12**	1.171**e-11**	3.136**e-15**	3.830**e-15**	8.327**e-17**
3.0	1.446**e-10**	1.837**e-10**	3.292**e-10**	1.075**e-13**	7.994**e-15**	0.000**e+00**
4.0	3.667**e-09**	1.863**e-09**	4.482**e-09**	1.009**e-12**	1.180**e-12**	4.263**e-14**
5.0	5.495**e-08**	2.794**e-08**	2.654**e-08**	2.183**e-11**	4.138**e-11**	2.274**e-12**
6.0	1.580**e-06**	7.451**e-07**	1.818**e-06**	1.368**e-09**	5.675**e-10**	4.729**e-11**

Наибольшую точность приближения порядка 10^{-10} среди представленных в таблице разностных методов дает метод Дормана-Принса 8-го порядка. Погрешность варьируемого разностно-полиномиального приближения не выше порядка 10^{-11} на всем отрезке интегрирования. Параметры метода: $\nabla = 1,\ \widetilde{L} = 10,\ n = 8,\ k = 5$. В [18] приводятся значения максимальной относительной погрешности приближения задачи (2.51) при $x = 6$ с применением численно-аналитического метода решения систем ОДУ, основанного на приближенном представлении решения и его производной в виде частичных сумм смещенных рядов Чебышева. При разбиении отрезка интегрирования на несколько частичных сегментов граница относительной погрешности приближения не превышает порядок 10^{-13}. Граница погрешности не снижается при решении

задачи многозначным методом Гира [137] с автоматическим выбором шага и переменным порядком (максимальный допустимый порядок равен семи) и одношаговыми методами Фельберга и Ингленда пятого порядка точности с автоматическим выбором шага [35, 138]. Относительная погрешность разностно-полиномиального приближения с указанными выше параметрами программы не превышает порядок 10^{-19} на всем отрезке приближения.

Как показывает эксперимент, решение нежестких задач с применением варьируемого разностно-полиномиального метода приближается с границей абсолютной погрешности порядка 10^{-19}, 10^{-18}. Указанная точность сохраняется и для задач, не устойчивых к возмущению начальных данных. Метод инвариантен относительно жестких и нежестких задач. Сравнительно низкая погрешность решения жестких задач достигается при допустимой границе трудоемкости.

2.6. Оценка трудоемкости метода. В [3, 35] трудоемкость оценивается числом обращений к подпрограмме вычисления функции правой части (1.1). Ниже трудоемкость T_{ri} (i – индекс интервала, r – вспомогательный индекс группы операций) оценивается числом последовательных арифметических операций на отрезке решения с учетом обращений к подпрограмме. При оценке сверху рассматривается кусочно-полиномиальное приближение на основе метода Эйлера, причем первоначально – для решения задачи (2.21). Для простоты k и n предполагаются априори зафиксированными. На интервале $[a_i, b_i]$ из (2.22) метод Эйлера вызывает правую часть (2.21) $2^k n$ раз для задания $n+1$ узлового значения интерполяционного полинома n-й степени на каждом из 2^k подынтервалов: $y_{j\,(p+1)} = y_{jp} + h \cdot f(x_{jp}, y_{jp})$, $p = \overline{0, n-1}$. Отсюда $T_{1\,i} = 2^k n\,(t_f + t_y + t_c)$, где t_f – время вычисления правой части (2.21), t_y, t_c – время бинарного умножения и сложения. Кроме того, на каждом подынтервале из (2.23) выполняется построение полинома Ньютона n-й степени, откуда с

учетом соотношений (2.28) по всем подынтервалам следует равенство $T_{2i} = 2^k (0.5(n^2 + n)\, t_y + n^2 t_c)$. Время вычисления полинома согласно (2.31) соответствует $T_{3i} = 2^k (n+1)\,(t_y + t_d + t_c)$, где t_d – время бинарного деления. Сложение оценок приводит к неравенству $\sum_r T_{r\,i} \le 2^k Q_i$, где $Q_i = n\, t_f + 0.5\,(n^2 + 5n + 2)\, t_y + (n^2 + 2n + 1)\, t_c + (n+1)\, t_d$. Остается учесть итерационное уточнение. Уточнение выполняется на каждом подынтервале путем вычисления на текущей итерации интерполяционного полинома; после задания значений в узлах интерполяции исключаются разностные шаги. Это приводит к следующей оценке: $T_{4i} \le 2^k \widetilde{L}\,(Q_i - n(t_y + t_c))$. Сложение с предыдущим неравенством обуславливает оценку трудоемкости T_{Si} на интервале $[a_i, b_i]$:

$$T_{Si} \le 2^k (\widetilde{L} + 1) Q_i,\; Q_i \sim n^2 \left(t_c + t_y / 2\right) + n \left(t_f + t_d\right).$$

После перехода от i-го к $(i+1)$-му интервалу все операции повторяются, верхняя граница их числа на интервале не увеличивается с ростом i. В условиях следствия 2.5 граница ε погрешности приближения на i-м интервале достигается за счет роста числа этапов k его разбиения на $P = 2^k$ подынтервалов при ограничении $1 \le n \le n_0$ (рис. 2.1).

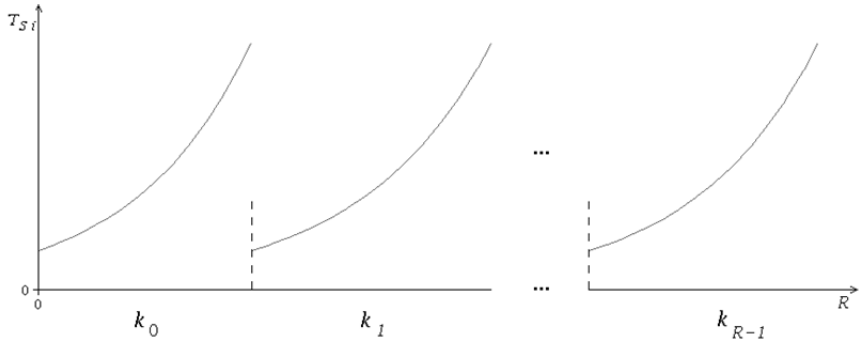

Рис. 2.1. Периодичность роста границы трудоемкости разностно-полиномиального приближения

Здесь $k_i = k_i(\varepsilon)$ – число этапов на i-м интервале, R определяется из (2.22). Граница трудоемкости на отрезке $[a, b]$ периодически возрастает в геометрической прогрессии на каждом отрезке $[a_i, b_i]$. При традиционном [35] измерении посредством $\log_{10} T_{Si}$ графики границы трудоемкости преобразуются к виду линейных зависимостей.

В системе (1.1) изложенные операции повторяются для каждого уравнения в отдельности, увеличение размерности изменит значения на графике пропорционально числу уравнений, сохранив в целом периодический вид и характер зависимости.

Ниже, для примера, сопоставлено время решения на персональном компьютере рассмотренными методами задачи (2.49) с точностью, указанной в табл. 2.8, 2.9:

Таблица 2.14

Время решения задачи (2.49) разностными и варьируемыми разностно-полиномиальными методами

	Euler	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*
D	0:11:10:781	0:34:20:606	0:0:0:297	0:0:0:47	0:0:0:110
DiffPol	0:0:0:609	0:0:0:625	0:0:0:640	0:0:0:656	0:0:0:718

В табл. 2.14 названия столбцов соответствуют разностным методам, обозначения *D*, *DiffPol* в строках соответствуют разностному и варьируемому кусочно-полиномиальному приближению на основе данных разностных методов, двоеточия отделяют часы, минуты, секунды, миллисекунды. При фиксировании параметров время компьютерной реализации кусочно-полиномиального приближения существенно снижается. Так, время разностно-полиномиального решения жестких задач из примеров 2.4, 2.6 менее одной миллисекунды.

Изложенный метод обладает резервом быстродействия на основе параллелизма построения полинома Ньютона на каждом подынтервале с оценкой (2.18).

2.7. Параллелизм обработки данных в ИВС в приложении к дифференциальным моделям. Предварительно рассматривается схема на основе метода Эйлера. Построение кусочно-полиномиальной аппроксимации выполняется последовательно по подынтервалам из (2.23), за одинаковое время на каждом подынтервале. После выполнения n шагов метода Эйлера на подынтервале $[x_j, x_{j+1}]$ из (2.23) аппроксимирующий правую часть ОДУ полином (2.28) строится с временной сложностью $O(\log_2 n)$ (2.18). Переход от этого полинома к табличной первообразной для аппроксимации решения сохраняет оценку по величине порядка. Ввиду последовательности из n разностных шагов на каждом подынтервале оценка (2.18) возрастает на $O(n)$, скорректированную оценку можно отнести только к отдельному подынтервалу из (2.23) [126].

При решении задачи (1.1) имеет место естественный параллелизм по уравнениям системы, время построения каждого компонента остается таким же, как в только что скорректированной оценке, число процессоров увеличится соответственно числу уравнений.

Все изменения, связанные с реализацией программного подбора степени полинома и количества подынтервалов, можно выполнять синхронно и взаимно независимо, длительность выполнения варьируемого кусочно-полиномиального метода будет соизмерима с оценкой (2.18), скорректированной на $O(n)$, число процессоров возрастет пропорционально произведению числа варьируемых значений n на число варьируемых значений k.

Замечание 2.6. В случае если точность решения определяется наперед заданным значением невязки, при этом текущие значения невязки сравниваются по уменьшаемым шагам согласно методу Рунге [35], то возможен следующий вариант параллельной схемы. Для каждого наперед заданного значения невязки вычисления посредством варьируемого кусочно-полиномиального метода можно производить отдельно и взаимно независимо

(параллельно) по всем заданным значениям невязки. Результат достижения точности определяется путем взаимного сравнения невязки, достигнутой на каждой параллельной ветви (на основе межпроцессорного обмена). Именно, совпавшие значения значащих цифр считаются верными, процесс останавливается при совпадении искомого количества цифр хотя бы в двух параллельных ветвях, выбираются параметры той ветви, которая обладает наименьшей временной сложностью. При этом для каждой параллельной ветви (для каждого отдельного значения невязки) сохраняются все описанные схемы параллелизма варьируемого кусочно-полиномиального метода.

Аналогично, параллельный алгоритм кусочно-полиномиального решения задачи Коши для системы (1.1) строится на основе других разностных методов.

При моделировании возникают следующие дополнительные возможности использования параллелизма кусочно-полиномиального решения ОДУ.

Параллелизм кусочно-полиномиальной аппроксимации функций (п 2.1.2) может непосредственно использоваться для ускорения вычисления правых частей систем вида (1.1). Вычисление значений полинома можно производить с использованием параллельных алгоритмов [75, 139]. При моделировании процесса с помощью системы (1.1) кусочно-полиномиальные приближения решения можно строить параллельно для различных начальных значений и различных границ погрешности. Моделирующие решения воспроизводятся параллельно по множеству подынтервалов путем считывания из памяти хранимых коэффициентов аппроксимирующих решение полиномов.

Кусочно-полиномиальное приближение компонента решения (1.1) аналитично внутри каждого подынтервала, непрерывно и дает непрерывное приближение производной от компонента решения на всем отрезке решения, в данном аспекте после предварительного вычисления и запоминания коэффициентов рассматриваемых полиномиальных приближений сами приближения можно воспроизводить параллельно по всем подынтервалам.

Заметим, что при существовании и непрерывности высших производных от решения на некотором отрезке, их можно приближать с помощью табличных

производных от полинома (2.29), причем непрерывно на всем отрезке, выполняя интерполяцию по замечанию 2.2, – путем склеивания интерполируемых значений на границах подынтервалов. Однако в этом случае нельзя гарантировать ту точность приближения, которая имела место для самого решения и его первой производной. Аналогично предыдущему, в процессе моделирования рассматриваемые приближения производных допускают параллельное воспроизведение на отрезке их построения.

Таким образом, в главе предложен метод разностно-полиномиальной обработки данных в ИВС с программным выбором варьируемых параметров для моделей периодических процессов. Метод позволяет повысить точность и уменьшить время обработки данных относительно известных методов, а также улучшить качество моделирования исследуемых процессов. Показана сходимость метода, дана оценка скорости сходимости. Для корректного применения метода достаточно двукратной дифференцируемости правой части дифференциальной системы, что положительно отличается от условий применения аналогов и улучшает качество моделирования в ИВС периодических процессов с быстро меняющейся динамикой.

2.8. Выводы

1. Разработан метод разностно-полиномиальной обработки данных в ИВС с программным выбором варьируемых параметров для моделей периодических процессов. Метод отличается от известных аналогов обработкой данных на временных подынтервалах интерполяционными полиномами Ньютона, программно преобразуемыми в форму алгебраических полиномов с числовыми коэффициентами, разностной обработкой узловых значений, применением итерационного уточнения, автоматизированным выбором параметров, что позволяет повысить точность и уменьшить время обработки данных относительно известных методов, а также улучшить качество моделирования исследуемых процессов.

2. Показана сходимость предложенного метода, дана оценка скорости сходимости. Для корректно обоснованного применения метода достаточно двукратной дифференцируемости правой части дифференциальной системы, что положительно отличается от условий применения аналогов, на практике обеспечивает применимость для моделирования в ИВС периодических процессов с быстро меняющейся динамикой и подтверждается результатами численных экспериментов.

3. Предложены схемы динамического распараллеливания компьютерного кусочно-полиномиального приближения функций в ИВС с параллельной архитектурой, применимые при разностно-полиномиальной обработке данных, даны верхние оценки временной сложности данных схем.

4. Предложены разновидности метода разностно-полиномиальной обработки данных моделей периодических процессов на основе высоких порядков приближений узловых значений. Экспериментально показано, что точность приближения данных при вариации числа подынтервалов и степени интерполяционного полинома совпадает во всех данных разновидностях метода, при этом она превышает точность известных аналогов.

5. Выполнена экспериментальная проверка предложенного метода, представлены оценки трудоемкости. Программная реализация метода отличается быстродействием и высокой точностью обработки данных применительно к широкому классу моделей периодических процессов, а также непрерывностью аналитического приближения данных.

ГЛАВА 3. МЕТОД КУСОЧНО-ИНТЕРПОЛЯЦИОННОЙ ОБРАБОТКИ ДАННЫХ С ИТЕРАЦИОННЫМ УТОЧНЕНИЕМ ДЛЯ МОДЕЛЕЙ ПЕРИОДИЧЕСКИХ ПРОЦЕССОВ

В главе представлен метод кусочно-интерполяционной обработки данных с итерационным уточнением являющийся модификацией описанного в главе 2 разностно-полиномиального метода обработки данных с автоматизированным выбором варьируемых параметров для моделей периодических процессов. Модификация построена на основе кусочной интерполяции на временных подынтервалах правой части дифференциальной системы и интегральном приближении решения в виде алгебраических полиномов с числовыми коэффициентами, использует итерационные уточнения аналогичные интегральным приближениям Пикара. В отличие от подхода, описанного в главе 2, разностные схемы не входят в состав метода, узловые значения определяются на основе итерационного уточнения: итерационной подстановкой в правую часть первообразной от интерполяционного полинома на каждом временном подынтервале. В программной реализации на общих границах смежных подынтервалов берутся равные узловые значения интерполяции и совпадающие значения констант первообразных. Для восстановления первообразной интерполяционный полином преобразуется к виду алгебраического полинома с числовыми коэффициентами. Ниже показано, что в условиях двукратной непрерывной дифференцируемости правой части дифференциальной системы метод равномерно сходится к решению с ростом числа подынтервалов, равномерно также сходится приближение к его производной, даны оценки скорости сходимости с учетом итерационного уточнения. Качества равномерной сходимости и аналитический вид приближения числовых данных отличают предложенную модификацию от известных аналогов. Метод реализован в виде стандартной программы ИВС. Автоматический выбор параметров обеспечивает наибольшую точность при наименьшем времени обработки. Это отличительное качество программной

реализации позволяет превышать точность аналогов на два десятичных порядка, при этом варьируемые параметры программно адаптируются к структуре модели и реализуют динамическую коррекцию начальных данных. В результате достигается вычислительная устойчивость, высокая точность, гладкость аналитического приближения данных на отрезке произвольной длины, что составляет отличие метода для широкого класса моделей периодических процессов в ИВС.

При внесении незначительных изменений метод распространяется на случай обработки данных дифференциальных моделей, представимых в виде нелинейной двухточечной задачи Коши с точными значениями в начале и на конце промежутка. В отличие от методов на основе конечно-разностных схем [19, 20] кусочно-интерполяционный метод с итерационным уточнением строится одновременно от начальной точки по направлению конечной и от конечной к начальной. В окрестности точки пересечения встречные приближения интерполируются, затем применяется итерационное уточнение. Метод позволяет получить непрерывное и непрерывно дифференцируемое компьютерное приближение решения рассматриваемого частного вида двухточечной задачи Коши с характерно малой погрешностью, описанной для одноточечной задачи Коши.

Материал главы соответствует содержанию публикаций автора [76, 140 – 143, 145, 146].

Пусть вначале рассматривается задача Коши для ОДУ в прямоугольной области R:

$$\begin{aligned} &y' = f(x,y),\ y(x_0) = y_0,\ (x,y) \in R, \\ &R: \{ x_0 \le x \le x_{fin}; | y - y_0 | \le b;\ x_{fin} = \text{const},\ b = \text{const} \}, \end{aligned} \tag{3.1}$$

где функция $f(x,y)$ определена, непрерывна, непрерывно дифференцируема (в точке x_0 – справа, в точке x_{fin} – слева) и удовлетворяет условию Липшица

$$| f(x,y) - f(x,\tilde{y}) | \le L | y - \tilde{y} |,\ L = \text{const},\ \forall (x,y), (x,\tilde{y}) \in R. \tag{3.2}$$

Предполагается, что решение задачи (3.1) существует и единственно в R. На i-м из множества отрезков, покрывающем $[x_0, x_{fin}]$, функция правой части $f(x, \varphi_i(x))$, где $y \approx \varphi_i(x)$, интерполируется полиномом Ньютона $\Psi_{in}(t)$, преобразуемым к виду алгебраического полинома с числовыми коэффициентами

$$\Psi_{in}(t) = a_{i0} + \sum_{\ell=1}^{n} a_{i\ell} t^{\ell}, \; f(x, \varphi_i(x)) \approx \Psi_{in}(t), \; t = (x - x_i)/h.$$

Решение приближается функцией $\tilde{\varphi}_i(x) = C_i + h\sum_{\ell=0}^{n} \frac{a_{i\ell}}{\ell+1} t^{\ell+1}$. Количество отрезков и степень полинома априори варьируются и фиксируются при минимизации невязки. На выходе вариаций приближение на i-м отрезке $y \approx \tilde{\varphi}_i(x)$ подставляется в правую часть, $f(x, \tilde{\varphi}_i(x))$ снова интерполируется на этом же отрезке, от интерполирующего полинома берется первообразная и повторно подставляется в правую часть. Число итераций – параметр метода, по окончании итераций выполняется переход к следующему отрезку, где воспроизводится аналогичный процесс.

Ниже показана возможность сравнительно высокой точности приближенного решения рассматриваемой задачи, представлены ограничения, выполнена оценка скорости сходимости и трудоемкости излагаемого метода.

3.1. Кусочно-интерполяционное приближение решения и его производной в дифференциальных моделях обработки данных в ИВС. С приводимыми ниже отличиями по изложенной в п. 2.1 схеме выполняется кусочно-полиномиальная аппроксимация $f(x, \varphi(x))$ из (3.1), где $y \approx \varphi(x)$. Основное отличие от аппроксимации, данной в п. 2.2 второй главы, заключается в том, что в качестве узловых значений интерполяции не используются пошаговые приближения разностных методов. Именно, в правую часть (3.1) вместо y подставляется y_0. Функция $f(x, y_0)$ приближается

полиномами вида (2.10) по изложенной в п. 2.1 схеме так, как если бы $u(x)$ равнялось $f(x, y_0)$. При фиксированных на выходе алгоритма значениях n и k на отрезке $[x_{i-1}, x_i]$ из (2.1), вначале при $i=1$, затем, аналогично, при каждом следующем i, выполняется итерационное уточнение искомого приближения. Вначале первообразная от найденного полинома

$$P_{(i-1)\,n+1}(x) = y_{0(i-1)} + \int_{x_{i-1}}^{x} \Psi_{(i-1)n}(t)\,dx \text{ в виде } y_{0(i-1)} + h\sum_{\ell=0}^{n} \frac{a_{(i-1)\ell}}{\ell+1} t^{\ell+1}$$

принимается за приближение решения на данном отрезке: $y(x) \approx P_{(i-1)\,n+1}(x)$. Затем y в правой части (3.1) заменяется на $P_{(i-1)\,n+1}(x)$: $f(x,y) \approx f(x, P_{(i-1)\,n+1}(x))$. После замены при той же степени n на том же отрезке строится аппроксимирующий полученную функцию полином Ньютона, преобразованный в форму (2.10): $\Psi_{(i-1)n}(t) \approx f(x, P_{(i-1)\,n+1}(x))$. От $\Psi_{(i-1)n}(t)$ снова берется первообразная с тем же значением константы

$$P^{(1)}_{(i-1)\,n+1}(x) = y_{0(i-1)} + \int_{x_{i-1}}^{x} \Psi_{(i-1)n}(t)\,dx \text{ в виде } y_{0(i-1)} + h\sum_{\ell=0}^{n} \frac{a^{(1)}_{(i-1)\ell}}{\ell+1} t^{\ell+1},$$

затем этот полином подставляется в правую часть, $f(x,y) \approx f(x, P^{(1)}_{(i-1)\,n+1}(x))$, которая аппроксимируется аналогично: $\Psi^{(1)}_{(i-1)n}(t) \approx f(x, P^{(1)}_{(i-1)\,n+1}(x))$. Итерации

$$P^{(\ell)}_{(i-1)\,n+1}(x) = y_{0(i-1)} + \int_{x_{i-1}}^{x} \Psi^{(\ell-1)}_{(i-1)n}(t)\,dx,\ \Psi^{(\ell)}_{(i-1)n}(t) \approx f(x, P^{(\ell)}_{(i-1)\,n+1}(x)),\ \ell = 1, 2, \ldots,$$

где $\Psi^{(0)}_{(i-1)n}(t) = \Psi_{(i-1)n}(t)$, продолжаются до достижения заданной малости невязки, при реализации их число ограничивается: $\ell \le q = \text{const}$. Здесь и выше $t = (x - x_i)/h,\ i = \overline{0, P-1},\ P = 2^k, k = 0, 1, \ldots,$ в дальнейшем структура итераций формализуется.

Именно после окончания итераций на $[x_{i-1}, x_i]$, при соответственном q, в качестве начального значения для отрезка $[x_i, x_{i+1}]$ полагается

$y_{0i} = P^{(q)}_{(i-1)\,n+1}(x_i)$. Если итерационное уточнение не выполняется, то $y_{0i} = P_{(i-1)\,n+1}(x_i)$.

Данным способом обрабатываются приближения на каждом отрезке из (2.1). В свою очередь весь отрезок решения задачи (3.1) разбит на отрезки равной длины:

$$[x_0, x_{fin}] = \bigcup_{r=0}^{M-1} [\alpha_r, \beta_r],\ \alpha_{r+1} = \beta_r \ \ \forall r = \overline{0, M-2}, \tag{3.3}$$

каждый отрезок $[\alpha_r, \beta_r]$ обрабатывается, как только что описано для $[\alpha, \beta]$ из (2.1), при этом вместо (2.1) рассматривается

$$[\alpha_r, \beta_r] = \bigcup_{i=0}^{P-1} [x_{ri}\ ,\ x_{r\,i+1}],\ r = 0,1,\ldots,M-1,\ P = 2^k,\ k = 0, 1, \ldots. \tag{3.4}$$

За начальное значение на левой границе $[x_{ri}\ ,\ x_{r\,i+1}]$ всегда принимается конечное значение на правой границе предыдущего отрезка $[x_{r\,i-1}\ ,\ x_{ri}]$: $y_{r\,0i} = P^{(q)}_{r(i-1)\,n+1}(x_{ri})$, $i = 1, 2, ..., P-1$, аналогично, при переходе от $[\alpha_r, \beta_r]$ к $[\alpha_{r+1}, \beta_{r+1}]$.

Предварительные (без учета итерационного уточнения) оценки показывают равномерную сходимость кусочно-интерполяционного метода на $[\alpha_r, \beta_r]$, а также на объединении (3.3) при $k \to \infty$. Пусть произвольно выбрано n, $1 \le n \le n_0$, $n_0 = \text{const}$. Предполагается, что функция $f(x, y(x))$ имеет $n+1$ непрерывную производную во всей области R из (3.1), на границах $[x_0, x_{fin}]$ производные понимаются как односторонние. Вводится обозначение $f(x, \tilde{y}(x)) \approx f(x, y(x))$, где $\tilde{y}(x)$ – приближение функции $y(x)$ с помощью полинома $P^{(\ell)}_{(i-1)\,n+1}(x)$ без итерационного уточнения: $\ell = 1$.

При $P=1,\quad i=0,\quad [x_{r0}, x_{r1}]=[\alpha_r, \beta_r]$ погрешность приближения $f(x,\tilde{y}(x))$ на данном отрезке полиномом (2.4) в форме (2.10), обозначаемым $\Psi_{r0n}(t)$, оценивается из неравенства [124, 144]:

$$|f(x,\tilde{y}(x))-\Psi_{r0n}(t)| \le \frac{\max\limits_{\xi\in[\alpha_r,\beta_r]} |f^{(n+1)}(\xi, \tilde{y}(\xi))|\, h^{n+1}}{(n+1)!} \left|\prod_{\ell=0}^{n}(t-\ell)\right|,\ x\in[x_{r0}, x_{r1}], \quad (3.5)$$

где $h=(x_{r1}-x_{r0})/n,\ t=(x-x_{r0})/h$. Оценка соответствует $k=0$ в (3.4), что интерпретируется как 0-й этап разбиения $[\alpha_r, \beta_r]$. Узлы интерполяции обозначаются через $z_{rj}=x_{r0}+jh,\quad j=\overline{0,n}$. Если $x\in[z_{rj}, z_{rj+1}]$, то $t-\ell=(x-z_{r\ell})/h,\ \ell=\overline{0,n}$, и из (3.5) имеем

$$|f(x,\tilde{y}(x))-\Psi_{r0n}(t)| \le \frac{c\,h^{n+1}}{(n+1)!}\left|\prod_{\ell=0}^{j}\frac{x-z_{r\ell}}{h}\prod_{\ell=j+1}^{n}\frac{x-z_{r\ell}}{h}\right|,$$

где $1\le n\le n_0$, при этом

$$c=\max_{R}\max_{1\le n\le n_0}\max_{\xi\in[x_0,x_{fin}]}|f^{(n+1)}(\xi,\tilde{y}(\xi))|,\ c=\text{const}. \quad (3.6)$$

Так как $z_{rj}\le x\le z_{rj+1}$, то

$$|f(x,\tilde{y}(x))-\Psi_{r0n}(t)| \le \frac{c\,h^{n+1}}{(n+1)!}\prod_{\ell=0}^{j}\frac{|z_{rj+1}-z_{r\ell}|}{h}\prod_{\ell=j+1}^{n}\frac{|z_{rj}-z_{r\ell}|}{h}.$$

С учетом $|z_{rj+1}-z_{r\ell}|=(j+1-\ell)h$ при $\ell=0,1,\ldots,j$ и $|z_{rj}-z_{r\ell}|=kh$ при $k=1,2,\ldots,n-j,\ \ell=j+1,j+2,\ldots,n$, имеем:

$$|f(x,\tilde{y}(x))-\Psi_{r0n}(t)| \le \frac{c\,h^{n+1}}{(n+1)!}(j+1)!\,(n-j)!.$$

Поскольку $(j+1)!\,(n-j)!\le(n+1)!$, при любом выборе $j=\overline{0,n-1}$ получится:

$$P=1,\ [\alpha_r,\beta_r]=[x_{r0},x_{r1}],\ |f(x,\tilde{y}(x))-\Psi_{r0n}(t)|\le c\,h^{n+1}\ \forall x\in[x_{r0},x_{r1}]. \quad (3.7)$$

Если $P=2$, и на каждом из отрезков $[x_{r0}, x_{r1}]$, $[x_{r1}, x_{r2}]$ разбиения (3.4) выполняется интерполяция полиномом (2.4) с тем же значением n, то с учетом равенства числа узлов, отстоящих друг от друга на каждом отрезке на $h/2$, получится аналог соотношения (3.7) с вдвое меньшим h: $|f(x,\tilde{y}(x))-\Psi_{rin}(t)| \le c\,(h/2)^{n+1}$, $i=0,1$. Более точно,

$$P=2,\ [\alpha_r,\beta_r]=\bigcup_{i=0}^{1}[x_{ri},x_{r\,i+1}],\ |f(x,\tilde{y}(x))-\Psi_{rin}(t)| \le c\,2^{-(n+1)}\,h^{n+1}$$
$$\forall i=\overline{0,1} \wedge \forall x \in [x_{ri},x_{r\,i+1}], \tag{3.8}$$

где c из (3.6), $t=(x-x_{ri})(h/2)^{-1}$ при каждом $i=0,1$. Здесь и ниже полином вида (2.4) в форме (2.10) обозначается $\Psi_{rin}(t)$ в соответствии с индексами из (3.3), (3.4). В (3.8) и всюду ниже h – расстояние между узлами интерполяционного полинома $\Psi_{r0n}(t)$ на начальном отрезке $[\alpha_r,\beta_r]$, которое в дальнейшем не меняется при его разбиении на $P=2^k$ отрезков ($k=1,2,\ldots$). По индукции удвоение числа подынтервалов (3.4) уменьшает дробь в (3.8) в 2^{n+1}, и для всех $k \ge 0$ справедливо соотношение (см. 2.15):

$$P=2^k,\ [\alpha_r,\beta_r]=\bigcup_{i=0}^{2^k-1}[x_{ri},x_{r\,i+1}],\ |f(x,\tilde{y}(x))-\Psi_{rin}(t)| \le c\,2^{-k(n+1)}\,h^{n+1},$$
$$\forall i=\overline{0,2^k-1} \wedge \forall x \in [x_{ri},x_{r\,i+1}], \tag{3.9}$$

h – шаг интерполирования полинома $\Psi_{r0n}(t)$ на $[\alpha_r,\beta_r]$ при $k=0$. Из (3.9), каково бы ни было n, $1 \le n \le n_0$, последовательность полиномов $\Psi_{rin}(t)$ равномерно сходится к функции $f(x,\tilde{y}(x))$ на данном отрезке при $k \to \infty$. Правая часть неравенства (3.9) не зависит от r, поэтому равномерная сходимость с учетом склеивания на границах распространяется на объединение (3.3) [131, 132].

Поскольку $1 \le n$, в предположении $h < 1$, для любого i из (3.4) верно неравенство

$$|f(x,\tilde{y}(x)) - \Psi_{rin}(t)| \le c\, 2^{-2k}\, h^2,\ \forall i = \overline{0,\, 2^k - 1} \wedge \forall x \in [x_{ri},\, x_{r\,i+1}], \qquad (3.10)$$

где правая часть не зависит от степени полинома n.

Согласно (3.9), (3.10) равномерная сходимость сохраняется в условиях рассматриваемых вариаций n, в границах которых она распространяется на объединение (3.3), а в силу (3.10) сохраняется в условиях двукратной дифференцируемости правой части (3.1).

Для оценки приближения $\tilde{y}(x)$ вводится обозначение $P_{ri\,n+1}(x) = P^{(1)}_{i\,n+1}(x),\ \ x \in [x_{ri},\, x_{ri+1}]$. Для рассматриваемых x выполнено условие

$$\tilde{y}(x) - P_{ri\,n+1}(x) = \tilde{y}(x_{ri}) + \int_{x_{ri}}^{x} f(x,\tilde{y}(x))\,dx - \left(\tilde{y}(x_{ri}) + \int_{x_{ri}}^{x} \Psi_{rin}(t)\,dx \right),$$

отсюда получаем

$$\left| \tilde{y}(x) - P_{ri\,n+1}(x) \right| \le \int_{x_{ri}}^{x} |f(x,\tilde{y}(x)) - \Psi_{rin}(t)|\,dx,\ x \in [x_{ri},\, x_{ri+1}].$$

С учетом (3.9) имеем

$$\left| \tilde{y}(x) - P_{ri\,n+1}(x) \right| \le c\, 2^{-k\,(n+1)}\, h^{n+1} (x - x_{ri}) \quad \forall x \in [x_{ri},\, x_{ri+1}], \qquad (3.11)$$

тем более,

$$\left| \tilde{y}(x) - P_{ri\,n+1}(x) \right| \le c\,(\beta_r - \alpha_r) 2^{-k\,(n+2)}\, h^{n+1} \quad \forall i = \overline{0,\, 2^k - 1} \ \wedge\ \forall x \in [x_{ri},\, x_{ri+1}]. \quad (3.12)$$

Ввиду $1 \le n \le n_0$, в предположении $h < 1$, имеем

$$\left| \tilde{y}(x) - P_{ri\,n+1}(x) \right| \le c\,(\beta_r - \alpha_r)\, 2^{-3k}\, h^2 \quad \forall i = \overline{0,\, 2^k - 1} \ \wedge\ \forall x \in [x_{ri},\, x_{ri+1}]. \quad (3.13)$$

Из (3.12), (3.13) следует равномерная сходимость $P_{ri\,n+1}(x)$ к $\tilde{y}(x)$ на $[\alpha_r, \beta_r]$, утверждение и оценки распространяются на $[x_0, x_{fin}]$ в силу $\beta_r - \alpha_r = \text{const}$.

Рассматриваемые условия заведомо обеспечивают сходимость метода Эйлера на всем промежутке решения. Погрешность ε_{h_e} этого метода на шаге $h_e = 2^{-k}$ оценивается как $\varepsilon_{h_e} = \widetilde{c}(h_e)^2 = \widetilde{c}2^{-2k}$, $\widetilde{c} = \text{const}$ [117]. Можно выбрать k достаточно большим, чтобы при $h < 1$ выполнялось $c(\beta_r - \alpha_r)h^2 2^{-k} < \widetilde{c}$. Тогда на $[x_0, x_{fin}]$ будет выполнено условие $c(\beta_r - \alpha_r)h^2 2^{-3k} < \widetilde{c}2^{-2k}$. Аналогично, при произвольном h с учетом $1 \le n \le n_0$ за счет выбора k можно обеспечить выполнение неравенства $c(\beta_r - \alpha_r)2^{-k(n+2)} h^{n+1} < \widetilde{c}2^{-2k}$. Начальное значение задачи (3.1) для рассматриваемого метода и метода Эйлера одинаково. Начиная с некоторого k, первый метод на шаге отклоняется от точного решения меньше второго, поэтому оба сходятся к одному и тому же решению.Таким образом,

$P_{ri\,n+1}(x) \rightrightarrows y(x)$.

Вследствие накопления погрешности скорость сходимости к $y(x)$ будет отличаться от (3.12), (3.13). Именно, погрешность на $[x_0, x_{fin}]$ складывается из погрешностей на отрезках (3.4) и на отрезках (3.3), поэтому не превзойдет их суммы в объединениях (3.3), (3.4):

$$\left| y(x) - P_{ri\,n+1}(x) \right| \le \sum_{r=0}^{M-1} \sum_{i=0}^{2^k-1} c(\beta_r - \alpha_r)2^{-k(n+2)} h^{n+1}$$
$$\forall r = \overline{0, M-1} \wedge \forall i = \overline{0, 2^k - 1} \wedge \forall x \in [x_{ri}, x_{r\,i+1}].$$

Поскольку $\sum_{r=0}^{M-1} \sum_{i=0}^{2^k-1} c(\beta_r - \alpha_r) = M 2^k c(\beta_r - \alpha_r)$, то

$$\left| y(x) - P_{ri\,n+1}(x) \right| \le c(x_{fin} - x_0)2^{-k(n+1)} h^{n+1}$$
$$\forall r = \overline{0, M-1} \wedge \forall i = \overline{0, 2^k - 1} \wedge \forall x \in [x_{ri}, x_{r\,i+1}]. \qquad (3.14)$$

Равномерная относительно n оценка получится с учетом $1 \le n$ при выборе $h < 1$:

$$\left| y(x) - P_{ri\,n+1}(x) \right| \le c\,(x_{fin} - x_0)\,2^{-2k}\,h^2$$
$$\forall\, r = \overline{0, M-1} \wedge \forall\, i = \overline{0,\, 2^k - 1} \wedge \forall\, x \in [\,x_{ri}\,,\, x_{r\,i+1}\,]. \qquad (3.15)$$

Таким образом, в рассматриваемых условиях имеет место

Теорема 3.1. При любом n, $1 \le n \le n_0$, $n_0 = \text{const}$, последовательность полиномов $P_{ri\,n+1}(x)$, где i из (3.4), r из (3.3), равномерно на $[x_0, x_{fin}]$ сходится к точному решению $y(x)$ задачи (3.1) при $k \to \infty$. Скорость сходимости оценивается из (3.14), где c из (3.6), h – расстояние между узлами полинома $\Psi_{r0n}(t)$, интерполирующего правую часть (3.1) на $[\alpha_r, \beta_r]$. Значение h не зависит от r и не меняется с ростом k при разбиении $[\alpha_r, \beta_r]$ на 2^k отрезков. При $h < 1$ скорость сходимости оценивается из (3.15).

В частности, теорема верна в случае двукратной непрерывной дифференцируемости правой части (3.1).

Следствие 3.1. В условиях теоремы последовательность интерполяционных полиномов $\Psi_{rin}(t)$ равномерно на $[x_0, x_{fin}]$ приближает производную $y'(x)$ при $k \to \infty$, при этом

$$| f(x, y(x)) - \Psi_{rin}(t) | \le (2L+1)\, c\, (x_{fin} - x_0)\, 2^{-k\,(n+1)}\, h^{n+1}$$
$$\forall\, r = \overline{0, M-1} \wedge \forall\, i = \overline{0,\, 2^k - 1} \wedge \forall\, x \in [\,x_{ri}\,,\, x_{r\,i+1}\,],$$

и, если $h < 1$, то

$$| f(x, y(x)) - \Psi_{rin}(t) | \le (2L+1)\, c\, (x_{fin} - x_0)\, 2^{-2k}\, h^2$$
$$\forall\, r = \overline{0, M-1} \wedge \forall\, i = \overline{0,\, 2^k - 1} \wedge \forall\, x \in [\,x_{ri}\,,\, x_{r\,i+1}\,].$$

В самом деле, из (3.12) имеем $| \tilde{y}(x) - P_{ri\,n+1}(x) | \le c\,(x_{fin} - x_0)\, 2^{-k\,(n+2)}\, h^{n+1}$, $\forall\, x \in [x_{ri}\,, x_{ri+1}]$. Отсюда и из (3.14) имеем

$$\left| y(x) - \tilde{y}(x) \right| \le 2c\,(x_{fin} - x_0)\, 2^{-k\,(n+1)}\, h^{n+1}\ \forall\, x \in [\,x_{ri}\,,\, x_{ri+1}\,].$$

Далее, согласно (3.9) имеем

$$| f(x,\tilde{y}(x)) - \Psi_{rin}(t) | \le c(x_{fin} - x_0)\, 2^{-k(n+1)}\, h^{n+1} \;\; \forall x \in [x_{ri}, x_{ri+1}].$$

Кроме того, поскольку

$$| f(x,y(x)) - \Psi_{rin}(t) | \le | f(x,y(x)) - f(x,\tilde{y}(x)) | + | f(x,\tilde{y}(x)) - \Psi_{rin}(t) |,$$

то с учетом (3.2) получится оценка

$$| f(x,y(x)) - \Psi_{rin}(t) | \le L\, | y(x) - \tilde{y}(x) | + | f(x,\tilde{y}(x)) - \Psi_{rin}(t) |.$$

Подстановка в правую часть и объединение данных неравенств по всем отрезкам из (3.3), (3.4), аналогично, с учетом (3.15) влечет утверждение следствия.

Замечание 3.1. На практике вариации $n, 1 \le n \le n_0$, и k, $1 \le k \le k_0$, реализуются по минимуму невязки так, чтобы дробь $2^{-k(n+1)}$ оказалась наименьшей за счет произведения $k(n+1)$ в данных границах, а не за счет лишь $k \to \infty$.

3.2. Скорость сходимости метода. Вначале на отрезке $[\alpha_r, \beta_r]$ выполняются вариации, на основе которых степень полиномов n, а также количество составляющих отрезков 2^k выбираются и фиксируются. Затем на каждом отрезке $[x_{ri}, x_{ri+1}]$, $i = 0, 1, \ldots, 2^k - 1$, выполняется итерационное уточнение кусочно-интерполяционного решения задачи (3.1). Именно, $y(x) \approx P^{(0)}_{ri\,n+1}(x)$, где $P^{(0)}_{ri\,n+1}(x) = P_{ri\,n+1}(x)$, $x \in [x_{ri}, x_{ri+1}]$, и $f(x,y) \approx f(x, P^{(0)}_{ri\,n+1}(x))$. Далее, вычисляется

$$\Psi^{(0)}_{rin}(t) \approx f(x, P^{(0)}_{ri\,n+1}(x)),\; t = (x - x_{ri}) h^{-1} 2^k, \tag{3.16}$$

строится приближение:

$$\bar{y}(x_{ri}) = P^{(q)}_{r\,i-1\,n+1}(x_{rin}),\; P^{(1)}_{ri\,n+1}(x) = \bar{y}(x_{ri}) + \int_{x_{ri}}^{x} \Psi^{(0)}_{rin}(t)\, dx,\; y(x) \approx P^{(1)}_{ri\,n+1}(x),$$

где q – конечный номер итераций на отрезке $[x_{ri-1}, x_{ri}]$, затем

$$\Psi^{(1)}_{rin}(t) \approx f(x, P^{(1)}_{ri\,n+1}(x)),\ P^{(2)}_{ri\,n+1}(x) = \bar{y}(x_{ri}) + \int_{x_{ri}}^{x} \Psi^{(1)}_{rin}(t)\,dx,\ y(x) \approx P^{(2)}_{ri\,n+1}(x).$$

В продолжение процесса, при $\ell \geq 2$, имеем

$$\begin{aligned} &\Psi^{(\ell)}_{rin}(t) \approx f(x, P^{(\ell)}_{ri\,n+1}(x)),\ \ P^{(\ell+1)}_{ri\,n+1}(x) = \bar{y}(x_{ri}) + \int_{x_{ri}}^{x} \Psi^{(\ell)}_{rin}(t)\,dx, \\ &y(x) \approx P^{(\ell+1)}_{ri\,n+1}(x),\ \ x \in [x_{ri},\ x_{ri+1}]. \end{aligned} \tag{3.17}$$

Обозначение $\bar{y}(x)$ вводится для отличия от $\tilde{y}(x)$, относительно которого итерационное уточнение не предполагалось.

В простейшем варианте реализации итерации (3.16), (3.17) продолжаются до некоторой априори фиксированной границы $\ell = q$ (другие варианты обсуждаются при описании численных экспериментов). Абстрактно допускается $\ell \to \infty$.

Возмущенное от накопления погрешности по предыдущим отрезкам начальное значение $\bar{y}(x_{ri})$ на левой границе i-го отрезка входит в выражение аналогично возмущенного решения задачи (3.1) в виде:

$$\bar{y}(x) = \bar{y}(x_{ri}) + \int_{x_{ri}}^{x} f(x, \bar{y})\,dx,\ x \in [x_{ri},\ x_{ri+1}].$$

Итерационное уточнение (3.17) относится именно к этому решению с тем же начальным значением на данном подынтервале:

$$P^{(\ell+1)}_{ri\,n+1}(x) = \bar{y}(x_{ri}) + \int_{x_{ri}}^{x} \Psi^{(\ell)}_{rin}(t)\,dx,$$

где t из (3.16). Разность между этими приближениями определяется равенством

$$\bar{y}(x) - P^{(\ell+1)}_{ri\,n+1}(x) = \int_{x_{ri}}^{x} (f(x, \bar{y}) - \Psi^{(\ell)}_{rin}(t))\,dx,\ x \in [x_{ri},\ x_{ri+1}],\ \ell = 0,1,\ldots.$$

Отсюда имеем

$$
\begin{aligned}
&|\,\bar{y}(x)-P^{(\ell+1)}_{ri\,n+1}(x)\,|\le\\
&\le\int\limits_{x_{ri}}^{x}\big(\,|\,f(x,\bar{y})-f(x,P^{(\ell)}_{ri\,n+1}(x))\,|+|\,f(x,P^{(\ell)}_{ri\,n+1}(x))-\Psi^{(\ell)}_{rin}(t)\,|\,\big)dx,
\end{aligned}
\tag{3.18}
$$

где $x\in[x_{ri},\,x_{ri+1}],\ \ell=0,1,\ldots$ Из (3.2) и (3.18) следует неравенство

$$
\begin{aligned}
&|\,\bar{y}(x)-P^{(\ell+1)}_{ri\,n+1}(x)\,|\le\\
&\le\int\limits_{x_{ri}}^{x}\big(\,L\,|\,\bar{y}(x)-P^{(\ell)}_{ri\,n+1}(x)\,|+|\,f(x,P^{(\ell)}_{ri\,n+1}(x))-\Psi^{(\ell)}_{rin}(t)\,|\,\big)dx.
\end{aligned}
\tag{3.19}
$$

В соответствии с (3.19) возможны две разновидности оценки итерационного уточнения. Именно, погрешность интерполирования правой части (3.1) на итерации с номером ℓ оценивается из (3.9) при $\widetilde{y}=P^{(\ell)}_{ri\,n+1}(x)$, $\Psi_{rin}(t)=\Psi^{(\ell)}_{rin}(t)$:

$$
\begin{aligned}
&|\,f(x,P^{(\ell)}_{rin}(x))-\Psi^{(\ell)}_{rin}(t)\,|\le c_{kr},\ \ c_{kr}=c\,2^{-k\,(n+1)}\,h^{\,n+1},\\
&\forall\, i=\overline{0,\,2^{k}-1}\wedge\forall\, x\in[x_{ri},\,x_{r\,i+1}].
\end{aligned}
\tag{3.20}
$$

Если на $[x_{ri},\,x_{r\,i+1}]$ выполнено соотношение

$$
\exists\,\ell\wedge\exists\,x\in[x_{ri},\,x_{ri+1}]:\ |\,\bar{y}(x)-P^{(\ell)}_{ri\,n+1}(x)\,|<|\,f(x,P^{(\ell)}_{ri\,n+1}(x))-\Psi^{(\ell)}_{rin}(t)\,|,
\tag{3.21}
$$

то из (3.19) следует завышенная оценка:

$$
|\,\bar{y}(x)-P^{(\ell+1)}_{ri\,n+1}(x)\,|\le\int\limits_{x_{ri}}^{x}\big(L\,|\,\bar{y}(x)-P^{(\ell)}_{ri\,n+1}(x)\,|+c_{kr}\big)\,dx,\,x\in[x_{ri},\,x_{ri+1}],
\tag{3.22}
$$

где c_{kr} из (3.20). Пусть обозначено $\varepsilon_{i\ell}=|\,\bar{y}(x)-P^{(\ell)}_{ri\,n+1}(x)\,|,\ \ \ell=0,1,\ldots,$ $x\in[x_{ri},\,x_{ri+1}]$. Согласно (3.22)

$$
\varepsilon_{i\ell+1}\le\int\limits_{x_{ri}}^{x}(L\varepsilon_{i\ell}+c_{kr})\,dx\,.
\tag{3.23}
$$

При $\ell = 0$ имеем $\varepsilon_{i1} \le \int\limits_{x_{ri}}^{x} \left(L \,|\, \bar{y}(x) - P^{(0)}_{ri\,n+1}(x) \,| + c_{kr} \right) dx$. Согласно (3.11)

$\left| \bar{y}(x) - P^{(0)}_{ri\,n+1}(x) \right| \le c_{kr}(x - x_{ri})$. Отсюда

$$\varepsilon_{i1} \le \int\limits_{x_{ri}}^{x} (L\, c_{kr}(x - x_{ri}) + c_{kr})\, dx \;\text{ или }\; \varepsilon_{i1} \le c_{kr}\left(L\frac{(x - x_{ri})^2}{2} + x - x_{ri} \right).$$

При $\ell = 1$ подстановка в (3.23) влечет

$$\varepsilon_{i2} \le c_{kr} \int\limits_{x_{ri}}^{x} \left(L\left(L\frac{(x - x_{ri})^2}{2} + x - x_{ri} \right) + 1 \right) dx$$

или $\varepsilon_{i2} \le c_{kr}\left(L^2 \frac{(x - x_{ri})^3}{6} + L\frac{(x - x_{ri})^2}{2} + x - x_{ri} \right)$. По индукции

$$\varepsilon_{i\ell+1} \le c_{kr}\left(\frac{L^{\ell+1}(x - x_{ri})^{\ell+2}}{(\ell+2)!} + \frac{L^{\ell}(x - x_{ri})^{\ell+1}}{(\ell+1)!} + \cdots + \frac{L(x - x_{ri})^2}{2} + (x - x_{ri}) \right).$$

Тем более, $\varepsilon_{i\ell+1} \le c_{kr}\, L^{-1}(e^{L(x - x_{ri})} - 1)$. Отсюда следует, что $\varepsilon_{i\ell+1} \le c_{kr} L^{-1}(e^{L(x_{ri+1} - x_{ri})} - 1)$. Показатель степени можно считать столь малым, что $e^{L(x_{ri+1} - x_{ri})} - 1 \sim L(x_{ri+1} - x_{ri})$, в результате $\varepsilon_{i\ell+1} \le c_{kr}(x_{ri+1} - x_{ri})$. С учетом $x_{ri+1} - x_{ri} = (\beta_r - \alpha_r) 2^{-k}$ имеем

$$| \bar{y}(x) - P^{(\ell+1)}_{ri\,n+1}(x) | \le c_{kr}(\beta_r - \alpha_r) 2^{-k} \quad \forall x \in [x_{ri}, x_{ri+1}]. \tag{3.24}$$

Правая часть неравенства (3.24) совпадает с правой частью (3.12). Таким образом, в предположении (3.21) итерационное уточнение итерационное уточнение кусочно-интерполяционного приближения не гарантирует улучшение приближения. Завышенная оценка не исключает возможность уточнения, однако из (3.24) не следует, что оно необходимо достигается.

Если же на $[x_{ri}, x_{r\,i+1}]$ выполнено соотношение

$$| f(x, P^{(\ell)}_{ri\,n+1}(x)) - \Psi^{(\ell)}_{rin}(t) | \le | \bar{y}(x) - P^{(\ell)}_{ri\,n+1}(x) |, \quad \ell = 0, 1, ..., \quad \forall x \in [x_{ri}, x_{ri+1}]. \tag{3.25}$$

итерационное уточнение на рассматриваемом отрезке существенно улучшает кусочно-интерполяционное приближение не только к решению задачи (3.1), но и к его производной. В самом деле, согласно (3.25) верно также неравенство

$$| f(x, P_{ri\,n+1}^{(\ell+1)}(x)) - \Psi_{rin}^{(\ell+1)}(t) | \le | \bar{y}(x) - P_{ri\,n+1}^{(\ell+1)}(x) |, \ \ell = 0,1, \ldots, \ \forall x \in [x_{ri}, x_{ri+1}]. \quad (3.26)$$

Из (3.19) и (3.25) получаем

$$| \bar{y}(x) - P_{ri\,n+1}^{(\ell+1)}(x) | \le \int_{x_{ri}}^{x} (L+1) \, | \bar{y}(x) - P_{ri\,n+1}^{(\ell)}(x) | dx. \quad (3.27)$$

С учетом (3.26) имеем

$$| f(x, P_{ri\,n+1}^{(\ell+1)}(x)) - \Psi_{rin}^{(\ell+1)}(t) | \le \int_{x_{ri}}^{x} (L+1) \, | \bar{y}(x) - P_{ri\,n+1}^{(\ell)}(x) | dx.$$

Отсюда и из (3.27) для $\forall x \in [x_{ri}, x_{ri+1}]$ выполнено

$$\max\left(| f(x, P_{ri\,n+1}^{(\ell+1)}(x)) - \Psi_{rin}^{(\ell+1)}(t) |, | \bar{y}(x) - P_{ri\,n+1}^{(\ell+1)}(x) | \right) \le \int_{x_{ri}}^{x} (L+1) | \bar{y}(x) - P_{ri\,n+1}^{(\ell)}(x) | dx,$$

тем более,

$$\begin{aligned} &\max\left(| f(x, P_{ri\,n+1}^{(\ell+1)}(x)) - \Psi_{rin}^{(\ell+1)}(t) |, | \bar{y}(x) - P_{ri\,n+1}^{(\ell+1)}(x) | \right) \le \\ &\le \int_{x_{ri}}^{x} (L+1) \max\left(| f(x, P_{ri\,n+1}^{(\ell)}(x)) - \Psi_{rin}^{(\ell)}(t) |, \ | \bar{y}(x) - P_{ri\,n+1}^{(\ell)}(x) | \right) dx. \end{aligned} \quad (3.28)$$

В обозначении $\tilde{\varepsilon}_{i\ell} = \max\left(| f(x, P_{ri\,n+1}^{(\ell)}(x)) - \Psi_{rin}^{(\ell)}(t) |, \ | \bar{y}(x) - P_{ri\,n+1}^{(\ell)}(x) | \right)$ неравенство (3.28) примет вид:

$$\tilde{\varepsilon}_{i\ell+1} \le \int_{x_{ri}}^{x} (L+1) \, \tilde{\varepsilon}_{i\ell} \, dx, \ \ell = 0,1, \ldots \ \forall x \in [x_{ri}, x_{ri+1}]. \quad (3.29)$$

При $\ell = 0$: $\widetilde{\varepsilon}_{i1} \le \int_{x_{ri}}^{x} (L+1)\widetilde{\varepsilon}_{i0}\, dx$. С учетом (3.16) и (3.11), $\widetilde{\varepsilon}_{i1} \le \int_{x_{ri}}^{x} (L+1) c_{kr}\, dx$, или

$\widetilde{\varepsilon}_{i1} \le (L+1) c_{kr}(x - x_{ri})$. Отсюда и из (3.29) $\widetilde{\varepsilon}_{i2} \le \int_{x_{ri}}^{x} (L+1)^2 c_{kr}(x - x_{ri})\, dx$, так что

$\widetilde{\varepsilon}_{i2} \le (L+1)^2 c_{kr} \dfrac{(x - x_{ri})^2}{2}$. По индукции $\widetilde{\varepsilon}_{i\ell+1} \le \dfrac{(L+1)^{\ell+1}(x - x_{ri})^{\ell+1}}{(\ell+1)!} c_{kr}$.

Следовательно $\widetilde{\varepsilon}_{i\ell+1} \le \dfrac{(L+1)^{\ell+1}(x_{ri+1} - x_{ri})^{\ell+1}}{(\ell+1)!} c_{kr}$. Окончательно получаем

$$\widetilde{\varepsilon}_{i\ell+1} \le \frac{((L+1)(\beta_r - \alpha_r))^{\ell+1}}{2^{k(\ell+1)}(\ell+1)!} c_{kr} \quad \forall x \in [x_{ri}, x_{ri+1}], \tag{3.30}$$

где c_{kr} из (3.20). Согласно (3.30) граница c_{kr} погрешности кусочно-интерполяционного приближения производной из (3.9) на рассматриваемом отрезке $[x_{ri}, x_{ri+1}]$ при умножении снижается пропорционально $\dfrac{((L+1)(\beta_r - \alpha_r))^{\ell+1}}{2^{k(\ell+1)}(\ell+1)!} \to 0$ при $\ell \to \infty$. Одновременно с тем граница $c(\beta_r - \alpha_r)\, 2^{-k(n+2)} h^{n+1}$ погрешности приближения решения из (3.12) на этом же отрезке уменьшается пропорционально $\dfrac{((L+1)(\beta_r - \alpha_r))^{\ell+1}}{2^{k\ell}(\ell+1)!} \to 0$ при $\ell \to \infty$. Из изложенного вытекает

Теорема 3.2. В условиях теоремы 3.1 на отрезке $[x_{ri}, x_{ri+1}]$, где выполнено (3.25), итерационное уточнение (3.16), (3.17) снижает абсолютную погрешность кусочно-интерполяционных приближений решения и его производной пропорционально множителю, не превосходящему $\dfrac{((L+1)(\beta_r - \alpha_r))^{\ell}}{2^{k\ell}\,\ell!} \to 0$ при $\ell \to \infty$, где L – константа Липшица, $[\alpha_r, \beta_r]$– отрезок произвольно задаваемой длины из (3.3), (3.4). Если на $[x_{ri}, x_{ri+1}]$ выполнено (3.21), снижение абсолютной погрешности приближения решения

вследствие итерационного уточнения возможно, однако из (3.24) не следует, что оно необходимо достигнется.

В случае невысокого порядка гладкости (3.1) значение c_{kr} в (3.20) и (3.25) можно взять при $n=1$ и $h<1$, сохранив проделанные рассуждения.

Следствие 3.2. Утверждение теоремы сохраняется в случае $n=1$, $c_{kr}=c\,2^{-2k}\,h^2$ и $h<1$.

Замечание 3.2. Если условие (3.25) выполняется на множестве подряд расположенных отрезков из (3.3), (3.4), то оценка погрешности может быть просуммирована аналогично тому, как при выводе (3.14), при этом сумма сохранит коэффициент пропорции из теоремы 3.2. В этом случае накопление погрешности кусочно-интерполяционного метода с итерационным уточнением будет стремиться к нулю на всем множестве этих промежутков за счет процесса итерационного уточнения при фиксированных n и k.

Как показывают численные эксперименты, для нежестких ОДУ с малой границей производных именно если длина $[x_{ri}, x_{ri+1}]$ больше единицы, при соответственно большой длине $[\alpha_r, \beta_r]$ и большом количестве подынтервалов 2^k, происходит существенное итерационное уточнение приближения. Для жестких систем итерационное уточнение существенно, если $(x_{ri+1}-x_{ri})<<1$.

Для компьютерной реализации принципиальна следующая особенность процесса (3.16), (3.17). Начальное значение на левой границе смежных отрезков $\bar{y}(x_{ri})$ для (3.17) заимствуется с правой границы предыдущего отрезка по окончании на нем процесса итерационного уточнения: $\bar{y}(x_{ri})=P^{(q)}_{r\,i-1\,n+1}(x_{rin})$. Результат интерполяции может скорректировать приближение в сторону невозмущенного решения, в этом случае итерационное уточнение будет ограничивать накопление погрешности на текущем отрезке. Как следствие в значении погрешности при сравнении с точным решением возникает множество нулей (пример 3.1, ниже). Иногда они заполняют целиком отрезок

$[\alpha_r, \beta_r]$, иногда его продолжение, в том числе после ненулевых значений погрешности. Это создает предпосылку «динамической» коррекции «начальных» значений с соответственным уточнением приближенного решения.

Замечание 3.3. Кусочно-интерполяционное приближение решения на каждом отрезке $[x_{ri}, x_{ri+1}]$ имеет вид алгебраического полинома с числовыми значениями коэффициентов $P_{ri\,n+1}(x) = \sum_{\ell=0}^{n+1} a_{\ell\,ri\,n+1}\, x^{\ell}$, его производная, аналогично, $\psi_{ri\,n}(x) = \sum_{\ell=0}^{n} b_{\ell\,ri\,n}\, x^{\ell}$, итерационное уточнение сохраняет их алгебраический вид на каждом отрезке. Приближение решения и производной можно восстановить по соответствию индексов коэффициентов индексам отрезка. С учетом равных узловых значений на границах смежных отрезков метод по соответствию i, n, k, ℓ позволяет воспроизвести на $[\alpha_r, \beta_r]$ аналитический вид непрерывного приближения решения и его производной.

В главе 2 (п. 2.1.3) показано, что в условиях $(n+1)$-кратной непрерывной дифференцируемости функции производные от интерполирующих ее полиномов сходятся к ее производной. В рамках метода можно получить табличные производные высших порядков от интерполирующих полиномов в случае «вложенного» выполнения условий существования и сходимости.

3.3. Верификация достоверности с помощью численного и программного эксперимента. Невязка, в отличие метода, описанного в главе 2, и от работ [131, 132], выбирается на основе приближения к модулю разности между точным решением и первообразными от аппроксимирующих правую часть (3.1) полиномов на отрезках с общими границами, их длина фиксируется как многократно меньшая $\beta_r - \alpha_r$. На отрезке $[x_j, x]$ рассматриваемое приближение можно представить в виде

$$y(x)-y(x_j)-(P^{(\ell+1)}_{r\tilde{i}\,n+1}(x)-P^{(\ell+1)}_{ri\,n+1}(x_j))\approx\int_{x_j}^{x}(f(x,y)-f(x,P^{(\ell)}_{ri\,n+1}(x)))\,dx,\qquad(3.31)$$

где $\tilde{i}$ и i – переменные значения индекса отрезка из (3.4), включающего x. Определяется разность приближений к (3.31) на текущем и предшествующем количестве отрезков из (3.4):

$$\begin{aligned}&\Big|\int_{x_j}^{x}f(x,y)\,dx-\int_{x_j}^{x}f(x,P^{(\ell)}_{ri\,n+1}(x))\,dx\Big|\approx\Delta_M(r,k,n,\ell),\\&\Delta_M(r,k,n,\ell)=\big|\,(P^{(\ell)}_{kri_1\,n+1}(x)-P^{(\ell)}_{kri_2\,n+1}(x_j))-(P^{(\ell)}_{(k-1)ri_3\,n+1}(x)-P^{(\ell)}_{(k-1)ri_4\,n+1}(x_j))\,\big|,\end{aligned}\qquad(3.32)$$

где M из (3.3); i_1, i_2 – индексы соответственно конечного и начального отрезков, которым принадлежат x и x_j в случае 2^k отрезков (3.4). По определению, $P^{(\ell)}_{kri_1\,n+1}(x)=P^{(\ell)}_{ri_1\,n+1}(x)$ для того же количества отрезков, аналогично, – для i_2 в соответствии 2^k отрезков, также по определению $P^{(\ell)}_{(k-1)ri_3\,n+1}(x)=P^{(\ell)}_{ri_3\,n+1}(x)$ в случае 2^{k-1} отрезков из (3.4), аналогично, – для i_4. Для 2^{k-1} отрезков i_3, i_4 – номера конечного и начального отрезков, включающих, соответственно, x и x_j на $[\alpha_r,\beta_r]$.

Для начальной пары фиксированных n и k на объединении (3.4) находится и запоминается наибольшее $\Delta_M(r,k,n,\ell)$, обозначаемое $\nabla_M(r,k,n,\ell)$. Затем поиск $\nabla_M(r,k,n,\ell)$ возобновляется при том же n и возрастании k на единицу в пределах $\overline{k=1,k_0}$, затем при увеличении n на единицу от начального k, $n=\overline{1,n_0}$, и т.д. Среди таких максимумов определяется минимальный по всем разбиениям на отрезки при всех вариациях степени:

$$\nabla_M(r,k,n,\ell;\min)=\min_{\substack{k,\,n\\ 1\le n\le n_0;\ 1\le k\le k_0}}\nabla_M(r,k,n,\ell);$$
$$\nabla_M(r,k,n,\ell)=\max_{\substack{\bigcup_{[x_j,\,x]=[\alpha_r,\,\beta_r]}\\ k=\text{const},\ n=\text{const}}}\Delta_M(r,k,n,\ell). \qquad (3.33)$$

В соответствии со значением $\nabla_M(r,k,n,\ell;\min)$ фиксируется то число отрезков 2^k и та степень полинома n на каждом из них в границах $[\alpha_r,\beta_r]$, при которых достигается этот минимум. В программной реализации минимум (3.33) определяется с итерационным уточнением полиномов в (3.32) при некотором $\ell=\text{const}$, в дальнейшем ℓ_{start}. Только после этого, причем заново, для соответственно зафиксированных k и n выполняется окончательное итерационное уточнение приближенного решения согласно (3.16), (3.17) с новым $\ell=\text{const}$, в дальнейшем ℓ_{fin}.

Погрешность минимизируется в результате данной вариации k, n, ℓ.

Наибольшая точность и устойчивость метода достигается со следующей модификацией (3.32). Строится сумма значений $\Delta_M(r,k,n,\ell)$ из (3.32) по равномерному разбиению $[x_j,x]$, которая нормируется:

$$\widetilde{\Delta}_M(r,k,n,\ell)= \\ =\frac{\left|\sum\limits_{q=0}^{\omega-1}\left|P^{(\ell)}_{kri_1n+1}(x_j+(q+1)\tau)-P^{(\ell)}_{kri_2n+1}(x_j+q\tau)\right|-\sum\limits_{q=0}^{\omega-1}\left|P^{(\ell)}_{(k-1)ri_3n+1}(x_j+(q+1)\tau)-P^{(\ell)}_{(k-1)ri_4n+1}(x_j+q\tau)\right|\right|}{\varepsilon+\sum\limits_{q=0}^{\omega-1}\left|P^{(\ell)}_{kri_1n+1}(x_j+(q+1)\tau)-P^{(\ell)}_{kri_2n+1}(x_j+q\tau)\right|}. \qquad (3.34)$$

Здесь ω – постоянное целое, выбранное экспериментально: $\omega=5$; $\omega\tau$ – фиксированная часть длины $[\alpha_r,\beta_r]$, в случае $1<<\beta_r-\alpha_r$, для нежестких задач экспериментально выбранная как $\omega\tau=\sqrt[q]{\beta_r-\alpha_r}$, $q=2,4$. В случае жестких задач $\beta_r-\alpha_r<1$ и $\omega\tau=(\beta_r-\alpha_r)/p$, $p\ge 2$. Знаменатель дроби (3.34) нормирует числовое значение невязки; $\varepsilon>0$ – малое число для исключения деления на ноль. Относительно $\widetilde{\Delta}_M(r,k,n,\ell)$ выполняются те же действия, что относительно $\Delta_M(r,k,n,\ell)$, с целью выбора $\widetilde{\nabla}_M(r,k,n,\ell;\min)$, определяемого

аналогично $\nabla_M(r, k, n, \ell; \min)$ из (3.33). С невязкой (3.34) метод устойчив при произвольном выборе длины отрезка $[\alpha_r, \beta_r]$. Однако согласно эксперименту наилучшему приближению в случае нежесткой задачи в зависимости от ее вида отвечают $\beta_r - \alpha_r = 512$ или $\beta_r - \alpha_r = 256$, а также $\beta_r - \alpha_r = 128$. Выбор $\omega\tau$, аналогично, определяется наибольшей точностью приближения. Границы вариации n и k могут быть заданы произвольно с сохранением высокой точности. Практически достаточно взять $n \le 12$ и $k \le 12$, время решения при этом в целом не выше, чем у разностных методов. Наибольшая точность достигается в границах $n \le 14$ и $k \le 15$ с замедлением решения. Ниже даны примеры численного эксперимента с динамической коррекцией начальных значений.

Пример 3.1. Задача $y' = x - y$, $y(0)=1$, имеет аналитическое решение $y = x - 1 + 2e^{-x}$. В листинге 3.1 представлен программный код кусочно-интерполяционного решения данной задачи с итерационным уточнением на основе $\widetilde{\Delta}_M(r, k, n, \ell)$ из (3.34) на отрезке $[0, 512]$.

Листинг 3.1

```
Program Varying_Piecewise_Interpolation; {$APPTYPE CONSOLE}
uses Math, SysUtils; const Anach=0; Bkonech=512;
var K_iter,output,K_iter_itog:integer;
Nmin,Nmax,Kmin,Kmax:byte; Vel_int,otr_int,Ynach :extended;
 function f(x,y: extended):extended; begin f:=x-y {cos(x+y)} end;
function fun(x:extended):extended;
begin fun:=x-1+2*exp(-x) end;
procedure RD(Ynach,Vel_int:extended; otr_int:extended);
const nn=20; type matr=array[0..nn,0..nn] of extended;
 vect=array[0..nn] of extended; matrC=array[-5..nn+1] of extended;
 matrAll=array[0..33000] of matrC; matrC_=^matrAll;
var pod,pod1,pod2:longint; i,j,int:integer;
 x,y0,max,min,a0,b0,a00,b00,s,Xfinal,h,MinGL:extended;
 Ck,Ck1,Ck2,Ck1_,Ck1_opt,Ck_,Ck_opt:matrC_; d:matr; Min_b0,s_l1,s_r1:extended;
n,k,n_,k_,n_opt,k_opt:byte;
//Расчет коэффициентов полинома
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr;
begin e[1,1]:=1; e[1,0]:=0;
for k:=2 to n do begin e[k,0]:=-e[k-1,0]*(k-1);
 for i:=1 to k-1 do e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1);
 e[k,k]:=e[k-1,k-1] end;
for k:=1 to n do for i:=0 to k do d[i,k]:=e[k,i] end;
//вычисление конечных разностей
procedure Konech_Raznoct(fy:vect; n:byte; var dy:matr); var i,j:byte;
begin for j:=0 to n-1 do dy[1,j]:=fy[j+1]-fy[j];
 for i:=2 to n do for j:=0 to n-i do dy[i,j]:=dy[i-1,j+1]-dy[i-1,j] end;
```

```
//построение полинома Ньютона в алгебраической форме по узловым значениям
procedure Newton(U:Vect; n:byte; var Mcoef:matrC);
var dy:matr; b:vect; p,s:extended; j,i:byte;
begin Konech_Raznoct(U,n,dy); p:=1;
for j:=1 to n do begin p:=p*j; b[j]:=dy[j,0]/p; end; Mcoef[0]:=U[0];
for i:=1 to n do begin s:=0; for j:=i to n do s:=s+d[i,j]*b[j];
 Mcoef[i]:=s; end end;
//вычисление значения полинома по схеме Горнера
function Gorner(Mcoef:matrC; x:extended):extended;
 var i,n:byte; s,t:extended;
 begin  t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
  for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
//построение приближения решения на текщем подынтервале,
// выполнение итерационного уточнения
procedure Polynomial(x:vect; h,y0:extended; n,k,K_it:integer; var C:matrC);
 var i,iter:integer; sum:extended; ytemp,fy,y:vect; A:matrC;
 begin  for i:=0 to n do y[i]:=y0;for iter:=1 to K_it do
 begin for i:=1 to n do ytemp[i]:=y[i]; for i:=0 to n do fy[i]:=f(x[i],y[i]);
 Newton(fy,n,A); C[0]:=y[0]; C[-1]:=x[0]; C[-2]:=h; C[-3]:=n+1;C[-4]:=k;
C[-5]:=n*h;  for  i:=1  to  n+1  do  C[i]:=A[i-1]*h/i;  for  i:=1  to  n  do
y[i]:=Gorner(C,x[i]); sum:=0; for i:=1 to n do sum:=sum+abs(y[i]-ytemp[i]);
if (sum<1e-35) or (sum>=MaxDouble) then break; end end;
//построение приближения на текущем отрезке
procedure Subinterval(k,n,K_it:integer; a0,b0,Ynach:extended; var Ck:matrC_);
var hpd,a00,b00,y0,h:extended; m,pod:longint; x:vect; j:byte;
begin hpd:=(b0-a0)/exp(k*ln(2)); a00:=a0; b00:=a00+hpd; y0:=Ynach;
 x[0]:=a0; m:=0; pod:=0;
while a00<=b0-hpd/2 do begin h:=(b00-a00)/n;
 for j:=1 to n do begin inc(m); x[j]:=a0+m*h end;
 Polynomial(x,h,y0,n,k,K_it,Ck^[pod]); y0:=Gorner(Ck^[pod],x[n]);
 x[0]:=x[n]; inc(pod);a00:=a00+hpd; b00:=a00+hpd    end end;
//вычисление интегральной невязки на отрезке
function Integral(a0,b0,a00,b00:extended; Ck:matrC_):extended;
var pod1,pod2,m:longint; x1,x2,h,ss:extended;
begin h:=(b00-a00)/5; m:=0;x1:=a00;x2:=a00+h;ss:=0;
repeat pod1:=trunc((x1-a0)/Ck^[0,-5]); if abs(x2-b0)<1e-15 then
pod2:=trunc(exp(Ck^[0,-4]*ln(2)))-1 else pod2:=trunc((x2-a0)/Ck^[0,-5]);
ss:=ss+abs(Gorner(Ck^[pod2],x2)-Gorner(Ck^[pod1],x1));
inc(m); x1:=a00+m*h; x2:=x1+h;
until x2>b00+h/2;
Integral:=ss; end;
function S_min(x1:extended):extended; var pd,pd1:longint;
begin pd:= trunc((x1-a0)/ Ck_opt^[0,-5]); pd1:= trunc((x1-a0)/Ck1_opt^[0,-5]);
 S_min:=abs(Gorner(Ck_opt^[pd],x1)-Gorner(Ck1_opt^[pd1],x1))/
  (1e-8+abs(Gorner(Ck_opt^[pd],x1))); end;
begin
Viet(nn,d); New(Ck); New(Ck1); New(Ck2); New(Ck_); New(Ck_opt);
New(Ck1_); New(Ck1_opt); a0:=Anach; b0:=Anach+Vel_int;
while a0 < Bkonech do begin
inc(int);MinGL:=MaxDouble; n:=Nmin; repeat k:=Kmin; Min:=MaxDouble;
 repeat Subinterval(k,n,K_iter,a0,b0,Ynach,Ck1);
Subinterval(k+1,n,K_iter,a0,b0,Ynach,Ck2);
max:=0; a00:=a0; b00:=a00+otr_int; i:=0;
while b00<=b0 do begin
 s:=abs(Integral(a0,b0,a00,b00,Ck2)-Integral(a0,b0,a00,b00,Ck1))/(1e-7
+abs(Integral(a0,b0,a00,b00,Ck2)));
if s>max then max:=s; inc(i);
a00:=a0+i*otr_int; b00:=a00+otr_int; end;
if max<Min then begin Min:=max; Ck_^:=Ck2^;Ck1_^:=Ck1^ end;
inc(k); until k>Kmax;
if Min<MinGL then begin MinGL:=Min;Ck_opt^:=Ck_^;Ck1_opt^:=Ck1_^ end;inc(n);
 until n>Nmax;
```

```
N:=trunc(Ck_opt^[0,-3])-1;k:=trunc(Ck_opt^[0,-4]);
  Subinterval(k,n,K_iter_itog,a0,b0,Ynach,Ck_opt);
//Поиск минимума для динамической коррекции начальных данных
pod:=trunc(exp(trunc(Ck_opt^[0,-4])*ln(2)))-1;
pod1:=trunc(exp(trunc(Ck1_opt^[0,-4])*ln(2)))-1;
Min_b0:=abs(Gorner(Ck_opt^[pod],b0)-Gorner(Ck1_opt^[pod1],b0))/(1e-
7+abs(Gorner(Ck_opt^[pod],b0))); XFinal:=b0;
h:=Vel_int/100; x:=a0+20*h; Min:=maxExtended; i:=21;
repeat s_l1:=S_min(x-h); s_r1:=S_min(x+h); s:=S_min(x);
if (s<=s_l1) and (s<=s_r1) and (s<=Min_b0) and (s<Min) then
 begin XFinal:=x; Min:=s; end; inc(i); x:=a0+i*h; until x> b0-2*h;
   for i:=0 to output-1 do //вывод приближения
    begin x:=a0+i*Vel_int / output; pod:= trunc((x-a0) / Ck_opt^[0,-5]);
if (x>=Xfinal) or ((x-Bkonech)>1e-16) then break;writeln(x:7:3,' ',
 Gorner(Ck_opt^[pod],x), ', ', abs(Gorner(Ck_opt^[pod],x)-fun(x))); end;
if abs(Xfinal-b0) < 1e-16 then pod:= trunc(exp(trunc(Ck_opt^[0,-4])*ln(2)))-1
   else pod:=trunc((Xfinal-a0)/Ck_opt^[0,-5]);
 Ynach:=Gorner(Ck_opt^[pod],Xfinal); a0:=Xfinal; b0:= a0 + Vel_int;
end; Dispose(Ck2); Dispose(Ck); Dispose(Ck1); Dispose(Ck_); Dispose(Ck1_opt);
Dispose(Ck_opt); Dispose(Ck1_); end;
begin {блок инструкций}
Ynach:=1{0}; {начальное условие} K_iter:=30;{число начальных итераций ℓ_start }
K_iter_itog:=300;{число завершающих итераций ℓ_fin }
Vel_int:=512{128};{длина отрезка [α_r, β_r] } output:=100; {число точек вывода}
Nmin:=1; Nmax:=11{12};{диапазон вариации степени полинома}
Kmin:=0; Kmax:=10;{диапазон вариации k (число подынтервалов 2^k)}
otr_int:=sqrt(vel_int); {длина отрезка интегрирования}
RD(Ynach,Vel_int,otr_int); readln; end.
```

Результат работы программы *Varying_Piecewise_Interpolation* представлен в столбце *VPI* табл. 3.1. В 100 равномерно распределенных проверочных точках на отрезке [0,512] располагаются значения абсолютной погрешности предложенного метода. Погрешность в этих точках равна «нулю» в формате вывода *extended* (при выводе свыше 10000 значений на том же отрезке в диапазоне $0<x<50$ выявляются погрешности порядка $10^{-21}-10^{-18}$).

Таблица 3.1

Абсолютная погрешность приближения решения задачи из примера 3.1

x	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*VPI*
	$h=10^{-7}$	$h=10^{-6}$	$h=10^{-4}$	$h=10^{-2}$	$h=10^{-2}$	
5.12	3.060**e-09**	1.023**e-14**	4.337**e-18**	4.120**e-17**	3.036**e-18**	0.000**e+00**
10.24	4.657**e-11**	5.378**e-17**	1.475**e-17**	2.602**e-18**	8.674**e-19**	0.000**e+00**
…	…	…	…	…	…	…
256.00	4.063**e-11**	5.939**e-12**	4.075**e-14**	1.624**e-15**	3.886**e-16**	0.000**e+00**
261.12	1.100**e-10**	1.006**e-11**	2.864**e-14**	3.358**e-15**	6.939**e-16**	0.000**e+00**
…	…	…	…	…	…	…
506.88	1.100**e-10**	1.288**e-11**	2.864**e-14**	3.386**e-15**	6.661**e-16**	0.000**e+00**
512.00	1.100**e-10**	1.288**e-11**	2.864**e-14**	3.358**e-15**	6.939**e-16**	0.000**e+00**

В тех же проверочных точках представлена абсолютная погрешность решения данной задачи по методу Эйлера (*Euler*), Эйлера-Коши (*Euler-Cauchy*), Рунге-Кутты 4-го порядка (*Runge-Kutta_4*), Бутчера (*Butcher_6*), Дормана-Принса (*Dormand-Prince_8*). Во второй строке таблицы – шаг разностного метода, уменьшение которого в формате *extended* на практике не снижает погрешность. Разностные методы (с учетом более плотного вывода) уступают в точности не менее 2 десятичных порядков. Если для сравнения брать не только точки худшего приближения, сравнительная малость погрешности метода наглядна на множестве всех точек приближенного решения или в среднем. Метод классифицируется как аналитический (замечание 3.3), дает непрерывное и непрерывно дифференцируемое приближение решения.

Пример 3.2. Задача $y' = \cos(x+y)$, $y(0)=0$, имеет решение $y = -x + 2\,arctg\,x$. Закомментированная часть программы, представленной в листинге 3.1, соответствует приближенному решению этой задачи на $[0, 512]$. Программа выводит значения абсолютной погрешности, данные в столбце *VPI*.

Таблица 3.2

Абсолютная погрешность приближения решения задачи из примера 3.2

x	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*VPI*
	$h=10^{-7}$	$h=10^{-6}$	$h=10^{-4}$	$h=10^{-3}$	$h=10^{-3}$	
5.12	1.214**e-08**	3.143**e-14**	1.084**e-18**	4.120**e-18**	5.855**e-18**	0.000**e+00**
10.24	4.404**e-09**	8.854**e-15**	4.337**e-18**	6.072**e-18**	4.337**e-18**	8.674**e-19**
…	…	…	…	…	…	…
250.88	1.755**e-11**	5.038**e-15**	1.749**e-15**	2.623**e-15**	1.069**e-15**	1.388**e-17**
256.00	1.692**e-11**	6.106**e-15**	4.260**e-15**	2.817**e-15**	7.078**e-16**	0.000**e+00**
261.12	1.631**e-11**	6.606**e-15**	8.188**e-15**	1.804**e-15**	1.943**e-16**	0.000**e+00**
…	…	…	…	…	…	…
506.88	4.675**e-12**	4.718**e-15**	3.914**e-15**	5.190**e-15**	0.000**e+00**	0.000**e+00**
512.00	4.624**e-12**	4.108**e-15**	1.110**e-16**	6.883**e-15**	4.718**e-16**	0.000**e+00**

Погрешности разностных методов расположены в прежнем соответствии иуступают в точности до 2 и более десятичных порядков.

Замечание 3.4. В сравнении с результатами решения задачи (3.1) разностно-полиномиальным методом удалось значительно расширить область

высокоточного приближенного решения и повысить устойчивость метода к изменению параметров. В частности, решение задачи из примера 3.2 на основе варьируемого разностно-полиномиального метода приближалось с высокой точностью на интервале $x \in [0, 10]$ при значениях параметров: $[\alpha_r, \beta_r] = 1$, число итераций – 6, $4 \le n \le 8$, $6 \le k \le 9$ (см. листинг 2.2, табл. 2.1). Кусочно-интерполяционное решение этой же задачи получено с высокой точностью на промежутке [0, 512] при значении параметров: $[\alpha_r, \beta_r] = 128$, число начальных итераций $\ell_{start} = 30$, число завершающих итераций $\ell_{fin} = 300$, $1 \le n \le 12$, $0 \le k \le 10$ (листинг 3.1, табл. 3.2). В целом такое увеличение промежутка высокоточного решения и повышение устойчивости метода к изменению параметров наблюдается и при решении других задач вида (3.1) [76, 145].

3.4. Приближенное решение дифференциальной системы для моделей обработки данных периодических процессов. Рассматривается задача

$$Y' = F(x, Y),\ Y(x_0) = Y_0, \tag{3.35}$$

где $F(x,Y) = (f_1(x,Y), f_2(x,Y), \ldots, f_N(x,Y))$, $Y = (y_1(x), y_2(x), \ldots, y_N(x))$, $Y_0 = (y_{01}, y_{02}, \ldots, y_{0N})$. Используется каноническая норма вектора $\| Y \| = \max\limits_{1 \le j \le N} | y_j |$.

Предполагается, что в области

$$R_0 : \{ x_0 \le x \le x_{fin};\ \| Y - Y_0 \| \le b;\ \ x_{fin} = \text{const}, b = \text{const} \}$$

выполнены все условия существования и единственности, функция $F(x,Y)$ определена, непрерывна и непрерывно дифференцируема (в точке x_0 – справа и в точке x_{fin} – слева), удовлетворяет условию Липшица:

$$\| F(x,Y) - F(x,\widetilde{Y}) \| \le \widetilde{L} \| Y - \widetilde{Y} \|,\quad \widetilde{L} = \text{const},\quad \forall (x, Y), (x, \widetilde{Y}) \in R_0. \tag{3.36}$$

Кусочно-полиномиальное приближение и итерационное уточнение строятся для каждого уравнения системы (3.35) в полной аналогии случаю одной переменной. Сложность может состоять во взаимодействии компонентов

приближения, которые могут соответствовать различным степеням полиномов и различным количествам отрезков. Здесь и ниже приближения всех компонентов на шаге рассматриваются при одних и тех же значениях ℓ, k, n одновременно во всех уравнениях системы (3.35). Если правая часть (3.35) $(n+1)$-кратно непрерывно дифференцируема в R_0, то при каждом $j=\overline{1,N}$ неравенство (3.5) переносится на j-е уравнение системы в виде

$$|f_j(x,\widetilde{Y})-\Psi_{jrin}(t)| \le \frac{\max\limits_{\xi_j\in[\alpha_r,\beta_r]} |f_j^{(n+1)}(\xi_j,\widetilde{Y}(\xi_j)|\,h^{n+1}}{(n+1)!}\left|\prod_{\ell=0}^{n}(t-\ell)\right|,\ x\in[x_{r0},x_{r1}],$$

где $\Psi_{jrin}(t)$ – интерполяционный полином Ньютона в алгебраической форме с числовыми коэффициентами, построенный для аппроксимации $f_j(x,\widetilde{Y})$, $h=(x_{rn}-x_{r0})/n$, $t=(x-x_{r0})/h$. Отсюда с учетом

$$C=\max_{1\le n\le n_0}\ \max_{R_0}\ \max_{1\le j\le N}\ \max_{\xi_j\in[x_0,x_{fin}]} |f_j^{(n+1)}(\xi_j,\widetilde{Y}(\xi_j)|,\ C=\text{const}, \tag{3.37}$$

следует неравенство

$$\left\| F(x,\widetilde{Y})-\{\Psi_{jrin}(t)\}_{j=1}^{N}\right\| \le \frac{C\,h^{n+1}}{(n+1)!}\left|\prod_{\ell=0}^{n}(t-\ell)\right|,\quad x\in[x_{r0},x_{r1}]. \tag{3.38}$$

Оценка (3.9) переходит в соотношение

$$\begin{aligned} &P=2^k,\ [\alpha_r,\beta_r]=\bigcup_{i=0}^{2^k-1}[x_{ri},x_{r\,i+1}],\\ &\left\| F(x,\widetilde{Y})-\{\Psi_{jrin}(t)\}_{j=1}^{N}\right\| \le C\,2^{-k(n+1)}\,h^{n+1}\quad \forall i=\overline{0,2^k-1}\wedge\forall x\in[x_{ri},x_{r\,i+1}], \end{aligned} \tag{3.39}$$

где C из (3.37), h – шаг интерполирования полинома $\Psi_{jrin}(t)$ на $[\alpha_r,\beta_r]$ при $k=0$, одинаковый для всех $k=1,2,\ldots$ и для всех $j=\overline{1,N}$. Оценка (3.10) перейдет в соотношение вида:

$$\left\| F(x,\widetilde{Y})-\{\Psi_{jrin}(t)\}_{j=1}^{N}\right\| \le C\,2^{-2k}\,h^2\quad \forall i=\overline{0,2^k-1}\wedge\forall x\in[x_{ri},x_{r\,i+1}]. \tag{3.40}$$

На основе (3.38) – (3.40) получится аналог неравенств (3.11) – (3.13):

$$\begin{aligned}&\left\| \widetilde{Y}(x)-\{P_{jri\,n+1}(x)\}_{j=1}^{N} \right\| \le C\ 2^{-k(n+1)}\ h^{n+1}(x-x_{ri}) \le \\ &\le C\ (\beta_r-\alpha_r)\ 2^{-k(n+2)}\ h^{n+1} \qquad \forall i=\overline{0,2^k-1} \wedge \forall x\in[x_{ri}\,,\,x_{ri+1}],\end{aligned} \tag{3.41}$$

где $P_{jri\,n+1}(x)$ – первообразная от $\Psi_{jrin}(t)$, приближающая j-й компонент решения. При ограничении $n \le n_0$, $n_0 = \text{const}$, в предположении $h<1$, получится

$$\left\| \widetilde{Y}(x)-\{P_{jri\,n+1}(x)\}_{j=1}^{N} \right\| \le C\ (\beta_r-\alpha_r) 2^{-3k}\ h^2,\ \forall x\in[x_{ri}\,,\,x_{ri+1}]\ . \tag{3.42}$$

Из (3.41), (3.42) следует равномерная сходимость $\{P_{jri\,n+1}(x)\}_{j=1}^{N}$ к $\widetilde{Y}(x)$. Погрешность приближения к точному решению $Y(x)$ оценится из неравенств

$$\begin{aligned}&\left\| Y(x)-\{P_{jri\,n+1}(x)\}_{j=1}^{N} \right\| \le \sum_{r=0}^{M-1}\sum_{i=0}^{2^k-1} C\ (\beta_r-\alpha_r) 2^{-k(n+2)}\ h^{n+1} \\ &\forall r=\overline{0,M-1} \wedge \forall i=\overline{0,2^k-1} \wedge \forall x\in[x_{ri}\,,\,x_{ri+1}],\end{aligned}$$

согласно которым получатся аналоги в норме оценок (3.14), (3.15):

$$\begin{aligned}&\left\| Y(x)-\{P_{jri\,n+1}(x)\}_{j=1}^{N} \right\| \le C\ (x_{fin}-x_0) 2^{-k(n+1)}\ h^{n+1} \\ &\forall r=\overline{0,M-1} \wedge \forall i=\overline{0,\,2^k-1} \wedge \forall\, x\in[x_{ri}\,,\,x_{r\,i+1}]\end{aligned}$$

и, при $h<1$,

$$\begin{aligned}&\left\| Y(x)-\{P_{jri\,n+1}(x)\}_{j=1}^{N} \right\| \le C\ (x_{fin}-x_0)\ 2^{-2k}\ h^2 \\ &\forall r=\overline{0,M-1} \wedge \forall i=\overline{0,2^k-1} \wedge \forall\, x\in[x_{ri}\,,\,x_{r\,i+1}].\end{aligned}$$

Отсюда следует равномерная сходимость $\{P_{jri\,n+1}(x)\}_{j=1}^{N}$ к $Y(x)$ на $[x_0, x_{fin}]$, с данными оценками формулируется N-мерный аналог теоремы 3.1 для системы (3.35).

Аналогично следствию 3.1 доказывается, что последовательность $\{\Psi_{jrin}(t)\}_{j=1}^{N}$ равномернона $[x_0, x_{fin}]$ приближает точную производную $Y'(x)$ при $k\to\infty$ с оценками:

$$\left\| F(x,Y) - \{\Psi_{jrin}(t)\}_{j=1}^{N} \right\| \le$$
$$\le (2\widetilde{L}+1) C (x_{fin} - x_0) 2^{-k(n+1)} h^{n+1} \quad \forall r = \overline{0, M-1} \wedge \forall i = \overline{0, 2^k - 1} \wedge \forall x \in [x_{ri}, x_{r\,i+1}],$$

и, при $h < 1$,

$$\left\| F(x,Y) - \{\Psi_{jrin}(t)\}_{j=1}^{N} \right\| \le$$
$$\le (2\widetilde{L}+1) C (x_{fin} - x_0) 2^{-2k} h^{2} \quad \forall r = \overline{0, M-1} \wedge \forall i = \overline{0, 2^k - 1} \wedge \forall x \in [x_{ri}, x_{r\,i+1}].$$

3.5. Итерационное уточнение данных. Как и в одномерном случае, вначале на отрезке $[\alpha_r, \beta_r]$ из (3.3) при помощи вариаций выбирается и фиксируется степень полиномов n и количество отрезков 2^k из (3.4). По окончании вариаций, последовательно по $i = 0, 1, \ldots, 2^k - 1$, на каждом отрезке $[x_{ri}, x_{ri+1}]$, выполняется итерационное уточнение. Именно,

$$Y(x) \approx \left\{ P^{(0)}_{jri\,n+1}(x) \right\}_{j=1}^{N}, \ \left\{ P^{(0)}_{jri\,n+1}(x) \right\}_{j=1}^{N} = \left\{ P_{jri\,n+1}(x) \right\}_{j=1}^{N}, \ F(x,Y) \approx F\left(x, \left\{ P^{(0)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right),$$

$$\left\{ \Psi^{(0)}_{jri\,n}(t) \right\}_{j=1}^{N} \approx F\left(x, \left\{ P^{(0)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right), \ t = (x - x_{ri}) h^{-1} 2^k,$$

$$\overline{Y}(x_{ri}) = \left\{ P^{(q)}_{jri-1\,n+1}(x_{rin}) \right\}_{j=1}^{N}, \ \left\{ P^{(1)}_{jri\,n+1}(x) \right\}_{j=1}^{N} = \overline{Y}(x_{ri}) + \int_{x_{ri}}^{x} \left\{ \Psi^{(0)}_{jrin}(t) \right\}_{j=1}^{N} dx,$$

$$Y(x) \approx \left\{ P^{(1)}_{jri\,n+1}(x) \right\}_{j=1}^{N},$$

где q – конечный номер итераций на предыдущем отрезке $[x_{ri-1}, x_{ri}]$. Далее,

$$\left\{ \Psi^{(1)}_{jrin}(t) \right\}_{j=1}^{N} \approx F\left(x, \left\{ P^{(1)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right), \ \left\{ P^{(2)}_{jri\,n+1}(x) \right\}_{j=1}^{N} = \overline{Y}(x_{ri}) + \int_{x_{ri}}^{x} \left\{ \Psi^{(1)}_{jrin}(t) \right\}_{j=1}^{N} dx,$$

$$Y(x) \approx \left\{ P^{(2)}_{jri\,n+1}(x) \right\}_{j=1}^{N}, \ \left\{ \Psi^{(\ell)}_{jrin}(t) \right\}_{j=1}^{N} \approx F\left(x, \left\{ P^{(\ell)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right),$$

$$\left\{ P^{(\ell+1)}_{jri\,n+1}(x) \right\}_{j=1}^{N} = \overline{Y}(x_{ri}) + \int_{x_{ri}}^{x} \left\{ \Psi^{(\ell)}_{jrin}(t) \right\}_{j=1}^{N} dx, \ Y(x) \approx \left\{ P^{(\ell+1)}_{jri\,n+1}(x) \right\}_{j=1}^{N}.$$

По аналогии с одномерным случаем с учетом (3.36) получится неравенство

$$\left\| \overline{Y}(x) - \left\{ P^{(\ell+1)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| \le$$
$$\le \int_{x_{ri}}^{x} \left(\widetilde{L} \left\| \overline{Y}(x) - \left\{ P^{(\ell)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| + \left\| F\left(x, \left\{ P^{(\ell)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right) - \left\{ \Psi^{(\ell)}_{jrin}(t) \right\}_{j=1}^{N} \right\| \right) dx, \tag{3.43}$$

относительно которого на $[x_{ri}, x_{ri+1}]$ рассматриваются два предположения. Первое имеет вид

$$\exists \ell \wedge \exists\, x \in [x_{ri}, x_{ri+1}]:$$
$$\left\| \overline{Y}(x) - \left\{ P^{(\ell)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| < \left\| F\left(x, \left\{ P^{(\ell)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right) - \left\{ \Psi^{(\ell)}_{jrin}(t) \right\}_{j=1}^{N} \right\|. \tag{3.44}$$

На отрезок, где выполнено (3.44), переносятся оценки (3.22) – (3.24):

$$\left\| \overline{Y}(x) - \left\{ P^{(\ell+1)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| \le \int_{x_{ri}}^{x} \left(\widetilde{L} \left\| \overline{Y}(x) - \left\{ P^{(\ell)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| + C_{kr} \right) dx,$$

где $C_{kr} = C\, 2^{-k(n+1)} h^{n+1}$ из (3.39). В обозначении $\varepsilon_{i\ell} = \left\| \overline{Y}(x) - \left\{ P^{(\ell)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\|$ получим: $\varepsilon_{i\ell+1} \le \int_{x_{ri}}^{x} (\widetilde{L}\varepsilon_{i\ell} + C_{kr})\, dx$, $\ell = 0,1,\ldots$, $x \in [x_{ri}, x_{ri+1}]$. При $\ell = 0$ имеем

$$\varepsilon_{i1} \le \int_{x_{ri}}^{x} \left(\widetilde{L} \left\| \overline{Y}(x) - \left\{ P^{(0)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| + C_{kr} \right) dx\ .$$

Согласно (3.44) и (3.39) имеем $\left\| \overline{Y}(x) - \left\{ P^{(0)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| \le C_{kr}(x - x_{ri})$, отсюда

$$\varepsilon_{i1} \le \int_{x_{ri}}^{x} (\widetilde{L} C_{kr}(x - x_{ri}) + C_{kr})\, dx \ \text{ или } \ \varepsilon_{i1} \le \widetilde{L} C_{kr} \frac{(x - x_{ri})^2}{2} + C_{kr}(x - x_{ri}).$$

Как и при выводе (3.24),

$$\varepsilon_{i\ell+1} \le C_{kr}\, \widetilde{L}^{-1}(e^{\widetilde{L}(x_{ri+1} - x_{ri})} - 1);$$
$$\left\| \overline{Y}(x) - \left\{ P^{(\ell+1)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| \le C_{kr}(\beta_r - \alpha_r) 2^{-k} \quad \forall\, x \in [x_{ri}, x_{r\,i+1}]. \tag{3.45}$$

В данном случае итерационное уточнение, возможно, понижает погрешность кусочной интерполяции, однако, (3.45) не означает, что это выполняется с необходимостью.

Альтернативное предположение относительно $[x_{ri}, x_{r\,i+1}]$ и (3.44) состоит в том, что

$$\left\| F\left(x, \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N}\right) - \left\{\Psi^{(\ell)}_{jrin}(t)\right\}_{j=1}^{N} \right\| \le \left\| \overline{Y}(x) - \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N} \right\|, \quad \ell = 0,1,\ldots, \forall x \in [x_{ri}, x_{ri+1}]. \tag{3.46}$$

Неравенство (3.46) сохранится при замене ℓ на $\ell+1$. Отсюда и из (3.43) имеем

$$\left\| \overline{Y}(x) - \left\{P^{(\ell+1)}_{jri\,n+1}(x)\right\}_{j=1}^{N} \right\| \le \int_{x_{ri}}^{x} (\widetilde{L}+1) \left\| \overline{Y}(x) - \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N} \right\| dx\,,$$

$$\left\| F\left(x, \left\{P^{(\ell+1)}_{jri\,n+1}(x)\right\}_{j=1}^{N}\right) - \left\{\Psi^{(\ell+1)}_{jrin}(t)\right\}_{j=1}^{N} \right\| \le \int_{x_{ri}}^{x} (\widetilde{L}+1) \left\| \overline{Y}(x) - \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N} \right\| dx\,.$$

По аналогии с выводом (3.28) получится

$$\max\left(\left\| F\left(x, \left\{P^{(\ell+1)}_{jri\,n+1}(x)\right\}_{j=1}^{N}\right) - \left\{\Psi^{(\ell+1)}_{jrin}(t)\right\}_{j=1}^{N} \right\|, \left\| \overline{Y}(x) - \left\{P^{(\ell+1)}_{jri\,n+1}(x)\right\}_{j=1}^{N} \right\| \right) \le$$
$$\le \int_{x_{ri}}^{x} (\widetilde{L}+1) \max\left(\left\| F\left(x, \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N}\right) - \left\{\Psi^{(\ell)}_{jrin}(t)\right\}_{j=1}^{N} \right\|, \left\| \overline{Y}(x) - \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N} \right\| \right) dx. \tag{3.47}$$

В обозначении

$$\widetilde{\varepsilon}_{i\ell} = \max\left(\left\| F\left(x, \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N}\right) - \left\{\Psi^{(\ell)}_{jrin}(t)\right\}_{j=1}^{N} \right\|, \left\| \overline{Y}(x) - \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N} \right\| \right)$$

неравенство (3.47) примет вид: $\widetilde{\varepsilon}_{i\,\ell+1} \le \int_{x_{ri}}^{x} (\widetilde{L}+1)\,\widetilde{\varepsilon}_{i\ell}\,dx$, $\ell = 0,1,\ldots$. При $\ell = 0$:

$\widetilde{\varepsilon}_{i1} \le \int_{x_{ri}}^{x} (\widetilde{L}+1)\,\widetilde{\varepsilon}_{i0}\,dx$, и, с учетом (3.39) и (3.44),

$\widetilde{\varepsilon}_{i1} \le \int_{x_{ri}}^{x} (\widetilde{L}+1) C_{kr}\, dx = (\widetilde{L}+1) C_{kr} (x - x_{ri})$. Как и в одномерном случае имеем

$$\widetilde{\varepsilon}_{i\ell+1} \le \frac{((\widetilde{L}+1)(\beta_r - \alpha_r))^{\ell+1}}{2^{k(\ell+1)}(\ell+1)!} C_{kr} \text{ или}$$

$$\max\left(\left\| F(x,\overline{Y}) - \left\{ \Psi^{(\ell+1)}_{jrin}(t) \right\}_{j=1}^{N} \right\|, \left\| \overline{Y}(x) - \left\{ P^{(\ell+1)}_{jri\,n+1}(x) \right\}_{j=1}^{N} \right\| \right) \le$$
$$\le \frac{((\widetilde{L}+1)(\beta_r - \alpha_r))^{\ell+1}}{2^{k(\ell+1)}(\ell+1)!} C_{kr}. \qquad (3.48)$$

Согласно (3.48) граница $C_{kr} = C\, 2^{-k(n+1)} h^{n+1}$ погрешности кусочно-интерполяционного приближения производной из (3.39) на отрезке $[x_{ri}, x_{ri+1}]$, где выполнено (3.46), уменьшается пропорционально $\frac{((\widetilde{L}+1)(\beta_r - \alpha_r))^{\ell+1}}{2^{k(\ell+1)}(\ell+1)!} \to 0$ при $\ell \to \infty$. Одновременно с тем граница $C(\beta_r - \alpha_r) 2^{-k(n+2)} h^{n+1}$ погрешности приближения решения из (3.41) на этом отрезке уменьшается пропорционально $\frac{((\widetilde{L}+1)(\beta_r - \alpha_r))^{\ell+1}}{2^{k\ell}(\ell+1)!} \to 0$ при $\ell \to \infty$.

С оценками (3.45), (3.48) формулируются аналоги теоремы 3.2, следствия 3.2 и последующих замечаний для задачи (3.35).

3.6. Численный эксперимент для моделей обработки данных в случае дифференциальной системы. Всюду ниже сохраняется структура невязки (3.34), выбор $[\alpha_r, \beta_r]$ и других параметров дан по ходу изложения.

Пример 3.3. Рассматривается задача Коши (2.47), имеющая неустойчивое в смысле Ляпунова решение с соответственно выраженным накоплением погрешности приближенного решения. Программа, реализующая предложенный метод, представлена в листинге 3.2.

Листинг 3.2

```
program VPI_S2; {$APPTYPE CONSOLE}
uses Math, SysUtils; const Anach=1; Bkonech=513;MaxExtended:extended =1.18E4932;
var K_iter,K_iter_itog,output:integer; Nmin,Nmax,Kmin,Kmax:byte;
Vel_int,Ynach1,Ynach2,otrezok_integr:extended; Method_Iter,Nev:byte;res:text;
```

```
function f1(x,y1,y2: extended):extended; begin  f1:=x+(2*y1)/x-sqrt(y2) end;
function f2(x,y1,y2: extended):extended; begin  f2:=2*sqrt(y2) end;
function fun1(x:extended):extended; begin fun1:=x*(1+x) end;
function fun2(x:extended):extended; begin fun2:=(x+1)*(x+1) end;
procedure RD(Ynach1,Ynach2,Vel_int:extended;K_iter,Nmin,Nmax,
Kmin,Kmax,output:integer;otrezok_integr:extended);
const nn=30;type matr=array[0..nn,0..nn] of extended;
 vect=array[0..nn] of extended; matrC=array[-5..nn+1] of extended;
 matrAll=array[0..33000] of matrC; matrC_=^matrAll;
var pod,pod1,pod2:longint; n,k,n_,k_,n_opt,k_opt:byte; i,j,int:integer;
x,y01,y02,a0,b0,a00,b00,max,min,MinGL,s,Min0,Xfinal,Xfinal1,
Xfinal2,h,s1,s2:extended; Ck1,Ck11,Ck12,Ck1_,Ck1_opt,Ck11_,Ck11_opt:matrC_;
Ck2,Ck21,Ck22,Ck2_,Ck2_opt,Ck21_,Ck21_opt:matrC_; d:matr;
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr;
begin e[1,1]:=1; e[1,0]:=0; for k:=2 to n dobegin e[k,0]:=-e[k-1,0]*(k-1);
for i:=1 to k-1 do e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1);
e[k,k]:=e[k-1,k-1] end;
for k:=1 to n do for i:=0 to k do d[i,k]:=e[k,i] end;
procedure Konech_Raznoct(fy:vect; n:byte; var dy:matr);
var i,j:byte; begin for j:=0 to n-1 do dy[1,j]:=fy[j+1]-fy[j];
 for i:=2 to n do for j:=0 to n-i do dy[i,j]:=dy[i-1,j+1]-dy[i-1,j] end;
procedure Newton(U:Vect; n:byte; var Mcoef:matrC);
var dy:matr; b:vect; p,s:extended; j,i:byte;
begin Konech_Raznoct(U,n,dy); p:=1;
 for j:=1 to n do begin p:=p*j; b[j]:=dy[j,0]/p; end; Mcoef[0]:=U[0];
 for i:=1 to n do begin s:=0;
for j:=i to n do s:=s+d[i,j]*b[j]; Mcoef[i]:=s; end end;
function Gorner(Mcoef:matrC; x:extended):extended;
var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
 for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
procedure Polynomial(x:vect; h,y01,y02:extended; n,k,K_iter:integer;
var C1,C2:matrC);
var i,iter:integer; sum1,sum2,sum:extended;
ytemp1,ytemp2,fy1,y1,fy2,y2:vect; A1,A2:matrC;
begin for i:=0 to n do y1[i]:=y01; for i:=0 to n do y2[i]:=y02;
 for iter:=1 to K_iter do begin for i:=1 to n do begin
 ytemp1[i]:=y1[i]; ytemp2[i]:=y2[i]; end;
 if Method_Iter=1 then begin  //Аналог метода простой итерации
  for i:=0 to n do
   begin fy1[i]:=f1(x[i],y1[i],y2[i]); fy2[i]:=f2(x[i],y1[i],y2[i]) end;
  Newton(fy1,n,A1); Newton(fy2,n,A2);
  C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;
  C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h; C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h;
  for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i; C2[i]:=A2[i-1]*h/i; end;
  for i:=1 to n do begin y1[i]:=Gorner(C1,x[i]); y2[i]:=Gorner(C2,x[i]); end;
 end else // Аналог метода Зейделя
  begin for i:=0 to n do fy1[i]:=f1(x[i],y1[i],y2[i]); Newton(fy1,n,A1);
  C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;
  for i:=1 to n+1 do C1[i]:=A1[i-1]*h/i;
  for i:=1 to n do y1[i]:=Gorner(C1,x[i]);
  for i:=0 to n do fy2[i]:=f2(x[i],y1[i],y2[i]); Newton(fy2,n,A2);
  C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h; C2[-3]:=n+1;C2[-4]:=k; C2[-5]:=n*h;
  for i:=1 to n+1 do C2[i]:=A2[i-1]*h/i;
  for i:=1 to n do y2[i]:=Gorner(C2,x[i]); end;
 sum1:=0; sum2:=0;
 for i:=1 to n do begin sum1: =sum1+abs(y1[i]-ytemp1[i]);
 sum2: =sum2+abs(y2[i]-ytemp2[i]); end;
 if sum1 > sum2 then sum:=sum1 else sum:=sum2;
 if (sum<1e-25) or (sum>=1e45) then break; end end;
procedure Subinterval(k,n,K_iter:integer; a0,b0,Ynach1,Ynach2:extended;
var Ck1,Ck2:matrC_);
```

```
var hpd,a00,b00,y01,y02,h:extended;m,pod:longint; x:vect; j:byte;
begin hpd:=(b0-a0)/exp(k*ln(2)); a00:=a0; b00:=a00+hpd; y01:=Ynach1;y02:=Ynach2;
 x[0]:=a0; m:=0; pod:=0;
 while a00<=b0-hpd/2 do begin h:=(b00-a00)/n;
  for j:=1 to n do begin inc(m); x[j]:=a0+m*h end;
  Polynomial(x,h,y01,y02,n,k,K_iter,Ck1^[pod],Ck2^[pod]);
  y01:=Gorner(Ck1^[pod],x[n]);y02:=Gorner(Ck2^[pod],x[n]); x[0]:=x[n]; inc(pod);
  a00:=a00+hpd; b00:=a00+hpd end end;
function Integral(a0,b0,a00,b00:extended; Ck:matrC_):extended;
var pod1,pod2,m:longint; x1,x2,h,ss:extended;
begin h:=(b00-a00)/5; m:=0;x1:=a00; x2:=a00+h; ss:=0;
 repeat pod1:=trunc((x1-a0)/Ck^[0,-5]);
  if abs(x2-b0)<1e-15 then pod2:=trunc(exp(Ck^[0,-4]*ln(2)))-1
else pod2:=trunc((x2-a0)/Ck^[0,-5]);
  ss:=ss+abs(Gorner(Ck^[pod2],x2)-Gorner(Ck^[pod1],x1));
inc(m); x1:=a00+m*h;x2:=x1+h; until x2>b00+h/2;
Integral:=ss; end;
begin
Viet(nn,d); writeln('x':4,'y':15,'Pogr':25); writeln;
New(Ck1); New(Ck11);New(Ck12);New(Ck1_);New(Ck1_opt);New(Ck11_);New(Ck11_opt);
New(Ck2); New(Ck21); New(Ck22); New(Ck2_);New(Ck2_opt);New(Ck21_);New(Ck21_opt);
a0:=Anach; b0:=Anach+Vel_int;
 while a0 < Bkonech do begin inc(int);MinGL:=MaxExtended;n:=Nmin;
  repeat Min:= MaxExtended; k:=Kmin;
   repeat Subinterval(k,n,K_iter,a0,b0,Ynach1,Ynach2,Ck11,Ck21);
//приближение для текущего значения k
    Subinterval(k+1,n,K_iter,a0,b0,Ynach1,Ynach2,Ck12,Ck22);
//приближение для значения k+1
max:=0; a00:=a0; b00:=a00+otrezok_integr; i:=0;//вычисление интеграла
   while b00<=b0 do begin if Nev=1 then begin
s1:=abs(Integral(a0,b0,a00,b00,Ck12)-Integral(a0,b0,a00,b00,Ck11));
s2:=abs(Integral(a0,b0,a00,b00,Ck22)-Integral(a0,b0,a00,b00,Ck21)); end
 else begin s1:=abs(Integral(a0,b0,a00,b00,Ck12)-
Integral(a0,b0,a00,b00,Ck11))/(1e-7+abs(Integral(a0,b0,a00,b00,Ck12)));
  s2:=abs(Integral(a0,b0,a00,b00,Ck22)-
Integral(a0,b0,a00,b00,Ck21))/(1e-7+abs(Integral(a0,b0,a00,b00,Ck22))); end;
    if s1>s2 then s:=s1 else s:=s2; if s>max then max:=s;
    inc(i); a00:=a0+i*otrezok_integr; b00:=a00+otrezok_integr; end;
   if max<Min then begin Min:=max;
Ck1_^:=Ck12^;Ck11_^:=Ck11^;Ck2_^:=Ck22^;Ck21_^:=Ck21^ end; inc(k);
  until k>Kmax-1; if Min<MinGL then begin MinGL:=Min;
Ck1_opt^:=Ck1_^;Ck11_opt^:=Ck11_^;Ck2_opt^:=Ck2_^;Ck21_opt^:=Ck21_^ end;
  inc(n); until n>Nmax;
 writeln('PogrMax= ',MinGL,'  N= ',(Ck1_opt^[0,-3]-1):3:0,
 ' K= ',Ck1_opt^[0,-4]:3:0); writeln;
 Subinterval(trunc(Ck1_opt^[0,-4]),trunc(Ck1_opt^[0,-3]-1),K_iter_itog,a0,b0,
Ynach1,Ynach2,Ck1_opt,Ck2_opt);
 for i:=0 to output do begin //Вывод приближения
  x:=a0 + i*Vel_int / output;
  if abs(x-b0)>1e-17 then pod:= trunc((x-a0)/Ck1_opt^[0,-5])
  else pod:= trunc((x-a0)/Ck1_opt^[0,-5]) - 1;
writeln(x:18:3,', pod=',pod,', ',abs(Gorner(Ck1_opt^[pod],x)-fun1(x)),
'  ',abs(Gorner(Ck2_opt^[pod],x)-fun2(x)));
  writeln(res,x:18:3,'  pod=',pod,', ', abs(Gorner(Ck1_opt^[pod],x)-fun1(x)),
', ', abs(Gorner(Ck2_opt^[pod],x)-fun2(x))); end;
 readln;pod:=trunc(exp(trunc(Ck1_opt^[0,-4])*ln(2)))-1;
 Ynach1:=Gorner(Ck1_opt^[pod],b0); Ynach2:=Gorner(Ck2_opt^[pod],b0);
 a0:=b0; b0:= a0+Vel_int;end;
Dispose(Ck1_); Dispose(Ck1_opt); Dispose(Ck1); Dispose(Ck11); Dispose(Ck12);
Dispose(Ck21);Dispose(Ck22); Dispose(Ck11_); Dispose(Ck11_opt); Dispose(Ck2);
Dispose(Ck21_); Dispose(Ck21_opt); Dispose(Ck2_); Dispose(Ck2_opt); end;
begin assign(res,'result.txt'); rewrite(res);
```

```
Ynach1:=2; Ynach2:=4; // начальные условия
K_iter:=30; K_iter_itog:=30; //количество итераций
Vel_int:=512;  // величина интервала
output:=100; //количество точек для вывода
Nmin:=1; Nmax:=3; Kmin:=1; Kmax:=12;
Method_Iter:=1;// 1 - аналог метода простой итерации, 2 - аналог метода Зейделя
Nev:=2;// 1 - аболютная погрешность, 2 - относительная погрешность
otrezok_integr:=sqrt(sqrt(vel_int));
RD(Ynach1,Ynach2,Vel_int,K_iter,Nmin,Nmax,Kmin,Kmax,output,otrezok_integr);
 readln;close(res);end.
```

Результатом работы программы является непрерывное приближение решения задачи (2.47) на отрезке $[1, 513]$, в границах вариаций $1 \le n \le 3$, $1 \le k \le 12$, $\ell_{start} = \ell_{fin} = 30$, $\beta_r - \alpha_r = 512$, $\omega\tau = \sqrt[4]{\beta_r - \alpha_r}$. Норма абсолютной погрешности в равномерно распределенных проверочных точках представлена в столбце *VPI_S2* табл. 3.3 (при разрешении в 1000 точек встречаются значения порядка 10^{-16}).

Таблица 3.3

Абсолютная погрешность приближения решения задачи из примера 3.3

x	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*VPI_S2*
	$h = 10^{-7}$	$h = 10^{-7}$	$h = 10^{-5}$	$h = 10^{-4}$	$h = 10^{-3}$	
6.12	2.507**e-06**	1.431**e-13**	3.469**e-18**	3.539**e-16**	8.674**e-17**	0.000**e+00**
11.24	8.688**e-06**	4.828**e-13**	4.163**e-17**	1.249**e-16**	3.469**e-16**	0.000**e+00**
…	…	…	…	…	…	…
257.00	4.365**e-03**	2.480**e-10**	9.237**e-14**	2.842**e-14**	6.821**e-13**	0.000**e+00**
262.12	4.540**e-03**	2.579**e-10**	9.237**e-14**	4.934**e-14**	3.837**e-13**	0.000**e+00**
…	…	…	…	…	…	…
507.88	1.697**e-02**	9.673**e-10**	5.826**e-13**	1.847**e-13**	2.416**e-13**	0.000**e+00**
513.00	1.731**e-02**	9.867**e-10**	6.537**e-13**	4.832**e-13**	4.263**e-13**	0.000**e+00**

Как и в примерах 3.1, 3.2, в той же последовательности представлены нормы абсолютной погрешности разностных методов на $[1, 513]$. В данном примере разностные методы уступают в точности около 3 десятичных порядков. Метод сохраняет относительно высокую точность приближения решения, неустойчивого к возмущениям начальных условий, при значительном расширении отрезка интегрирования и соответственном увеличении величины отрезка $[\alpha_r, \beta_r]$ из (3.4) (табл. 2.4, табл. 3.3). Как отмечалось в замечании 3.4, длина промежутка кусочно-интерполяционного приближения с высокой точностью по сравнению с разностно-полиномиальным приближением

значительно увеличена. В главе 5 показано, что на этой основе повышается качество моделирования динамических процессов в планетной астрономии и астрофизике.

Для снижения трудоемкости в случае приближенного решения жесткой задачи Коши возможно применение метода с фиксированными значениями параметров n, k, ℓ_{start} и при $\ell_{fin} = 0$.

Пример 3.4. Рассматривается жесткая задача Коши (2.50). В табл. 3.4 приведены нормы абсолютных погрешностей приближения ее решения на отрезке $x \in [0, 128]$, соответствующие наилучшим приближениям, полученным на основе разностных методов и с использованием рассматриваемого метода.

Таблица 3.4

Абсолютная погрешность приближения решения задачи (2.50) на отрезке $x \in [0, 128]$

x	*ode15s* (*MATLAB*)	*Radau* (*SciPy*)	*Butcher_6*	*Dormand-Prince_8*	*FPI_S2*
1.28	6.821**e-13**	9.095**e-13**	1.998**e-15**	6.772**e-15**	0.000**e+00**
2.56	5.457**e-12**	1.819**e-12**	5.684**e-14**	5.196**e-14**	6.939**e-18**
…	…	…	…	…	…
64.00	1.863**e-08**	4.657**e-09**	2.028**e-10**	5.002**e-11**	0.000**e+00**
65.28	2.421**e-08**	4.657**e-09**	2.024**e-10**	3.547**e-11**	4.547**e-13**
…	…	…	…	…	…
126.72	5.774**e-08**	2.608**e-08**	1.913**e-09**	4.284**e-10**	9.095**e-13**
128.00	5.960**e-08**	2.235**e-08**	1.881**e-09**	4.575**e-10**	0.000**e+00**

Обозначения методов *Radau* (*SciPy*), *Butcher_6*, *Dormand-Prince_8* соответствуют обозначениям в табл. 2.12; *ode15s* (*MATLAB*) – функция, реализующая метод конечных разностей переменного порядка с переменным шагом в сочетании с методом Гира для решения жестких задач; *FPI_S2* – кусочно-интерполяцинный метод с фиксированными значениями параметров (программная реализация представлена в приложении к главе 3, п. 1). Граница абсолютной погрешности кусочно-интерполяционного приближения не превышает порядка 10^{-13} на всем отрезке интегрирования (параметры метода $n = 3$, $k = 15$, $\ell_{start} = 28$, $\ell_{fin} = 0$, $\beta_r - \alpha_r = 128$). Разностные методы, как и в примере 3.3, уступают в точности около трех десятичных порядков.

При зафиксированных значениях параметров метода и $\ell_{fin} = 0$ легко рассчитывается число обращений к правой части дифференциальной системы: $fc = 2^k (n\ell_{start} + 1)(x_{fin} - x_0)/(\beta_r - \alpha_r)$. Это количество используется в качестве меры для сравнения трудоемкости различных методов [3, 35]. Во второй строке табл. 3.5 приводятся значения числа обращений к вектор-функции правой части при решении задачи (2.50) различными методами с абсолютной погрешностью, представленной в табл. 3.4, в третьей строке таблицы представлено время выполнения соответствующих программ на персональном компьютере (двоеточия отделяют секунды и миллисекунды).

Таблица 3.5

Количество обращений к правой части системы и время работы программ при приближении решения задачи (2.50) на отрезке $x \in [0, 128]$

	ode15s (*MATLAB*)	*Radau* (*SciPy*)	*Butcher_6*	*Dormand-Prince_8*	*FPI_S2*
time	0:011	0:062	0:375	0:094	0:184
fc	163	939	8960000	1664013	1313078

Таким образом, предложенный метод характеризует значение *fc* меньшее, чем метод Бутчера 6-го и метод Дормана-Принса 8-го порядков аппроксимации, но большее, чем *ode15s* (*MATLAB*) и *Radau* (*SciPy*). Вместе с тем метод позволяет достигать меньшей погрешности приближения: в сравнении со специализированными методами для решения жестких задач, характеризующимися малой трудоемкостью, – на пять десятичных порядков, в сравнении с явными методами Рунге-Кутты высокого порядка – на 3 десятичных порядка.

В случае вариации значений параметров количество обращений кусочно-интерполяционного метода к правой части оценивается из соотношения

$$fc \le 2^{k_{start}+(k_{start}+1)+\cdots+k_{fin}} \times \left(n_{start} \times (n_{start}+1) \times \cdots \times n_{fin} \times \ell_{start} + 1\right) \times (x_{fin} - x_0)/(\beta_r - \alpha_r) +$$
$$+ 2^{k_{fin}} \times (n_{fin} \times \ell_{fin} + 1) \times (x_{fin} - x_0)/(\beta_r - \alpha_r),$$

где *start* и *fin* индексируют начальные и конечные границы вариаций, рост точности приближенного решения достигается ценой увеличения трудоемкости согласно этому соотношению.

Дополнительно, в главе 5, посвященной численному моделированию, будут приведены результаты с оценкой погрешности и временной сложности кусочно-интерполяционного решения модели реакции Белоусова-Жаботинского, характеризующейся высокой степенью жесткости.

В случае системы возможны видоизменения итераций: на текущей итерации в уравнение с номером j можно подставлять приближения на этой же итерации всех переменных с номерами $1, 2, \ldots, j-1$, остальные приближения берутся с предыдущей итерации (по аналогии с методом Зейделя из [144] решения систем линейных алгебраических уравнений). Согласно эксперименту такое видоизменение не улучшает точность приближения. Возможен процесс с различными параметрами k, n, ℓ для различных уравнений системы, вследствие усложнения алгоритма это видоизменение программно не реализовано.

3.7. Динамическая коррекция данных дифференциальной модели периодических процессов. Динамическая коррекция начальных значений выполняется на основе соответствия между минимальным значением погрешности решения и минимальным значением невязки вида

$$\Delta_M(x, r, k, n, \ell) = \frac{\left| P^{(\ell)}_{kri_1 n+1}(x) - P^{(\ell)}_{(k-1)ri_2 n+1}(x) \right|}{\varepsilon + \left| P^{(\ell)}_{kri_1 n+1}(x) \right|}, \qquad (3.49)$$

где M из (3.3); i_1, i_2 – индексы отрезков, которым принадлежит x из $[\alpha_r, \beta_r]$ соответственно в случае 2^k и 2^{k-1} отрезков (3.4), $P^{(\ell)}_{kri_1 n+1}(x) = P^{(\ell)}_{ri_1 n+1}(x)$ из (3.17) для количества отрезков 2^k, аналогично, $P^{(\ell)}_{(k-1)ri_2 n+1}(x) = P^{(\ell)}_{ri_2 n+1}(x)$ для количества отрезков 2^{k-1}. При решении задачи (3.1) в границах отрезка $[\alpha_r, \beta_r]$

фиксируется $\min\limits_{i} \Delta_M(x, r, k, n, \ell)$, где i – номер текущего значения невязки, и сравнивается со значением невязки в β_r. Если оно не больше последнего, при этом

$$\min_{i} \Delta_M(x,r,k,n,\ell) \le \min_{i-1} \Delta_M(x,r,k,n,\ell)$$

и

$$\min_{i} \Delta_M(x,r,k,n,\ell) \le \min_{i+1} \Delta_M(x,r,k,n,\ell),$$

то в точке, где достигается $\min\limits_{i} \Delta_M(x,r,k,n,\ell)$, значение приближенного решения выбирается в качестве нового начального значения. От точки минимума отсчитывается новый отрезок длины $|\beta_r - \alpha_r|$, и весь процесс приближения возобновляется от этой точки как от начальной. Процесс целиком воспроизводится на сдвинутом таким образом отрезке длины $|\beta_r - \alpha_r|$, включая поиск границ, вариацию параметров k, n, ℓ, итерационное уточнение и коррекцию начальных значений. Сдвиг отрезка с описанным процессом рекуррентно возобновляется до конца промежутка приближенного решения $[x_0, x_{fin}]$. В листинге 3.1 значение невязки (3.49) вычисляет функция *S_max*:

```
function S_max(xx:extended):extended; var pod,pod1:integer;
begin
 pod :=trunc((xx-a0)/Ck_opt^[0,-5]); pod1:=trunc((xx-a0)/Ck1_opt^[0,-5]);
 S_min:=abs(Gorner(Ck_opt^[pod],xx)-Gorner(Ck1_opt^[pod1],xx))/
  (1e-7+abs(Gorner(Ck_opt^[pod],xx)));
end;
```

Значение x из $[\alpha_r, \beta_r]$, где достигается $\min\limits_{i} \Delta_M(x,r,k,n,\ell)$, определяется дискретно в равномерно распределенных проверочных точках x_i, расстояние между которыми составляет величину $(\beta_r - \alpha_r)/\gamma$, где γ – параметр (в программной реализации $\gamma = 100$). Искомое значение x определяется с помощью следующих операторов:

```
pod:=trunc(exp(trunc(Ck_opt^[0,-4])*ln(2)))-1;
pod1:=trunc(exp(trunc(Ck1_opt^[0,-4])*ln(2)))-1;
Min_b0:=abs(Gorner(Ck_opt^[pod],b0)-Gorner(Ck1_opt^[pod1],b0))/
```

```
        (1e-7+abs(Gorner(Ck_opt^[pod],b0)));
XFinal:=b0; h:=Vel_int/100; x:=a0+20*h; Min:=maxExtended; i:=21;
repeat
s_l1:=S_min(x-h); s_r1:=S_min(x+h);s:=S_min(x);
if (s<=s_l1) and (s<=s_r1) and (s<=Min_b0) and (s<Min) then
  begin XFinal:=x; Min:=s; end;
inc(i);x:=a0+i*h;
until x> b0-2*h;
```

С целью снижения трудоемкости значение x определяется из отрезка $[\alpha_r + \delta\times|\beta_r - \alpha_r|, \beta_r]$, $0<\delta<1$, (в программной реализации $\delta = 0.2$).

При решении задачи (3.35) аналогичный процесс выполняется на основе минимума нормы невязки, компоненты которой имеют структуру (3.49). Данная коррекция использована в экспериментах примеров 3.1, 3.2 и не использована в примерах 3.3, 3.4.

3.8. Модификация обработки данных для модели двухточечной задачи Коши. Рассматривается двухточечная задача Коши для системы ОДУ (3.35) на $[x_0, x_{fin}]$ в виде следующего частного случая (заданы начальные и конечные значения решения задачи Коши):

$$Y' = F(x,Y),\ Y(x_0)=Y_0,\ Y(x_{fin})=Y_{fin},$$

где Y, $F(x,Y)$, Y_0 – те же, что в (3.35), $Y_{fin}=(y_{fin\,1}, y_{fin\,2},\ldots, y_{fin\,N})$. Для ее решения отрезок $[x_0, x_{fin}]$ делится на две части, примерно, возле середины: $[x_0, x_{fin}]=[x_0, x_{pr}]\bigcup[x_{pr}, x_{fin}]$. На отрезке $[x_0, x_{pr}]$ задача решается в точности описанным выше способом (движение вдоль этого отрезка обозначается $\uparrow[x_0, x_{pr}]$). На отрезке $[x_{pr}, x_{fin}]$ решение задачи выполняется в полной аналогии с предыдущим, однако за вектор начальных значений принимается $Y(x_{fin})=Y_{fin}$, и от точки x_{fin} движение решения идет в обратном направлении – к x_{pr} (движение в обратном направлении обозначается $[x_{pr}, x_{fin}]\downarrow$). Обратное направление реализуется отрицательным значением шага $-h$ во всех конечных разностях, построенных от x_{fin} как от начальной точки, в каждом

интерполяционном полиноме Ньютона на каждом отрезке из (12). Итерационное уточнение также строится в обратном направлении и имеет вид:

$$\left\{\Psi^{(\ell)}_{jrin}(t)\right\}_{j=1}^{N} \approx F\left(x,\, \left\{P^{(\ell)}_{jri\,n+1}(x)\right\}_{j=1}^{N}\right),$$

$$\left\{P^{(\ell+1)}_{jri\,n+1}(x)\right\}_{j=1}^{N} = \overline{Y}(\widetilde{x}_{ri}) + \int\limits_{\widetilde{x}_{ri}}^{x} \left\{\Psi^{(\ell)}_{jrin}(t)\right\}_{j=1}^{N} dx, \quad Y(x) \approx \left\{P^{(\ell+1)}_{jri\,n+1}(x)\right\}_{j=1}^{N},$$

где $\widetilde{x}_{ri}$ – правый конец отрезка и $x < \widetilde{x}_{ri}$. Данных изменений достаточно для получения кусочно-интерполяционного приближения на $[x_{pr}, x_{fin}]\downarrow$ с теми же свойствами и с той же точностью, как в прямом направлении [145, 146]. Приближения решения и его производной сшиваются на $\uparrow[x_0, x_{pr}]$ и на $[x_{pr}, x_{fin}]\downarrow$ в точке x_{pr} с помощью интерполяционного полинома Ньютона той же степени, которая получена на правой границе x_{pr} по направлению $\uparrow[x_0, x_{pr}]$. При этом часть узловых значений заимствуется слева от x_{pr}, часть – справа от этой точки на $[x_{pr}, x_{fin}]\downarrow$. Полученный полином подвергается итерационному уточнению. Метод реализует программа, листинг которой приводится в приложении к главе 3 (п. 2). Результат работы данной программы в 100 проверочных точках на отрезке [1, 512] приводится для задачи:

$$y_1' = x + 2y_1 / x - \sqrt{y_2},\ y_2' = 2\sqrt{y_2},\ y_1(1) = 2,\ y_2(1) = 4,$$

$$y_1(512) = 262656,\ y_2(512) = 263169$$

и имеет вид

x	1.00	6.12	…	26.60	31.72	36.84	…	509.44	512.00
ε	0.00	0.00	…	0.00	1.11**e-16**	0.00	…	0.00	0.00

В первой строке таблицы – значения независимой переменной, во второй – соответственная норма абсолютной погрешности приближенного решения. Точность приближения сравнима с точностью изложенного метода для одноточечной задачи Коши, приближение решения также представляется в

аналитическом виде (сохраняется непрерывность и дифференцируемость приближения на всем промежутке).

Таким образом, в главе предложено и обосновано модифицированное построение кусочно-интерполяционного приближения данных с итерационным уточнением в ИВС для различных классов задач, включая задачи с неустойчивыми решениями и жесткие задачи. Приближение выполняется на основе единой программы, переменные параметры которой настраиваются в зависимости от условий задачи. Для повышения быстродействия, в частности при моделировании в реальном времени, автоматический программный подбор параметров может быть заменен пользовательским подбором и фиксированием значений параметров, что дополнительно исследовано в главе 5. Предложенная модификация отличается повышенной точностью обработки данных на больших отрезках времени и позволяет улучшать качество моделирования в ИВС периодических процессов.

3.9. Выводы

1. Разработан метод кусочно-интерполяционной обработки данных с итерационным уточнением для моделей периодических процессов. Метод отличается от аналогов кусочной интерполяцией на временных подынтервалах правой части дифференциальной системы и интегральным приближением решения в виде алгебраических полиномов с числовыми коэффициентами, а также выполнением итераций на подынтервалах, аналогичных интегральным приближениям Пикара. Метод отличается повышением точности обработки данных на больших отрезках времени относительно известных методов, а также улучшением качества численного моделирования.

2. Показана сходимость предложенной модификации метода кусочно-интерполяционной обработки данных и сходимость итерационного уточнения, даны оценки скорости сходимости. Качества равномерной сходимости и аналитический вид приближения числовых данных отличают предложенную модификацию от известных аналогов.

3. Выполнена алгоритмизация и программная реализация метода в виде стандартной программы ИВС, на вход которой поступает правая часть дифференциальной системы, моделирующей процесс, начальные данные и параметры, включающие границы временного отрезка обработки данных модели. Автоматический выбор параметров обеспечивает наибольшую точность при наименьшем времени обработки. Варьируемые параметры программно адаптируются к структуре модели и реализуют динамическую коррекцию начальных данных. В результате достигается вычислительная устойчивость, высокая точность, гладкость аналитического приближения данных на отрезке произвольной длины, что составляет отличие метода для широкого класса моделей периодических процессов в ИВС, в числе которых модели жестких задач и задач с неустойчивыми по Ляпунову решениями.

4. Отличительным качеством предложенного метода является его применимость к обработке данных для частного случая модели двухточечной задачи Коши, где сохраняется высокая точность и гладкость аналитического приближения данных.

ГЛАВА 4. ВАРЬИРУЕМАЯ КУСОЧНО-ИНТЕРПОЛЯЦИОННАЯ ОБРАБОТКА ДАННЫХ ДЛЯ МОДЕЛИ ПЕРЕНОСА

В главе 1 даны некоторые примеры задач из различных областей науки, при моделировании которых актуальны проблемы повышения точности обработки данных для моделей с частными производными, в частности, обработки данных модели переноса. Глубокому исследованию проблемы посвящены работы [147 – 149], где отмечается принципиальное значение модели переноса для численного решения нестационарных задач механики сплошной среды и задач газовой динамики. Границы погрешности существующих методов в прямоугольной области небольшого размера, в условиях гладкости решения, как правило, находятся в диапазоне $10^{-11} - 10^{-8}$. Для снижения погрешности в главе предлагается метод варьируемой кусочно-интерполяционной обработки числовых данных модели переноса с итерационным уточнением. Построение выполняется на основе интерполяционного полинома Ньютона от двух переменных, программно преобразуемого в алгебраический полином с числовыми коэффициентами. Применяется двумерное итерационное уточнение обработанных данных, что позволяет получить кусочно гладкое аналитическое приближение движения волны в прямоугольной области. В случае линейной модели переноса доказывается сходимость кусочно-интерполяционной обработки данных, оценивается скорость сходимости. Для квазилинейной модели переноса предложена схема вывода аналогичных оценок. Выполнена алгоритмизация и программная реализация метода, представлены результаты численного эксперимента. Схема построения метода формально допускает аналоги для для моделей на основе интегральных и интегро-дифференциальных уравнений.

Материал главы соответствует содержанию публикаций автора [151 – 166].

4.1. Приближение функций в ИВС с вариацией подобласти и степени полинома. В декартовой системе координат UXT кусочно-интерполяционное приближение действительной функции $u = u(x, t)$ двух действительных переменных в прямоугольной области

$$G = \{ (x, t) \mid x \in [a, b],\ t \in [c, d] \} \tag{4.1}$$

строится следующим образом. Область (4.1) разбивается на прямоугольные подобласти G_{ij} с пересекающимися границами:

$$G = \bigcup_{j=0}^{P_t - 1} \bigcup_{i=0}^{P_x - 1} G_{ij}, \tag{4.2}$$

$$\begin{aligned} &G_{ij} = \{ (x, t) \mid x \in [x_i, x_{i+1}],\ t \in [t_j, t_{j+1}] \}, \\ &P_x = 2^{k_x},\ P_t = 2^{k_t},\ k_x, k_t \in \{ 0, 1, \dots \}. \end{aligned} \tag{4.3}$$

Интерполяционный полином Ньютона от двух переменных в подобласти (4.3) использует $(n+1)(n+2)/2$ узлов с треугольным расположением:

$$\begin{array}{lllll} (x_{i0}, t_{jn}) & & & & \\ (x_{i0}, t_{j\,(n-1)}) & (x_{i1}, t_{j(n-1)}) & & & \\ \quad \dots & \quad \dots & \dots & & \\ (x_{i0}, t_{j0}) & (x_{i1}, t_{j0}) & \dots & (x_{i\,(n-1)}, t_{j0}) & (x_{i\,n}, t_{j0}) \end{array} \tag{4.4}$$

При расположении узлов (4.4) функция интерполируется в нижней треугольной части G_{ij}, в верхней части фактически нужна экстраполяция, что учитывается в дальнейшем. Пусть произвольно задана граница ε абсолютной погрешности приближения функции $u(x, t)$. В G_{ij} интерполяционный полином Ньютона $\Psi_n^{ij}(z, w)$ строится с равноотстоящими на шаги h_x, h_t по направлениям координат узлами $x_{i\ell}$, t_{jm}, где

$$\begin{aligned} &z = (x - x_i)/h_x,\ w = (t - t_j)/h_t,\ (x, t) \in G_{ij}, \\ &h_x = (x_{i+1} - x_i)/n,\ h_t = (t_{j+1} - t_j)/n, \end{aligned} \tag{4.5}$$

$$x_{i\ell} = x_i + \ell h_x,\ t_{jm} = t_j + m h_t,\ \ell = \overline{0,\ n},\ m = \overline{0,\ n-\ell}\ . \qquad (4.6)$$

Искомый полином примет вид [144]

$$\Psi_n^{ij}(z, w) = u(x_{i0}, t_{j0}) + \sum_{k=1}^{n} \sum_{s=0}^{k} \frac{\Delta^k_{x^s t^{k-s}} u(x_{i0}, t_{j0})}{s!\,(k-s)!} \prod_{\ell=0}^{s-1}(z-\ell) \prod_{m=0}^{k-s-1}(w-m), \qquad (4.7)$$

где $\Delta^k_{x^s t^{k-s}} u(x_{i0}, t_{j0})$ – конечные разности k-го порядка, которые при $k = \overline{1,\ n},\ s = \overline{0,\ k}$ с учетом (4.6) вычисляются из соотношений:

$$\begin{cases} \Delta_{x^1 t^0} u(x_{i\ell}, t_{jm}) = u(x_{i(\ell+1)}, t_{jm}) - u(x_{i\ell}, t_{jm}), \\ \Delta_{x^0 t^1} u(x_{i\ell}, t_{jm}) = u(x_{i\ell}, t_{j(m+1)}) - u(x_{i\ell}, t_{jm}), \\ \Delta^k_{x^s t^{k-s}} u(x_{i\ell}, t_{jm}) = \Delta^{k-1}_{x^s t^{k-s-1}} u(x_{i\ell}, t_{j(m+1)}) - \Delta^{k-1}_{x^s t^{k-s-1}} u(x_{i\ell}, t_{jm}), \text{ при } k > s, \\ \Delta^k_{x^s t^{k-s}} u(x_{i\ell}, t_{jm}) = \Delta^{k-1}_{x^{s-1} t^{k-s}} u(x_{i(\ell+1)}, t_{jm}) - \Delta^{k-1}_{x^{s-1} t^{k-s}} u(x_{i\ell}, t_{jm}), \text{ при } k = s, \end{cases}$$

$k = \overline{2,\ n}$, $s = \overline{0,\ k}$, $\ell = \overline{0,\ n-k}$, $m = \overline{0,\ n-k-\ell}$.

Код процедуры *Dual_finite_differences*, выполняющей вычисление конечных разностей, дан в листинге 4.1 (код данной процедуры и все приводимые ниже программные реализации констуируемых алгоритмов, как и в предыдущих главах, представлены на языке *Object Pascal* среды *Delphi*).

Листинг 4.1

```
Procedure Dual_finite_differences(UU:Mass2; N1:integer; var dz:mass5_);
var i,j,k,m:integer;
begin
for i:=0 to N1-1 do for j:=0 to N1-1-i do begin
  dz[1,1,0,i,j]:=UU[i+1,j]-UU[i,j]; dz[1,0,1,i,j]:=UU[i,j+1]-UU[i,j]; end;
 for m:=2 to N1 do for k:=0 to m do for i:=0 to N1-m do for j:=0 to N1-i-m do
if m-k<>0 then dz[m,k,m-k,i,j]:=dz[m-1,k,m-k-1,i,j+1]-dz[m-1,k,m-k-1,i,j]
else dz[m,k,m-k,i,j]:=dz[m-1,k-1,m-k,i+1,j]-dz[m-1,k-1,m-k,i,j]
end;
```

Для произвольно заданного ε степень полинома n выбирается одинаковой для всех G_{ij} и минимальной при условии

$$|u(x,t) - \Psi_n^{ij}(z,w)| \le \varepsilon\ \forall (x,t) \in G_{ij},\ i = \overline{0,\ P_x - 1},\ j = \overline{0,\ P_t - 1}. \qquad (4.8)$$

Проверка точности приближения (4.8) в каждой подобласти выполняется на множестве точек, $x_{is} = x_i + s\,h_x/\gamma$, $t_{jr} = t_j + r\,h_t/\gamma$, $s, r \in \{0, 1, \dots\}$, где $\gamma \geq 3$ – параметр. Минимальность n обеспечивается следующим образом. Построение и проверка точности начинаются с $n = 1$ и $k_x = 0$, $k_t = 0$ во всех G_{ij} из (4.3). При нарушении неравенства (4.8) хоть в одной проверочной точке какой-либо подобласти значения k_x и k_t увеличиваются на единицу, и проверка начинается сначала. Процесс продолжается до нарушения априори заданных границ $k_x \leq k_0$, $k_t \leq k_0$. Если в результате заданная точность не достигнута, то полагается $k_x = 0$, $k_t = 0$, степень n увеличивается на единицу и проверка возобновляется сначала для этой степени в тех же границах k_x и k_t. Описанный процесс циклически воспроизводится до нарушения априори заданной границы $n \leq N_0$. В качестве искомого фиксируется наименьшее n, при котором неравенство (4.8) выполняется одновременно во всех проверочных точках всех $2^{k_x+k_t}$ подобластей, в соответствии с этим фиксируются текущие значения k_x и k_t.

Замечание 4.1. Ниже (при выполнении оценок сходимости и скорости сходимости) абстрактно предполагается, что степень интерполяционного полинома ограничена, в то время как количество подобластей в (4.2), (4.3) не ограничивается: $n \leq N_0$, $N_0 = \text{const}$; $k_x \to \infty$, $k_t \to \infty$.

Полином (4.7) эквивалентно преобразуется к виду алгебраического полинома с числовыми коэффициентами. С этой целью, по соотношениям (2.6) – (2.9), реализующим видоизменение формул Виета [75], вычисляются коэффициенты полиномов, представленных в (4.7) в виде разложения по корням, – $P_s(z) = \prod_{\ell=0}^{s-1}(z-\ell)$, $P_{k-s}(w) = \prod_{m=0}^{k-s-1}(w-m)$. При уже вычисленных

коэффициентах полиномов $P_s(z)=\sum_{\ell=0}^{s} d_{s\ell}\, z^{\ell}$, $P_s(w)=\sum_{m=0}^{k-s} \widetilde{d}_{(k-s)m}\, w^{m}$ имеет место равенство

$$\prod_{\ell=0}^{s-1}(z-\ell)\prod_{m=0}^{k-s-1}(w-m)=\sum_{\ell=0}^{s}\sum_{m=0}^{k-s} D_{ks\ell m}\, z^{\ell}\, w^{m},$$

где $D_{ks\ell m}=d_{s\ell}\,\widetilde{d}_{(k-s)m}$. Коэффициенты $D_{ks\ell m}$ не зависят от аппроксимируемой функции, после вычисления по стандартной подпрограмме, код которой представлен в листинге 4.2, они хранятся в памяти компьютера для дальнейшего применения посредством считывания.

Листинг 4.2

```
procedure Viet2(var DD: Mass4_); var k,l,i,j,m_: shortint; d,e: Mass2_int;
begin e[1,1]:=1; e[1,0]:=0;
for k:=2 to nn do begin
e[k,0]:=-e[k-1,0]*(k-1);
for l:=1 to k-1 do e[k,k-l]:=e[k-1,k-l-1]-e[k-1,k-l]*(k-1);
  e[k,k]:=e[k-1,k-1] end;
 for k:=1 to nn do for l:=0 to k do d[l,k]:=e[k,l];
 for m_:=1 to nn do for k:=0 to m_ do
for i:=0 to k do for j:=0 to m_-k do
  begin if (k=0) then Dd[m_,k,i,j]:= d[j,m_-k]
else if (m_-k=0) then Dd[m_,k,i,j]:= d[i,k]
else Dd[m_,k,i,j]:=d[i,k]*d[j,m_-k] end
end;
```

После данных преобразований полином (4.7) переводится в форму полинома с числовыми коэффициентами по дистрибутивности с приведением подобных. В результате

$$\Psi_n^{ij}(z,w)=\sum_{\ell=0}^{n}\sum_{m=0}^{n-\ell} a_{\ell m}^{ij}\; z^{\ell}\, w^{m}. \qquad (4.9)$$

где $a_{00}^{ij}=u(x_{i0},t_{j0})$ и, если $\ell\neq 0\vee m\neq 0$, то $a_{\ell m}^{ij}=\sum_{k=m}^{n}\sum_{s=\ell}^{k-m}\frac{\Delta^{k}_{x^{s}t^{k-s}}u(x_{i0},t_{j0})}{s!\,(k-s)!}D_{ks\ell m}$,

$\ell=\overline{0,n}$, $m=\overline{0,n-\ell}$.

Значение (4.9) вычисляется следующим способом:

$$\Psi_n^{ij}(z,w)=(\dots[\,(a_{0n}^{ij}w+a_{1\,n-1}^{ij}z+a_{0\,n-1}^{ij})\,w+$$
$$+(a_{2n-2}^{ij}z+a_{1\,n-2}^{ij})\,z+a_{0\,n-2}^{ij}\,]\,w+\dots)\,w+\dots+a_{00}^{ij}.$$

В листинге 4.3 приводится код функции *Polinom*, реализующей вычисление значений полинома (4.9) при известных коэффициентах по данной схеме.

Листинг 4.3

```
function Polinom(xx,tt: extended; CC: mass2): extended;
var i,j,N: integer; zp,wp,polin,polinom_z: extended;
begin zp:=(xx-CC[-1,0])/CC[-2,0]; wp:=(tt-CC[0,-1])/CC[0,-2];
 N:=trunc(CC[-1,-1]); polin:=CC[0,N];
for i:=N-1 downto 0 do begin
  polin:=polin*wp; polinom_z:=CC[N-i,i];
for j:=N-i-1 downto 0 do polinom_z:=polinom_z*zp+CC[j,i];
  polin:=polin+polinom_z; end;
Result:=polin; end;
```

Для вычисления $u(x,t)$, $\forall(x,t)\in G$, дешифрируются индексы G_{ij}: $i=\left[(x-a)/\rho_x\right]$, $j=\left[(t-c)/\rho_t\right]$, $[\ ]$ – целая часть числа, $\rho_x=x_{i+1}-x_i$, $\rho_t=t_{j+1}-t_j$, $x\in[x_i,x_{i+1})$, $t\in[t_j,t_{j+1})$. Индексы определяют адрес массива коэффициентов полинома (4.9) в памяти компьютера, программа вычисления приводится в приложении к главе 4, п. 1.

С целью оценки погрешности полином (4.7) рассматривается в эквивалентной форме

$$\Psi_n^{ij}(x,t)=u(x_{i\,0},t_{j0})+\sum_{m=1}^{n}\sum_{k=0}^{m}\frac{\Delta^m_{x^k t^{m-k}}\,u(x_{i\,0},t_{j0})}{h_x^k\,h_t^{m-k}k!\,(m-k)!}\prod_{q=0}^{k-1}(x-x_{iq})\prod_{r=0}^{m-k-1}(t-t_{jr}),\quad(4.10)$$

узлы интерполяции определяются из (4.5), (4.6). Сначала (4.10) рассматривается в области G, при этом не используются индексы подобласти. Остаточный член интерполяции в этом случае можно представить в виде [167]

$$R_G(x,t)=\sum_{i=0}^{n+1}\frac{\partial^{n+1}u(\xi_i,\eta_i)}{\partial x^i\partial t^{n+1-i}}\frac{\prod_{\ell=0}^{i-1}(x-x_\ell)}{i!}\frac{\prod_{k=0}^{n-i}(t-t_k)}{(n-i+1)!},\quad(4.11)$$

где (x_ℓ,t_k) – узлы интерполяции, $\ell=\overline{0,\,i-1}$, $k=\overline{0,\,n-i}$, (ξ_i,η_i) – некоторая точка внутри G. Если не оговорено иное, внутри области (подобласти)

производные понимаются в обычном смысле, на границе – как односторонние производные по направлению изнутри к границе. Для дальнейшего предполагается существование, непрерывность, и, следовательно, ограниченность в замкнутой области G всех частных производных до порядка $2n+1$ включительно. Согласно (4.10) достаточно было бы ввести такое предположение относительно порядка $n+1$. Смысл формального завышения порядка гладкости заключается в следующем. Вследствие треугольности расположения узлов (4.4) остаточный член (4.11) фактически относится только к нижней треугольной части, а не ко всей прямоугольной области G. Для оценки погрешности именно во всей области G преобразуется расположение узлов. Интерполяционный полином строится в описанном прямоугольном треугольнике, катеты которого продолжают нижнюю и левую стороны прямоугольника G, длина каждого катета вдвое больше продолжаемой стороны. Гипотенуза треугольника, охватывающего все узлы, пройдет через правую вершину G. В описанном треугольнике узлы (4.4) полностью охватят область G, остаточный член (4.11) будет распространяться на всю эту область. Аналогичное преобразование выполняется для каждой подобласти G_{ij}. Вследствие преобразования расстояния h_x, h_t между узлами в (4.5), при неизменности степени полинома n, окажутся вдвое больше первоначально предполагавшихся в (4.5) значений. Чтобы не возросла погрешность, расстояния будут сохранены такими, как они были даны в (4.5), тогда степень интерполирующего полинома станет равной $2n$. Это соответствует количеству $(2n+1)(2n+2)/2$ узлов в их преобразованном расположении. В дальнейших оценках и в соотношениях вида (4.7) – (4.11) n всегда будет заменяться на $2n$. С данными изменениями имеет место соотношение

$$\left| R_G(x,t) \right| \le C_0 \sum_{i=0}^{2n+1} \max_{0 \le i_0 \le i} h_x^{i_0} h_t^{2n-i_0+1},\ C_0 = \text{const}\ \forall (x,t) \in G,\ \ \forall n \le N_0 . \qquad (4.12)$$

Неравенство (4.12) получается следующим образом. Если для точки (x,t) из (4.11) в слагаемом под знаком $\sum$ (с заменой n на $2n$) выполняется $x\in[x_j, x_{j+1}]$, $0\le j\le i-2$, $t\in[t_r, t_{r+1}]$, $0\le r\le 2n-i-1$, то

$$\prod_{\ell=0}^{i-1}(x-x_\ell)=\prod_{\ell=0}^{j}(x-x_\ell)\prod_{\ell=j+1}^{i-1}(x-x_\ell);\ \prod_{k=0}^{2n-i}(t-t_k)=\prod_{k=0}^{r}(t-t_k)\prod_{k=r+1}^{2n-i}(t-t_k).$$

Поэтому

$$\left|\prod_{\ell=0}^{i-1}(x-x_\ell)\right|\le\prod_{\ell=0}^{j}\left|x_{j+1}-x_\ell\right|\prod_{\ell=j+1}^{i-1}\left|x_j-x_\ell\right|;\ \left|\prod_{k=0}^{2n-i}(t-t_k)\right|\le\prod_{k=0}^{r}\left|t_{r+1}-t_k\right|\prod_{k=r+1}^{2n-i}\left|t_r-t_k\right|.$$

Отсюда

$$\left|\prod_{\ell=0}^{i-1}(x-x_\ell)\right|\le h_x^i\prod_{\ell=0}^{j}(j+1-\ell)\prod_{\ell=j+1}^{i-1}(\ell-j),$$

$$\left|\prod_{k=0}^{2n-i}(t-t_k)\right|\le h_t^{2n-i+1}\prod_{k=0}^{r}(r+1-k)\prod_{k=r+1}^{2n-i}(k-r),$$

или,

$$\left|\prod_{\ell=0}^{i-1}(x-x_\ell)\right|\le h_x^i(j+1)!\,(i-1-j)!,\ \left|\prod_{k=0}^{2n-i}(t-t_k)\right|\le h_t^{2n-i+1}(r+1)!\,(2n-i-r)!.$$

С учетом $(k+1)!\,(m-k)!\le(m+1)!\ \forall\ k,\ 0\le k\le m,$ получится

$$\left|\prod_{\ell=0}^{i-1}(x-x_\ell)\right|\le h_x^i\,i!,\ \left|\prod_{k=0}^{2n-i}(t-t_k)\right|\le h_t^{2n-i+1}(2n-i+1)!.$$

С другой стороны, если для (x,t) в рассматриваемом слагаемом с индексом i_0, $1\le i_0<i$, под знаком $\sum$ выполняется $x\in[x_j, x_{j+1}]$, $j\le i-2$, но $j\ge i_0-1$, то $\left|\prod_{\ell=0}^{i_0-1}(x-x_\ell)\right|\le\prod_{\ell=0}^{i_0-1}\left|x_{j+1}-x_\ell\right|$. Поэтому $\left|\prod_{\ell=0}^{i_0-1}(x-x_\ell)\right|\le h_x^{i_0}\prod_{\ell=0}^{i_0-1}(j+1-\ell)$, следовательно, $\left|\prod_{\ell=0}^{i_0-1}(x-x_\ell)\right|\le h_x^{i_0}\,i!$. Если под знаком $\sum$ в (4.11) с

рассматриваемым видоизменением $t \in [t_r, t_{r+1}]$, $r \le 2n-i-1$, но для i_0, $1 \le i_0 < i$, выполняется $r \ge 2n-i_0$, то $\left|\prod_{\ell=0}^{2n-i_0}(t-t_\ell)\right| \le \prod_{\ell=0}^{2n-i_0}|t_{r+1}-t_\ell|$. Поэтому

$$\left|\prod_{\ell=0}^{2n-i_0}(t-t_\ell)\right| \le h_t^{2n-i_0+1}\prod_{\ell=0}^{2n-i_0}(r+1-\ell),$$

следовательно, $\left|\prod_{\ell=0}^{2n-i_0}(t-t_\ell)\right| \le h_t^{2n-i_0+1}(2n-i+1)!$. В результате

$$\left|\prod_{\ell=0}^{i-1}(x-x_\ell)\right|\left|\prod_{k=0}^{2n-i}(t-t_k)\right| \le \max_{1\le i_0\le i} h_x^{i_0}h_t^{2n-i_0+1}\,i!\,(2n-i+1)!\ \ \forall i \in \overline{1,\,2n}.$$

Тем более,

$$\left|\prod_{\ell=0}^{i-1}(x-x_\ell)\right|\left|\prod_{k=0}^{2n-i}(t-t_k)\right| \le \max_{0\le i_0\le i} h_x^{i_0}h_t^{2n-i_0+1}\,i!\,(2n-i+1)!. \qquad (4.13)$$

Неравенство (4.13) сохраняется в случае $i=0$ и выполняется $\forall i \in \overline{0,\,2n}$. Из (4.11) (при замене n на $2n$)

$$|R_G(x,t)| \le \sum_{i=0}^{2n+1}\left|\frac{\partial^{2n+1}u(\xi_i,\eta_i)}{\partial x^i \partial t^{2n+1-i}}\right|\frac{\left|\prod_{\ell=0}^{i-1}(x-x_\ell)\right|}{i!}\frac{\left|\prod_{k=0}^{2n-i}(t-t_k)\right|}{(2n-i+1)!}. \qquad (4.14)$$

В рассматриваемых предположениях с учетом замечания 4.1

$$\max_G\left|\frac{\partial^{2n+1}u(x,t)}{\partial x^i \partial t^{2n+1-i}}\right| = C_0,\ C_0 = \text{const}\ \ \forall i,\ 0 \le i \le 2n+1,\ \forall n \le N_0. \qquad (4.15)$$

Подстановка в (4.14) правых частей из (4.13) и C_0 из (4.15) влечет (4.12). Всюду ниже предполагается, что размеры G позволяют считать расстояния между узлами меньшими единицы при всех рассматриваемых n. Узлы являются равноотстоящими вдоль направлений осей, поэтому найдутся h, q, p, такие что

$$h_x = qh,\ q = \text{const},\ h_t = ph,\ p = \text{const},\ h < 1,\ q < 1,\ p < 1, p < q\,. \qquad (4.16)$$

Из (4.12) и (4.16) $\left|R_G(x,t)\right| \le C_0 h^{2n+1} \sum_{i=0}^{2n+1} q^{2n+1}$. Очевидно, $\sum_{i=0}^{2n+1} q^{2n+1} \le c_0$, $c_0 = \text{const}$. В обозначении $C = C_0 c_0$,

$$\left|R_G(x,t)\right| \le C h^{2n+1},\ C = \text{const}. \qquad (4.17)$$

Пусть теперь по этой же схеме в каждой подобласти G_{ij} построен интерполяционный полином (4.10) с заменой показателя степени n на $2n$. Тогда расстояния между проекциями узлов на оси координат уменьшатся соответственно в обратной пропорции $P_x = 2^{k_x}$ и $P_t = 2^{k_t}$, где k_x, k_t из (4.3). В (4.12) в тех же пропорциях соответственно уменьшатся h_x, h_t, поэтому $\left|R_{G_{ij}}(x,t)\right| \le C_0 \sum_{r=0}^{2n+1} \max_{0 \le i_0 \le r} \frac{h_x^{i_0}}{2^{i_0 k_x}} \frac{h_t^{2n-i_0+1}}{2^{(2n-i_0+1)k_t}}$. Если выбрать $2^{k_x} = 2^{k_t} = 2^k$, то h в (4.17) обратно пропорционально 2^k, в результате

$$\left|R_{G_{ij}}(x,t)\right| \le C 2^{-k(2n+1)} h^{2n+1}\ \ \forall (x,t) \in G_{ij}\ \forall\, i, j;\ i = \text{const}, j = \text{const}. \qquad (4.18)$$

В (4.18) C, h из (4.17), $h < 1$, i, j из (4.2). Таким образом, имеет место

Лемма 4.1. Пусть функция $u(x,t)$ определена в G из (4.1), где у нее существуют и непрерывны все частные производные до порядка $2n+1$ включительно. Тогда при условии разбиения G на 2^{2k} подобластей (4.2), (4.3) в случае $2^{k_x} = 2^{k_t} = 2^k$, кусочно-интерполяционное приближение данной функции в G с помощью полиномов вида (4.7), взятых в степени $2n$, может быть выполнено с абсолютной погрешностью (4.18), где шаги интерполяции h_x, h_t из (4.5) связаны с $h < 1$ из (4.17) соотношениями (4.16).

Рассматриваемое приближение инвариантно относительно i, j из (4.2), (4.3), поэтому в левой части (4.18) можно взять максимум по всем подобластям:

$$\max_{\forall (x,t)\in G_{ij},\ \forall i,\ j\in\overline{0,2^k-1}} \left| R_{G_{ij}}(x,t)\right| \le C\,2^{-k(2n+1)} h^{2n+1},\ C=\text{const}. \qquad (4.19)$$

Отсюда вытекает

Теорема 4.1. В условиях леммы 4.1 кусочно-интерполяционное приближение равномерно сходится к функции $u(x,t)$ в области G при $k\to\infty$ со скоростью сходимости (4.19).

В каждой подобласти полином (4.7) может быть преобразован к виду (4.9) со степенью $2n$ без изменения оценок (4.18), (4.19). В дальнейшем используются соотношения

$$u(x,t)\approx \sum_{\ell=0}^{2n}\sum_{m=0}^{2n-\ell} a_{\ell m} z^{\ell} w^{m},\ z=(x-x_{i0})/h_x,\ w=(t-t_{j0})/h_t, \qquad (4.20)$$

$$u_x \approx h_x^{-1}\sum_{\ell=1}^{2n}\sum_{m=0}^{2n-\ell} \ell\, a_{\ell m} z^{\ell-1} w^{m},\ u_t \approx h_t^{-1}\sum_{\ell=0}^{2n}\sum_{m=1}^{2n-\ell} m\, a_{\ell m} z^{\ell} w^{m-1}, \qquad (4.21)$$

$$\int u(x,t)\,dt \approx \widetilde{c} + h_t \sum_{\ell=0}^{2n}\sum_{m=0}^{2n-\ell} a_{\ell m} z^{\ell}\, w^{m+1}/(m+1),\ \widetilde{c}=\text{const}. \qquad (4.22)$$

Значения констант и параметров определяются по ходу изложения, h_x, h_t в (4.20) – (4.22) пропорциональны h согласно (4.16).

4.2. Кусочно-интерполяционная обработка данных модели переноса. Исходные предположения. Вначале рассматривается задача Коши для линейного уравнения

$$\frac{\partial u}{\partial t} + a(x,t)\,\frac{\partial u}{\partial x} = f(x,t),\ \ u(x,0)=\varphi(x), \qquad (4.23)$$

где $a(x,t)$, $f(x,t)$ – заданные функции, рассматриваемые в полуплоскости $\{(x,t)\,|\,x\in R, t\ge 0\}$, $\varphi(x)$ – заданная функция $x\in R$ [148]. Для построения метода приближенного решения выбрана прямоугольная область задания этих функций и одна из ее границ, определяемые непосредственно ниже. Ввиду

применения кусочной интерполяции с оценками (4.14), (4.19) в условиях теоремы 4.1 для анализа сходимости используются следующие ограничения.

I. Приближенное решение задачи (4.23) строится в области G из (4.1), объединяющей подобласти G_{ij} из (4.2), (4.3) при $2^{k_x} = 2^{k_t} = 2^k$, областью определения $\varphi(x)$ служит основание G на оси OX. Значения a и b не конкретизируются, вместе с тем всюду ниже $c = 0$, $d = T$, $t \in [0, T]$. Предполагается, что $u(x,t)$ принадлежит области $\{ |u(x,t) - u(x,0)| \le \widetilde{b}; (x,t) \in G \}$, где $\widetilde{b}$ – постоянная, значение которой может быть задано произвольно, при необходимости оно конкретно оговаривается.

II. Предполагается, что в области $G_\Delta = \{ (x,t) \mid x \in [a - \Delta, b + \Delta],\ t \in [0, T] \}$, при $\forall\, \Delta\colon\ 0 < \Delta < 2^{-1}(b-a)$, $\Delta = \text{const}$, существуют и непрерывны все частные производные $u(x,t)$ до порядка $2n+1$ включительно $\forall n \le N_0$, $N_0 = \text{const}$, с таким же порядком непрерывно дифференцируемы $a(x,t)$ и $f(x,t)$, функция $\varphi(x)$ определена и непрерывно дифференцируема $2n+1$ раз на отрезке $x \in [a - \Delta, b + \Delta]$.

III. Предполагается, что в подобласти $G_{\Delta ij} = \{ (x,t) \mid x \in [x_i - \Delta,\ x_{i+1} + \Delta],\ t \in [t_j, t_{j+1}] \}$ значение Δ может быть произвольным в границах $0 < \Delta < 2^{-k-1}(b-a)$, $\Delta = \text{const}$.

В предположениях I, II обусловлено существование и единственность, а также устойчивость решения задачи (4.23) относительно возмущения начальных данных [148].

В G_{ij} приближенное решение строится с помощью интерполяционного полинома с итерационным уточнением, последовательно по j выполняется переход от G_{ij} к $G_{i(j+1)}$. За начальные условия в $G_{i(j+1)}$ принимается приближение из G_{ij} на смежной с $G_{i(j+1)}$ границе. В G_{00} узловые значения интерполяции задает функция $\varphi(x)$. Конкретно интерполируется

$f(x,t)-a(x,t)\dfrac{\partial u}{\partial x}$, что дает приближение $\dfrac{\partial u}{\partial t}$. Интеграл по времени от интерполяционного полинома принимается за приближение решения, которое подставляется в выражение $\dfrac{\partial u}{\partial t}$ из (4.23). Полученное приближение $\dfrac{\partial u}{\partial t}$ снова интерполируется, и описанный процесс повторяется с использованием (4.20) – (4.22). На практике итерации выполняются до искомой точности приближения, абстрактно их количество предполагается неограниченным. Процесс воспроизводится в каждой подобласти до полного прохода области G.

4.3. Итерационное уточнение данных. В G_{ij}

$$\int_{t_0}^{t}\frac{\partial u(x,t)}{\partial t}dt=\int_{t_0}^{t}\left(f(x,t)-a(x,t)\frac{\partial u(x,t)}{\partial x}\right)dt, \qquad (4.24)$$
$$(x,t)\in G_{ij},\ t_0=t_0(i,j),\ u(x,t_0)=\varphi_{ij}(x,t_0).$$

В дальнейшем обозначения корректируются с целью отличия от использованных при описании интерполяции функции. К нижним индексам полинома $\Psi_{2n}^{ij}(z,w)$ из (4.20) добавлен индекс $2k$ в соответствии с числом подобластей 2^{2k}, аналогичная индексация применяется к другим выражениям. Полином $\Psi_{2k\,2n}^{ij}(z,w)$ интерполирует не $u(x,t)$, а подынтегральную функцию правой части (4.24), что отмечается слитным индексом ∂u, остаточный член интерполяции $c_{2k\,2n}$ оценивается из (4.19), –

$$\Psi_{\partial u\,2k\,2n}^{ij}(z,w)=f(x,t)-a(x,t)\frac{\partial\tilde{u}(x,t)}{\partial x}+c_{2k\,2n},\ \left|c_{2k\,2n}\right|\le C\,2^{-k(2n+1)}h^{2n+1}, \qquad (4.25)$$

где z, w из (4.20), $\tilde{u}(x,t)\approx u(x,t)$ – приближение решения на предыдущей итерации. Аналогично (4.20), $\Psi_{\partial u\,2k\,2n}^{ij}(z,w)$ преобразуется к виду (4.9). Решениена текущей итерации приближается интегралом от $\Psi_{\partial u\,2k\,2n}^{ij}(z,w)$, определяемым аналогично (4.22). Полученный полином обозначается

${}^{t}P_{u2k2n}(x,t)$ – по степени $2n$, числу подобластей 2^{2k} и приближению $u(x,t)$ путем интегрирования по t полинома $\Psi^{ij}_{\partial u\ 2k2n}(z,w)$. Итерационный процесс примет вид

$$\Psi^{ij}_{\partial u\ 2k2n(r-1)}(z,w)=f(x,t)-a(x,t)\frac{\partial {}^{t}P_{u2k2n(r-1)}(x,t)}{\partial x}+c_{2k2n},$$

$${}^{t}P_{u2k2nr}(x,t)={}^{t}P_{u2k2nr}(x,t_0)+\int_{t_0}^{t}\Psi^{ij}_{\partial u\ 2k2n(r-1)}(z,w)\,dt+{}^{t}c_{u2k2n},$$

где ${}^{t}P_{u2k2nr}(x,t_0)=\varphi_{ij}(x,t_0)$, $r=1,2,\ldots$, ${}^{t}c_{u2k2n}=\int_{t_0}^{t}c_{2k2n}\,dt$ – остаточный член от приближения $u(x,t)$ полиномом ${}^{t}P_{u2k2nr}(x,t)$ на отдельно взятой итерации. С учетом (4.25) $\left|\int_{t_0}^{t}c_{2k2n}\,dt\right|\le C2^{-k(2n+1)}h^{2n+1}(t-t_0)$, согласно предположению II $\left|{}^{t}c_{u2k2n}\right|\le C2^{-k(2n+1)}h^{2n+1}\times 2^{-k}T$. Без явного выражения $\Psi^{ij}_{\partial u\ 2k2n(r-1)}(z,w)$ этот же процесс запишется в виде

$$\begin{aligned}{}^{t}P_{u2k2n\,r}(x,t)={}^{t}P_{u2k2n\,r}(x,t_0)+\int_{t_0}^{t}\left(f(x,t)-a(x,t)\,\frac{\partial\,{}^{t}P_{u2k2n\,(r-1)}(x,t)}{\partial x}\right)dt+\\ +{}^{t}c_{u2k2n},\ (x,t)\in G_{ij},\ t_0=t_0(i,j),\end{aligned}\qquad(4.26)$$

где $r=1,2,\ldots$, коэффициенты $\frac{\partial\,{}^{t}P_{u2k2n\,(r-1)}(x,t)}{\partial x}$ связаны с коэффициентами ${}^{t}P_{u2k2n\,(r-1)}(x,t)$ согласно (4.21), остаточный член оценивается из соотношения

$$\left|{}^{t}c_{u2k2n}\right|\le\widetilde{C}2^{-k(2n+2)}h^{2n+1},\ \widetilde{C}=CT,\ \widetilde{C}=\text{const}\ \forall\ (x,t)\in G_{ij},\ \forall\ G_{ij}\in G.\qquad(4.27)$$

Пусть прямоугольник G_{ij} произвольно фиксирован и уравнение (4.24) рассматривается в нем с теми же начальными условиями на границе, с которыми выполняются итерации (4.26). В этом случае решение не является

точным, что отмечается чертой сверху, аналогично отмечается приближающий его полином, из (4.26) следует

$$
\begin{aligned}
{}^{t}\overline{P}_{u2k2nr}(x,t) &= {}^{t}\overline{P}_{u2k2nr}(x,t_0) + \\
&+ \int_{t_0}^{t} \left(f(x,t) - a(x,t)\ \frac{\partial\, {}^{t}\overline{P}_{u2k2n\,(r-1)}(x,t)}{\partial x} + c_{2k2n} \right) dt,\ (x,t) \in G_{ij},
\end{aligned}
\tag{4.28}
$$

где использовано ${}^{t}c_{u2k2n} = \int_{t_0}^{t} c_{2k2n}$, $\overline{u}(x,t) \approx u(x,t)$, $\overline{u}(x,t_0) = \varphi_{ij}(x,t_0)$,

$\varphi_{ij}(x,t_0) = {}^{t}\overline{P}_{u2k2nr}(x,t_0)$, $r = 1, 2, \ldots$.

В предположениях I, II $u(x,t)$ удовлетворяет соотношению

$$
\begin{aligned}
&\forall\, \varepsilon_0 > 0\ \exists\, \Delta > 0,\ \Delta = \Delta(\varepsilon_0),\ \Delta = \text{const}: \\
&\forall\, \Delta x,\ 0 < |\Delta x| \le \Delta,\ \left| \frac{\partial u(x,t)}{\partial x} - \frac{u(x+\Delta x,t) - u(x,t)}{\Delta x} \right| \le \varepsilon_0 \\
&\forall\ (x,t) \in G,\ (x+\Delta x,t) \in G_{\Delta},
\end{aligned}
\tag{4.29}
$$

в частности, (4.29) выполняется в случае $\Delta x = \Delta$. В самом деле, по теореме о среднем,

$$
\frac{\partial u(x,t)}{\partial x} - \frac{u(x+\Delta x,t) - u(x,t)}{\Delta x} = \frac{\partial u(x,t)}{\partial x} - \frac{\partial u(x,t)}{\partial x}\Big|_{x=\xi},\ \xi = x + \lambda \Delta x,\ |\lambda| < 1.
$$

Повторное применение теоремы влечет

$$
\frac{\partial u(x,t)}{\partial x} - \frac{\partial u(x,t)}{\partial x}\Big|_{x=\xi} = \frac{\partial^2 u(x,t)}{\partial x^2}\Big|_{x=\bar{\xi}} \times \overline{\Delta x},\ \bar{\xi} = x + \bar{\lambda}\ \overline{\Delta x},\ |\bar{\lambda}| < 1,\ |\overline{\Delta x}| < |\Delta x|.
$$

Отсюда

$$
\left| \frac{\partial u(x,t)}{\partial x} - \frac{u(x+\Delta x,t) - u(x,t)}{\Delta x} \right| \le \max_{G} \left| \frac{\partial^2 u(x,t)}{\partial x^2} \right| |\Delta x|.
$$

Область G замкнута, поэтому $\max\limits_{G}\left|\dfrac{\partial^2 u(x,t)}{\partial x^2}\right|\le c$, $c=\text{const}$. Очевидно, $\forall\,\varepsilon_0>0$ достаточно взять любое $\Delta=\text{const}$, $0<\Delta\le c^{-1}\varepsilon_0$, чтобы выполнялось (4.29). Значение Δ в (4.29) можно произвольно уменьшить.

Для исследования сходимости (4.28) рассматривается вспомогательная задача

$$\begin{aligned}&\frac{\partial u_\Delta(x,t)}{\partial t}+a(x,t)\,\frac{u_\Delta(x+\Delta,t)-u_\Delta(x,t)}{\Delta}=f_\Delta(x,u_\Delta,t),\\&u_\Delta(x,0)=\varphi(x),\ (x,t)\in G,\ (x+\Delta,t)\in G_\Delta,\end{aligned}\tag{4.30}$$

где $f_\Delta(x,u_\Delta,t)=f(x,t)-a(x,t)\left(\dfrac{\partial u_\Delta(x,t)}{\partial x}-\dfrac{u_\Delta(x+\Delta,t)-u_\Delta(x,t)}{\Delta}\right)$, $f(x,t)$ из (4.23), Δ произвольно выбрано в соответствии с (4.29) и зафиксировано. Задача (4.30) является эквивалентным преобразованием (4.23), в области G решения задач совпадают и одновременно устойчивы к возмущению начальных данных. В предположениях I, II $f_\Delta(x,u_\Delta,t)$ удовлетворяет условию Липшица относительно u_Δ. В условиях (4.30) выполнено $u_\Delta(x,t)\equiv u(x,t)$, $\dfrac{\partial u_\Delta(x,t)}{\partial x}\equiv\dfrac{\partial u(x,t)}{\partial x}$, $\dfrac{\partial u_\Delta(x,t)}{\partial t}\equiv\dfrac{\partial u(x,t)}{\partial t}$. В G_{ij} решение задачи (4.30) представимо в виде

$$\begin{aligned}&u_\Delta(x,t)=u_\Delta(x,t_0)+\int_{t_0}^{t}\left(f_\Delta(x,u_\Delta,t)-a(x,t)\,\frac{u_\Delta(x+\Delta,t)-u_\Delta(x,t)}{\Delta}\right)dt,\\&t_0=t_0(i,j),\ u_\Delta(x,t_0)=\varphi_{ij}(x,t_0),\end{aligned}\tag{4.31}$$

где $(x,t)\in G_{ij}$, $(x+\Delta,t)\in G_{\Delta ij}$. Приближающий $u_\Delta(x,t)$ полином будет обозначаться ${}^{t}P_{u_\Delta 2k2nr}(x,t)$. Полином, приближающий $f_\Delta(x,u_\Delta,t)-a(x,t)\,\dfrac{u_\Delta(x+\Delta,t)-u_\Delta(x,t)}{\Delta}$, обозначается $\Psi^{ij}_{\Delta u\,2k2n}(z,w)$, где z и w из (4.20). Значения остаточных членов интерполяции изменятся, но их

обозначения сохраняются, что не должно приводить к недоразумениям. В этих обозначениях решение (4.31) приближает последовательность

$$
\begin{aligned}
{}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) = {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t_0) + \int_{t_0}^{t} \Bigg(& f_\Delta(x, {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t) - \\
& - a(x,t)\, \frac{{}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x+\Delta,t) - {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\Delta} + c_{2k2n} \Bigg) dt,
\end{aligned}
\tag{4.32}
$$

где $\overline{P}_{u_\Delta 2k2nr}(x,t_0) = \overline{u}_\Delta(x,t_0)$, $r = 0, 1, \ldots$, ввиду неточности начальных данных решение отмечается чертой. Соотношение (4.31) перейдет в соотношение

$$
\begin{aligned}
& \overline{u}_\Delta(x,t) = \overline{u}_\Delta(x,t_0) + \int_{t_0}^{t} \left(f_\Delta(x, \overline{u}_\Delta, t) - a(x,t)\, \frac{\overline{u}_\Delta(x+\Delta,t) - \overline{u}_\Delta(x,t)}{\Delta} \right) dt, \\
& \overline{u}_\Delta(x,t_0) = \varphi_{ij}(x,t_0),
\end{aligned}
\tag{4.33}
$$

где $(x,t) \in G_{ij}$, $(x+\Delta, t) \in G_{\Delta ij}$, $t_0 = t_0(i,j)$, $\varphi_{ij}(x,t_0) = \overline{P}_{u_\Delta 2k2nr}(x,t_0)$.

Всюду ниже значение Δ в (4.30) – (4.33) рассматривается как параметр, выбор которого в границах условия (4.29) позволит оценить сходимость последовательности (4.28) на основе оценки сходимости последовательности (4.32).

Из (4.32), (4.33)

$$
\begin{aligned}
\overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) = \int_{t_0}^{t} \Bigg(& f_\Delta(x, \overline{u}_\Delta, t) - f_\Delta(x, {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t) - \\
& - a(x,t) \left(\frac{\overline{u}_\Delta(x+\Delta,t) - {}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x+\Delta,t)}{\Delta} - \frac{\overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x,t)}{\Delta} \right) - c_{2k2n} \Bigg) dt.
\end{aligned}
$$

Замечание 4.2. С учетом постоянного числа подобластей 2^{2k}, $k = \text{const}$, не умаляя общности, можно считать, что к $f_\Delta(x, \overline{u}_\Delta, t)$ для всех $\overline{u}_\Delta$ из области $\widetilde{R}: \{ \left| \overline{u}_\Delta(x,t) - \overline{u}_\Delta(x,t_0) \right| \le \widetilde{b},\ t_0 = t_0(i,j),\ (x,t) \in G_{ij},\ (x+\Delta, t) \in G_{\Delta ij} \}$ применимо условие Липшица при некотором достаточно большом значении $\widetilde{b} = \text{const}$. Последовательность ${}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)$ не выводит из $\widetilde{R}$ при таком определении $\widetilde{b}$

по следующим причинам. При $r=1$ полином ${}^{t}\overline{P}_{u_\Delta 2k2n0}(x,t)$ отличается от $\overline{u}_\Delta$ не более, чем на остаточный член однократного интерполирования ${}^{t}c_{u2k2n}$, под интегралом можно применить условие Липшица при этом r:

$$\left| f_\Delta(x,\overline{u}_\Delta(x,t),t) - f_\Delta(x,{}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),t) \right| \le \widetilde{L} \left| \overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right|,$$

$$\widetilde{L} = \text{const} \;\; \forall (x,t) \in G_{ij}, (x+\Delta, t) \in G_{\Delta ij}.$$

Тогда $\left| \overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) \right| \le \widetilde{b}$, $r=1$, при некотором $\widetilde{b} = \text{const}$, таком, что $\left| {}^{t}c_{u2k2n} \right| \le \widetilde{b}$. Дальнейшие рассуждения строятся по индукции. Именно, с повторением данного приема для $r \ge 1$ ниже доказывается сходимость ${}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t)$ к $\overline{u}_\Delta(x,t)$ при $r \to \infty$. В частности, полином данной последовательности не будет выводить из $\widetilde{R}$ $\forall r \ge 1$, поэтому к $f_\Delta(x,{}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),t)$ применимо условие Липшица. В качестве константы Липшица $\widetilde{L}$ в дальнейшем принят ее максимум по всем подобластям из (4.1) – (4.3).

Таким образом, можно предположить, что для некоторого $r \ge 1$ выполнено –

$$\forall\, (x,t) \in G_{ij}, (x+\Delta,t) \in G_{\Delta ij}:$$

$$\left| \overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) \right| \le \int_{t_0}^{t} \left(\widetilde{L} \left| \overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right| + \right. \tag{4.34}$$

$$\left. + 2M\Delta^{-1} \max_{G_{ij}} \left| \overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right| + \max_{G} \left| c_{2k2n} \right| \right) dt.$$

В (4.34) и ниже $M = \max_{G} \left| a(x,t) \right|$, $M = \text{const}$, $\widetilde{L}$– отмеченное значение константы. Отсюда

$$\forall (x,t) \in G_{ij}, (x+\Delta,t) \in G_{\Delta ij}: \left| \overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) \right| \le$$

$$\le \int_{t_0}^{t} \left((\widetilde{L} + 2M\Delta^{-1}) \max_{G_{ij}} \left| \overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right| + \max_{G} \left| c_{2k2n} \right| \right) dt,$$

и

$$\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2nr}(x,t)\right|\le$$
$$\le\int_{t_0}^{t}\left((\widetilde{L}+2M\Delta^{-1})\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2n\,(r-1)}(x,t)\right|+\max_{G}\left|c_{2k2n}\right|\right)dt. \tag{4.35}$$

Сначала рассматривается случай, когда в (4.35) погрешность однократного интерполирования не превосходит погрешности $r-1$ итераций с некоторым постоянным коэффициентом, точнее,

$$\exists Q>0,\ Q=\text{const}:$$
$$0<\max_{G}\left|c_{2k2n}\right|\le Q\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2n\,(r-1)}(x,t)\right|,\ r=1,2,\ldots \tag{4.36}$$

В (4.36) значение Q можно произвольно увеличить, и для $\forall\varepsilon_0$ из (4.29) оно выбирается так, чтобы

$$Q^{-1}\max_{G}\left|c_{2k2n}\right|\le\varepsilon_0. \tag{4.37}$$

В дальнейшем исследуется случай, когда (4.36) нарушается: для некоторого $r_0>r$ окажется выполненным соотношение

$$\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2n\,(r_0-1)}(x,t)\right|<Q^{-1}\max_{G}\left|c_{2k2n}\right|,\ r_0\ge r+1, \tag{4.38}$$

при этом в силу (4.38) и (4.37) нарушение (4.36) влечет

$$\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2n\,(r_0-1)}(x,t)\right|<\varepsilon_0.$$

Пусть сначала (4.36) не нарушено, в этом случае согласно (4.35)

$$\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2nr}(x,t)\right|\le\int_{t_0}^{t}\Bigg((\widetilde{L}+2M\Delta^{-1})\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2n\,(r-1)}(x,t)\right|+$$
$$+Q\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2n\,(r-1)}(x,t)\right|\Bigg)dt. \tag{4.39}$$

Не умаляя общности, можно считать $\Delta\le1$, тогда

$$\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2nr}(x,t)\right|\le$$

$$\le N\int_{t_0}^{t}\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2n(r-1)}(x,t)\right|dt\ ,\ N=(\widetilde{L}+2M+Q)\Delta^{-1}.$$

Если обозначить $\varepsilon_{k\ell}=\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}2k2n\ell}(x,t)\right|$, то неравенство примет вид

$$\varepsilon_{k\ell}\le N\int_{t_0}^{t}\varepsilon_{k(\ell-1)}dt,\ \ell=1,2,\ldots, \tag{4.40}$$

где $\varepsilon_{k0}=\max_{G_{ij}}\left|\varphi_{ij}(x,t_0)-{}^{t}\overline{P}_{u_{\Delta}2k2n0}(x,t_0)\right|\le\max_{G_{ij}}\left|{}^{t}c_{u2k2n}\right|$, $\varphi_{ij}(x,t_0)$ – функция на смежной с G_{ij} границе $G_{i(j-1)}$, $\varphi_{ij}(x,t_0)={}^{t}\overline{P}_{u_{\Delta}2k2n\overline{\ell}}(x,t)$, $\overline{\ell}$ – номер заключительной итерации, выполнявшейся в $G_{i(j-1)}$. Из (4.40) и (4.27) $\varepsilon_{k1}\le\widetilde{C}2^{-k(2n+2)}h^{2n+1}N\int_{t_0}^{t}dt$, или, $\varepsilon_{k1}\le\widetilde{c}_{kh}N(t-t_0)$, здесь и ниже $\widetilde{c}_{kh}=\widetilde{C}2^{-k(2n+2)}h^{2n+1}$. Тогда $\varepsilon_{k2}\le\widetilde{c}_{kh}\dfrac{N^2(t-t_0)^2}{2}$. По индукции $\varepsilon_{k\ell}\le\widetilde{c}_{kh}\dfrac{N^{\ell}(t-t_0)^{\ell}}{\ell!}$, $\ell=3,4,\ldots$ Согласно предположению I, с учетом размеров G_{ij}, верно неравенство $t-t_0\le 2^{-k}T$, в результате

$$N\int_{t_0}^{t}\varepsilon_{k(\ell-1)}dt\le\widetilde{c}_{kh}\frac{(2^{-k}NT)^{\ell}}{\ell!} \tag{4.41}$$

и

$$\varepsilon_{k\ell}\le\widetilde{c}_{kh}\frac{(2^{-k}NT)^{\ell}}{\ell!},\ \ell=1,2,\ldots \tag{4.42}$$

В (4.42) $(2^{-k}NT)^{\ell}/\ell!\to 0$, $\ell\to\infty$, поэтому $\varepsilon_{k\ell}\to 0$, $\ell\to\infty$. Отсюда следует, что (4.36) необходимо окажется нарушенным, –

$$\exists L = L(i,\ j,\ k,\ Q,\ \Delta,\ \varepsilon_0):$$
$$\widetilde{c}_{kh}\frac{(2^{-k}NT)^L}{L!} < Q^{-1}\max_G\left|c_{2k2n}\right|,\ \varepsilon_{kL} < Q^{-1}\max_G\left|c_{2k2n}\right|,\ \varepsilon_{kL} < \varepsilon_0. \tag{4.43}$$

Пусть (4.43) и соответственно (4.38) впервые выполнились при $L = r_0 - 1 = r$. Для индекса $r-1$ еще сохранялось (4.36), не выполнялось (4.38) и оставалось верным (4.39), поэтому при $\ell = 0$

$$\begin{aligned}&\max_{G_{ij}}\left|\ \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r+\ell)}(x,t)\right| < \\ &< Q^{-1}\max_G\left|c_{2k2n}\right| \le \max_{G_{ij}}\left|\ \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r-1)}(x,t)\right|\end{aligned} \tag{4.44}$$

и

$$\max_{G_{ij}}\left|\ \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r+\ell)}(x,t)\right| < \max_{G_{ij}}\left|\ \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n(r-1)}(x,t)\right|. \tag{4.45}$$

Если предположить, что (4.44), (4.45) выполняются для некоторого $\ell \ge 0$, то для оценки погрешности $(r+\ell+1)$-й итерации можно воспользоваться соотношением (4.35) при соответственных индексах:

$$\begin{aligned}&\max_{G_{ij}}\left|\overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r+\ell+1)}(x,t)\right| \le \\ &\le \int_{t_0}^{t}\left((\widetilde{L} + 2M\Delta^{-1})\max_{G_{ij}}\left|\ \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r+\ell)}(x,t)\right| + \max_G\left|c_{2k2n}\right|\right)dt.\end{aligned}$$

С учетом (4.45) и (4.44)

$$\begin{aligned}&\max_{G_{ij}}\left|\overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r+\ell+1)}(x,t)\right| < \\ &< \int_{t_0}^{t}\left((\widetilde{L} + 2M\Delta^{-1})\max_{G_{ij}}\left|\overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r-1)}(x,t)\right| + Q\max_{G_{ij}}\left|\overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r-1)}(x,t)\right|\right)dt,\end{aligned}$$

или, в прежнем обозначении,

$$\max_{G_{ij}}\left|\overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r+\ell+1)}(x,t)\right| < N\int_{t_0}^{t}\left(\max_{G_{ij}}\left|\overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r-1)}(x,t)\right|\right)dt. \tag{4.46}$$

Согласно (4.41) правая часть (4.46) не превосходит $\widetilde{c}_{kh}\dfrac{(2^{-k}NT)^L}{L!}$ при $L=r$. С учетом (4.43)

$$\max_{G_{ij}}\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r+\ell+1)}(x,t)\right|<\widetilde{c}_{kh}\frac{(2^{-k}NT)^L}{L!}<\max_{G}\left|c_{2k2n}\right|Q^{-1}.$$

Отсюда

$$\max_{G_{ij}}\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r+\ell+1)}(x,t)\right|<Q^{-1}\max_{G}\left|c_{2k2n}\right|\le\max_{G_{ij}}\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|,$$

и соотношения (4.44), (4.45) сохраняются при замене ℓ на $\ell+1$. В силу индукции и согласно (4.37)

$$\exists\, r=r(i,j,k,Q,\Delta,\varepsilon_0):\ \max_{G_{ij}}\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\ (r+\ell)}(x,t)\right|<\varepsilon_0\ \ \forall\ell=0,1,\ldots \quad (4.47)$$

Из изложенного вытекает

Лемма 4.2. Пусть в (4.1) – (4.3) зафиксировано 2^{2k} подобластей и произвольно выбрана подобласть G_{ij}. При условии, что $\overline{u}_\Delta(x,t)$ из (4.33) рассматривается в G_{ij}, $(x,t)\in G_{ij}$, $(x+\Delta,t)\in G_{\Delta ij}$, с теми же начальными условиями на границе с $G_{i(j-1)}$, с которыми выполняются итерации (4.32), $\overline{P}_{u_\Delta 2k2nr}(x,t_0)=\overline{u}_\Delta(x,t_0)$, $r=0,1,\ldots$, $t_0=t_0(i,j)$, последовательность (4.32) равномерно $\forall\,(x,t)\in G_{ij}$ сходится к $\overline{u}_\Delta(x,t)$. До тех пор, пока не нарушается соотношение (4.36), скорость сходимости оценивается из (4.42). В любом случае имеет место (4.47).

Попутно показано, что последовательность ${}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)$ не выводит из области $\widetilde{R}$, как и предполагалось в замечании 4.2.

Рассматриваемое приближение непрерывно, следовательно, равномерно непрерывно в замкнутой подобласти G_{ij}. Поскольку для (4.33) и для (4.32) в G_{ij} выполнено $\overline{u}_\Delta(x,t_0)\big|_{(x,t)\in G_{ij},\ t_0=t_0(i,j)}={}^t\overline{P}_{u_\Delta 2k2n\,\overline{\ell}}(x,t)\Big|_{(x,t)\in G_{i(j-1)},\ t=t_0(i,j)}$, где $\overline{\ell}$ –

номер заключительной итерации, выполнявшейся в $G_{i\,(j-1)}$, то непрерывность и равномерная непрерывность приближения (4.32) сохраняется при переходе от $G_{i\,(j-1)}$ к $G_{i\,j}$ и, таким образом, во всем временном слое $\bigcup_{j=0}^{2^k-1} G_{i\,j}$ для каждого $i = \text{const}$.

Отсюда вытекает

Следствие 4.1. Кусочно-интерполяционное приближение решения задачи (4.30) с итерационным уточнением в каждой подобласти $G_{i\,j}$ при любом количестве итераций является равномерно непрерывной функцией. В частности, при $k = 0$ приближение равномерно непрерывно в G, при $k > 0$ оно кусочно-непрерывно в G и сохраняет равномерную непрерывность в слое подобластей $\bigcup_{j=0}^{2^k-1} G_{i\,j}$ для каждого $i = \text{const}$.

Для $\forall \varepsilon > 0$ (4.47) сохранится, если взять $\varepsilon_0 \le 2^{-k-1}\varepsilon$, априори в (4.29) взять соответственное значение Δ и в (4.47) указать зависящее от них r:

$$\begin{gathered}\exists \Delta_1 = \text{const}, \forall \Delta,\ 0 < \Delta \le \Delta_1,\ \Delta = \text{const},\ \exists\, r = r(i,\ j,\ k,\ Q,\ \Delta,\ \varepsilon): \\ \max_{G_{ij}} \left| \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\ (r+\ell)}(x,t) \right| < 2^{-k-1}\varepsilon\ \forall \ell = 0,1,\ldots\end{gathered} \tag{4.48}$$

Из того, что на выходе из $G_{i\,j}$ полином ${}^t\overline{P}_{u_\Delta 2k2n\ (r+\bar{\ell})}(x,t)$ задает начальные условия $\overline{u}_\Delta(x,t_0) = \varphi_{i\,(j+1)}(x,t_0)$, $t_0 = t_0(i, j+1)$, в $G_{i(j+1)}$, и при этом выполняется (4.48), вытекает оценка изменения начальных условий при переходе от одной подобласти к другой.

Следствие 4.2. Вусловиях леммы 4.2 $\forall \varepsilon > 0$ найдется $\Delta_1 = \text{const}$, такое что $\forall \Delta$ из (4.29), $0 < \Delta \le \Delta_1$, максимальное отклонение последовательности (4.32) в $G_{i\,j}$ от $\overline{u}_\Delta(x,t)$ оценивается из (4.48). Этого же значения не превысит максимальное отклонение начальных условий в $G_{i(j+1)}$ от тех начальных условий $\varphi_{i\,(j+1)}(x, t_0)$, которые задавались бы при переходе из $G_{i\,j}$ в $G_{i(j+1)}$

точным решением, взятым с начальными условиями из G_{ij} в виде $\varphi_{ij}(x,t_0) = {}^{t}\overline{P}_{u_\Delta 2k2n\overline{\ell}}(x,t)\Big|_{(x,t)\in G_{i(j-1)},\, t=t_0(i,j)}$.

Устойчивость решения задачи (4.30) сохраняется в каждой подобласти и означает, что $\forall \varepsilon_0 > 0\ \exists \delta > 0$, такое, что если $\left|\widetilde{u}_\Delta(x,t_0) - \overline{u}_\Delta(x,t_0)\right| < \delta$, $t_0 = t_0(i,j)$, то $\left|\widetilde{u}_\Delta(x,t) - \overline{u}_\Delta(x,t)\right| \le \varepsilon_0\ \forall\,(x,t)\in G_{ij}$. В качестве невозмущенного решения в G_{ij} рассматривается $\overline{u}_\Delta(x,t)$. Его возмущение – точное (не получающееся вследствие приближения) решение в этой подобласти $\widetilde{u}_\Delta(x,t)$, соответственное возмущенным начальным условиям $\widetilde{u}_\Delta(x,t_0)$. По следствию 4.2 максимальное отклонение начальных данных при переходе от G_{ij} к $G_{i(j+1)}$ оценивается из (4.48). Аналогично, при переходе от $G_{i(j-1)}$ к G_{ij}, где это достигается за счет выбора Δ и r в $G_{i(j-1)}$. Отсюда для $\varepsilon_0 \le 2^{-k-1}\varepsilon$ найдется $\delta > 0$, и такие Δ, r для последовательности (4.32) в $G_{i(j-1)}$, что выполняются соотношения:

$$\begin{gathered}\exists\,\Delta_0 = \text{const}, \forall\,\Delta,\ 0 < \Delta \le \Delta_0,\ \Delta = \text{const}, \exists\,\widetilde{r} = r(i, j-1, k, Q, \Delta, \delta):\\ \max_{G_{i(j-1)}} \left|\overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t)\right| \le \delta\ \ \forall r \ge \widetilde{r},\end{gathered} \tag{4.49}$$

и

$$\max_{G_{ij}} \left|\widetilde{u}_\Delta(x,t_0) - \overline{u}_\Delta(x,t_0)\right| \le \delta, t_0 = t_0(i,j),$$

поэтому, в силу устойчивости,

$$\max_{G_{ij}} \left|\widetilde{u}_\Delta(x,t) - \overline{u}_\Delta(x,t)\right| \le 2^{-k-1}\varepsilon. \tag{4.50}$$

Пусть $\overline{r} = \max(r,\widetilde{r})$, $\overline{\Delta} = \min(\Delta_1, \Delta_0)$, где r и Δ_1 из (4.48), $\widetilde{r}$ и Δ_0 из (4.49) . Из (4.48) и (4.50)

$$\max_{G_{ij}} \left|\widetilde{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t)\right| \le 2^{-k}\varepsilon \quad \forall r \ge \overline{r},\ \Delta = \text{const},\ 0 < \Delta \le \overline{\Delta}. \tag{4.51}$$

Поскольку число подобластей 2^{2k} зафиксировано, можно выбрать наименьшее по всем i, j значение $\overline{\Delta} = \Delta_{\min}$, и, при этом значении $\overline{\Delta}$, наибольший по всем i, j номер $r_{\max}$, которые обеспечат выполнение (4.48), следствия 4.2, соотношений (4.49), а также (4.50), (4.51) одновременно во всех подобластях G_{ij}:

$$\max_{\forall i,j:\, G_{ij}\in G} \left| \overline{u}_{\Delta}(x,t) - {}^{t}\overline{P}_{u_{\Delta}\, 2k\, 2n\, r}(x,t) \right| \le 2^{-k}\varepsilon \quad \forall r \ge r_{\max},\ \Delta = \text{const},\ 0 < \Delta \le \Delta_{\min}, \quad (4.52)$$

где $\Delta_{\min} = \Delta_{\min}(k)$, $\Delta_{\min} = \text{const}$, $r_{\max} = \max\limits_{\forall i,j:\, G_{ij}\in G} r(i, j, k, Q, \Delta, \varepsilon)$. Суммой отклонений во всех подобластях слоя $\bigcup\limits_{j=0}^{2^k-1} G_{ij}$, $i = \text{const}$, можно оценить максимальное отклонение кусочно-интерполяционного приближения с итерационным уточнением от точного невозмущенного решения задачи (4.30) во всем этом слое:

$$\max_{\bigcup_{j=0}^{2^k-1} G_{ij},\ i\in\overline{0,2^k-1}} \left| u_{\Delta}(x,t) - {}^{t}\overline{P}_{u_{\Delta}\, 2k\, 2n\ r}(x,t) \ \Big|_{(x,t)\in G_{ij}} \right| \le$$

$$\le \sum_{k=0}^{2^k-1} 2^{-k}\varepsilon \quad \forall r \ge r_{\max},\ \Delta = \text{const},\ 0 < \Delta \le \Delta_{\min},$$

или,

$$\max_{\bigcup_{j=0}^{2^k-1} G_{ij},\ i\in\overline{0,2^k-1}} \left| u_{\Delta}(x,t) - {}^{t}\overline{P}_{u_{\Delta}\, 2k\, 2n\ r}(x,t) \ \Big|_{(x,t)\in G_{ij}} \right| \le \varepsilon$$

$$\forall r \ge r_{\max},\ \Delta = \text{const},\ 0 < \Delta \le \Delta_{\min}.$$

Согласно выбору максимума в (4.52), использованному в этих оценках, они сохраняются для любого слоя $\bigcup\limits_{j=0}^{2^k-1} G_{ij}$ с постоянным $i \in \overline{0,2^k-1}$ из области G.

Отсюда, а также из того, что по построению вспомогательной задачи (4.30) при любом Δ из (4.29) $u_{\Delta}(x,t) \equiv u(x,t)$ $\forall\, (x,t) \in G_{ij}$, где $u(x,t)$ – решение уравнения (4.23), вытекает

Лемма 4.3. В условиях леммы 4.2 кусочно-интерполяционное приближение с итерационным уточнением (4.32) решения задачи (4.30) равномерно сходится в области G к решению $u(x,t)$ задачи (4.23). При этом

$$\begin{aligned}&\forall \varepsilon > 0 \ \exists \Delta_{\min} = \text{const}, \forall \Delta, \ 0 < \Delta \le \Delta_{\min}, \Delta = \text{const},\\ &\exists r_{\max} = \max_{\forall i,j:\, G_{ij} \in G} r(i, j, k, Q, \Delta, \varepsilon):\\ &\qquad \max_{\forall i,j:\, G_{ij} \in G} \left| u(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) \ \right|_{(x,t) \in G_{ij}} \Bigg| < \varepsilon \ \ \forall r \ge r_{\max}.\end{aligned} \tag{4.53}$$

Относительно равномерной и кусочной непрерывности последовательных приближений без изменений повторяется утверждение следствия 4.1, с той оговоркой, что оно относится не только к решению задачи (4.30), но также и к решению задачи (4.23).

Замечание 4.3. Условия и утверждение леммы 4.3 сохранятся без существенных изменений, если кусочно-интерполяционное приближение решения задачи (4.30) с итерационным уточнением применить к решению задачи

$$\frac{\partial u_\Delta(x,t)}{\partial t} + a(x,t)\,\frac{u_\Delta(x+\Delta,t) - u_\Delta(x,t)}{\Delta} = f(x,t), \ \ u_\Delta(x,0) = \varphi(x), \tag{4.54}$$

в виде

$$\begin{aligned}&{}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) = {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t_0) + \\ &+ \int_{t_0}^{t} \left(f(x,t) - a(x,t)\,\frac{{}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x+\Delta,t) - {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\Delta} \right) dt + {}^{t}c_{u2k2n}.\end{aligned} \tag{4.55}$$

Аналогично предыдущему, доказывается сходимость (4.55) к решению $u_\Delta(x,t)$ задачи (4.54), а также приближение этого решения с точностью до $2^{-1}\varepsilon$. Из того, что

$$\begin{aligned}&\left| u_\Delta(x,t) - u(x,t) \right| \le \int_{t_0}^{t} \left| a(x,t) \right| \left| \frac{\partial u(x,t)}{\partial x} - \frac{u(x+\Delta,t) - u(x,t)}{\Delta} \right| dt\\ &\forall \ (x,t) \in G_{ij}, \ (x+\Delta,t) \in G_{\Delta ij},\end{aligned}$$

где $u(x,t)$ – решение задачи (4.23), с учетом (4.29), следует $\max\limits_{G}\left|u_{\Delta}(x,t)-u(x,t)\right|\le M\,T\,\varepsilon_0$. При выборе $\varepsilon_0 \le 2^{-1}(MT)^{-1}\varepsilon$ получится $\left|u_{\Delta}(x,t)-u(x,t)\right|\le 2^{-1}\varepsilon$. В результате (4.55) будет приближать $u(x,t)$ с точностью до ε.

Тем не менее, сходимость (4.32) к решению задачи (4.30) означает сходимость непосредственно к решению задачи (4.23), что используется ниже.

4.4. Сходимость метода. Для перехода от (4.32) к (4.28), в условиях леммы 4.3 и с учетом (4.53), предварительно оценивается разность

$$D_{r-1}=\frac{\partial\,{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n\,(r-1)}(x,t)}{\partial x}-\frac{{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n\,(r-1)}(x+\Delta,t)-{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n\,(r-1)}(x,t)}{\Delta}. \qquad (4.56)$$

С применением теоремы о среднем,

$$\left|\,D_{r-1}\,\right|=\left|\frac{\partial\,{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n\,(r-1)}(x,t)}{\partial x}-\frac{\partial\,{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n\,(r-1)}(x,t)}{\partial x}\Big|_{x=\widetilde{\xi}}\right|,\quad \widetilde{\xi}=x+\widetilde{\alpha}\,\Delta,\ \ 0<\left|\,\widetilde{\alpha}\,\right|<1. \qquad (4.57)$$

Повторное применение теоремы влечет

$$\left|\,D_{r-1}\,\right|=\left|\frac{\partial^2\,{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n(r-1)}(x,t)}{\partial x^2}\Big|_{x=\bar{\gamma}}\right|\times\left|\,\overline{\Delta x}\,\right|,\ \left|\,\overline{\Delta x}\,\right|<\left|\,\widetilde{\alpha}\,\right|\Delta,\ \ \bar{\gamma}=x+\bar{\beta}\,\overline{\Delta x},\ \ 0<\left|\,\bar{\beta}\,\right|<1. \qquad (4.58)$$

По построению, $\dfrac{\partial^2\,{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n\,(r-1)}(x,t)}{\partial x^2}\Big|_{x=\bar{\gamma}}=\dfrac{1}{h_x}\dfrac{\partial\,\overline{\Psi}^{\,ij}_{\Delta u\,2k\,2n\,(r-1)}(z,w)}{\partial z}\Big|_{x=\bar{\gamma}}$, где z, w из (4.20), отсюда

$$\left|\frac{\partial^2\,{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n(r-1)}(x,t)}{\partial x^2}\Big|_{x=\bar{\gamma}}\right|\le\max_{G_{ij}}\frac{1}{h_x}\left|\frac{\partial\,\overline{\Psi}^{\,ij}_{\Delta u\,2k\,2n\,(r-1)}(z,w)}{\partial z}\right|. \qquad (4.59)$$

Правая часть (4.59) ограничена в силу следующих причин. Последовательность (4.32) сходится к решению $u(x,t)$ задачи (4.23), и выполнено (4.53). Функция $u(x,t)$ ограничена в G (в G_{Δ}), в частности в G_{ij}, поэтому при $\forall\, r\ge r_{\max}+1$

значения ${}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x,t)$ также ограничены в G_{ij}. Поскольку $r_{\max}=\text{const}$, то и вся последовательность ${}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x,t)$ ограничена в G_{ij} $\forall$ $r=1,2,\ldots$ В G, G_Δ и G_{ij} ограничена функция $f_\Delta(x,u_\Delta,t)$. Как следствие, при замене $u(x+\Delta,t)$, $u(x,t)$ в выражении $f_\Delta(x,u_\Delta,t)-a(x,t)\dfrac{u_\Delta(x+\Delta,t)-u_\Delta(x,t)}{\Delta}$ из (4.30) на полиномы ${}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x+\Delta,t)$, ${}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x,t)$ последовательность данных выражений останется ограниченной в G_{ij}. Элементы именно этой последовательности интерполируются полиномами $\overline{\Psi}^{ij}_{\Delta u\,2k2n(r-1)}(z,w)$:

$$\overline{\Psi}^{ij}_{\Delta u\,2k2n(r-1)}(z,w)\approx$$
$$\approx a(x,t)\frac{{}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x+\Delta,t)-{}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x,t)}{\Delta}-f_\Delta(x,{}^{t}\overline{P}_{u_\Delta 2k2n(r-1)}(x,t),t).$$

В процессе итерационного уточнения полином $\overline{\Psi}^{ij}_{\Delta u\,2k2n(r-1)}(z,w)$ не меняет степень и расстояние между узлами, при этом множество всех узловых значений оказывается ограниченным в продолжение всего итерационного процесса. Структура полинома аналогична (4.7), таким образом, ограничены его коэффициенты. В то же время коэффициенты стоят перед не меняющимися частями полинома, которые непрерывны в G_{ij}. Отсюда полином $\overline{\Psi}^{ij}_{\Delta u\,2k2n(r-1)}(z,w)$ ограничен в G_{ij}. Взятие производной от полинома с рассматриваемыми свойствами сохраняет ограниченность коэффициентов и непрерывность выражения полинома, представляющего производную. В результате $\max\limits_{G_{ij}}\dfrac{1}{h_x}\left|\dfrac{\partial\overline{\Psi}^{ij}_{\Delta u\,2k2n(r-1)}(z,w)}{\partial z}\right|\le c_0$, $c_0=\text{const}$. С учетом $n\le N_0$, $N_0=\text{const}$, и $k=\text{const}$, $\exists C_0=\text{const}:\ \max\limits_{G}\dfrac{1}{h_x}\left|\dfrac{\partial\overline{\Psi}^{ij}_{\Delta u\,2k2n(r-1)}(z,w)}{\partial z}\right|\le C_0$.

Подстановка C_0 в (4.59), затем в (4.58), с учетом (4.57) влечет $\left|D_{r-1}\right|\le C_0\times\Delta$. Отсюда $\forall\varepsilon>0$ при априорном выборе $\Delta\le\varepsilon C_0^{-1}$ будет выполнено

$$\forall \varepsilon > 0 \ \exists \Delta_{\min} = \text{const}, \forall \Delta, 0 < \Delta \le \min(\Delta_{\min}, \varepsilon C_0^{-1}),$$
$$\Delta = \text{const}, \exists R_{\max} = \max_{\forall i,j:\, G_{ij} \in G} r(i, j, k, Q, \Delta, \varepsilon): \quad \max_G \left| D_{r-1} \right| \le \varepsilon \quad \forall r \ge R_{\max}. \tag{4.60}$$

Пусть наряду с (4.32) рассматривается последовательность

$$\begin{aligned} {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t) &= {}^t\overline{P}_{u_\Delta 2k2n}(x,t_0) + \\ &+ \int_{t_0}^{t} \left(f_\Delta(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t) - a(x,t) \frac{\partial\, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\partial x} - c_{u2k2n} \right) dt,\ r = 1, 2, \ldots \end{aligned} \tag{4.61}$$

Из (4.33) и (4.61), при равенстве начальных условий в G_{ij},

$$\begin{aligned} \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2nr}(x,t) = \int_{t_0}^{t} \Bigg(& f_\Delta(x, \overline{u}_\Delta, t) - f_\Delta(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t) - \\ & - a(x,t) \left(\frac{\overline{u}_\Delta(x+\Delta,t) - \overline{u}_\Delta(x,t)}{\Delta} - \frac{\partial\, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\partial x} \right) - c_{2k2n} \Bigg) dt. \end{aligned} \tag{4.62}$$

В обозначении $F_{r-1} = f_\Delta(x, \overline{u}_\Delta, t) - f_\Delta(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t)$ (4.62) эквивалентно

$$\begin{aligned} \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t) = \int_{t_0}^{t} \Big(& F_{r-1} - a(x,t)\Delta^{-1} \Big(\overline{u}_\Delta(x+\Delta,t) - {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x+\Delta,t) - \\ & - \left(\overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right) \Big) + a(x,t) D_{r-1} - c_{2k2n} \Big) dt, \end{aligned}$$

D_{r-1} из (4.56). Применение условия Липшица к F_{r-1} влечет

$$\begin{aligned} \left| \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t) \right| \le \int_{t_0}^{t} \Bigg(& \widetilde{L} \left| \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right| + \\ & + 2M\Delta^{-1} \max_{G_{ij}} \left| \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right| + \left| a(x,t) D_{r-1} - c_{2k2n} \right| \Bigg) dt. \end{aligned} \tag{4.63}$$

Пусть $\overline{c}_{2k2n} = a(x,t) D_{r-1} - c_{2k2n}$. Согласно (4.60) $\left| a(x,t) D_{r-1} \right| \le M\varepsilon \quad \forall r \ge R_{\max}$. Если априори выбрать $\forall \varepsilon > 0$ так, чтобы $M\varepsilon \le \max_G \left| c_{2k2n} \right|$, то при Δ, $\Delta_{\min}$, $R_{\max}$ из (4.60), взятых для данного ε,

$$\max_G \left| \overline{c}_{2k2n} \right| \le 2 \max_G \left| c_{2k2n} \right| \quad \forall r \ge R_{\max}. \tag{4.64}$$

В предположении $\Delta \le 1$ из (4.63) следует $\forall\, (x,t) \in G_{ij}, (x+\Delta,t) \in G_{\Delta ij}$:

$$\left| \overline{u}_\Delta (x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t) \right| \le$$
$$\le \int_{t_0}^{t} \left((\widetilde{L} + 2M\Delta^{-1}) \max_{G_{ij}} \left| \overline{u}_\Delta (x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right| + \max_G \left| \overline{c}_{2k2n} \right| \right) dt$$

и

$$\max_{G_{ij}} \left| \overline{u}_\Delta (x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t) \right| \le$$
$$\le \int_{t_0}^{t} \left((\widetilde{L} + 2M\Delta^{-1}) \max_{G_{ij}} \left| \overline{u}_\Delta (x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t) \right| + \max_G \left| \overline{c}_{2k2n} \right| \right) dt. \tag{4.65}$$

Для (4.65) с точностью до обозначений и постоянных множителей воспроизводятся все рассуждения, преобразования и соотношения от (4.35) до (4.53) включительно. С такой оговоркой рассматриваемые преобразования инвариантны относительно Δ, выбранного в (4.60) и, соответственно, в (4.64). В (4.64) предполагалось $\varepsilon \le M^{-1} \max_G \left| c_{2k2n} \right|$, $\Delta \le \varepsilon C_0^{-1}$, или, $\Delta \le C_0^{-1} M^{-1} \max_G \left| c_{2k2n} \right|$. С этим ограничением получится, –

$$\forall \varepsilon > 0 \;\; \exists\, \Delta_{\min} = \text{const}, \forall \Delta,\; 0 < \Delta \le \min\left(\Delta_{\min},\, C_0^{-1} M^{-1} \max_G \left| c_{2k2n} \right|\right),$$
$$\Delta = \text{const},\; \exists\, R_{\max} = \max_{\forall i,j:\, G_{ij} \in G} r(i,\, j,\, k,\, Q,\, \Delta,\, \varepsilon):$$
$$\max_{\forall i,j:\, G_{ij} \in G} \left| u_\Delta (x,t) - {}^t\overline{P}_{u_\Delta 2k2n\;r}(x,t) \big|_{(x,t) \in G_{ij}} \right| \le \varepsilon \quad \forall r \ge R_{\max}\,, \tag{4.66}$$

где ${}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t)$ из (4.61), $u_\Delta(x,t)$ – решение задачи (4.30). Поскольку $u_\Delta(x,t) \equiv u(x,t) \;\; \forall\, (x,t) \in G$, последовательность (4.61) с оценкой (4.66) приближает решение задачи (4.23). По сравнению с (4.40) – (4.53) изменится коэффициент в аналоге (4.40). Именно, при выполнении оценки ε_{k0} с учетом (4.64) получится $\varepsilon_{k0} = \max_{G_{ij}} \left| \varphi_{ij}(x,t_0) - {}^t\overline{P}_{u_\Delta 2k2n0}(x,t_0) \right| \le 2 \max_{G_{ij}} \left| {}^t c_{u2k2n} \right|$.

Соответственно, $\widetilde{C}$ потребуется заменить на $2\widetilde{C}$. Если обозначить $\widetilde{c}_{kh} = 2\widetilde{C}2^{-k(2n+2)}h^{2n+1}$, то вид дальнейших оценок сохранится. Таким образом, имеет место

Лемма 4.4. В условиях леммы 4.2 кусочно-интерполяционное приближение с итерационным уточнением вида (4.61) решения задачи (4.30) равномерно сходится в области G к решению $u(x,t)$ задачи (4.23). При этом $\forall \varepsilon > 0$ выполняется соотношение (4.66), где $u_\Delta(x,t) \equiv u(x,t)\ \forall\ (x,t) \in G$. Относительно равномерной и кусочной непрерывности последовательных приближений (4.61) без изменений повторяется утверждение леммы 4.3.

В (4.61), в силу леммы 4.4, для Δ из (4.66) $f_\Delta\left(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t\right) \to f_\Delta\left(x, u(x,t), t\right),\ r \to \infty$, и

$$\forall \varepsilon > 0\ \exists\ r_\Delta : \left| f_\Delta\left(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t\right) - f_\Delta\left(x, u(x,t), t\right) \right| \le 2^{-2}\varepsilon\ \forall\, r \ge r_\Delta . \quad (4.67)$$

С другой стороны, из (4.23), (4.30) и (4.29) следует

$$\exists \Delta_\varepsilon,\ \Delta_\varepsilon = \text{const},\ \forall \Delta,\ 0 < \Delta \le \Delta_\varepsilon,\ \Delta = \text{const} : \left| f_\Delta(x,u,t) - f(x,t) \right| \le 2^{-2}\varepsilon . \quad (4.68)$$

Пусть рассматривается аналог последовательности (4.61), получаемый заменой в (4.61) $f_\Delta\left(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t\right)$ на $f(x,t) + f_\Delta\left(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t\right) - f(x,t)$, при этом разность $f_\Delta\left(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t\right) - f(x,t)$ присоединяется к остаточному члену $-c_{u2k2n}$ и в сумме образует новый остаточный член $-\widetilde{c}_{u2k2n}$. Из (4.67), (4.68) $\left| f_\Delta\left(x, {}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t), t\right) - f(x,t) \right| \le 2^{-1}\varepsilon$ при условии $\Delta \le \Delta_\varepsilon$ и $r \ge r_\Delta$. В предположении $2^{-1}\varepsilon \le \max\limits_G \left| c_{2k2n} \right|$, получится $\max\limits_G \left| \widetilde{c}_{2k2n} \right| \le 2 \max\limits_G \left| c_{2k2n} \right|$. В результате,

$$
\begin{aligned}
&{}^{t}\overline{P}_{u_\Delta 2k2n\,r}(x,t) = {}^{t}\overline{P}_{u_\Delta 2k2n}(x,t_0) + \\
&+\int_{t_0}^{t}\left(f(x,t) - a(x,t)\,\frac{\partial\,{}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\partial x} - \widetilde{c}_{u\,2k2n} \right) dt,\ r = 1, 2, \ldots,
\end{aligned}
\tag{4.69}
$$

где

$$
\max_{G}\left|\widetilde{c}_{2k2n}\right| \le 2\max_{G}\left|c_{2k2n}\right|,\ \Delta = \text{const},\ 0 < \Delta \le \Delta_\varepsilon,\ r \ge r_\Delta .
\tag{4.70}
$$

Из (4.33) и (4.69), при условии равенства начальных условий в G_{ij},

$$
\begin{aligned}
&\overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) = \\
&= \int_{t_0}^{t}\left(f_\Delta(x,\overline{u}_\Delta,t) - f(x,t) - a(x,t)\left(\frac{\overline{u}_\Delta(x+\Delta,t) - \overline{u}_\Delta(x,t)}{\Delta} - \frac{\partial\,{}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\partial x} \right) - \widetilde{c}_{2k2n} \right) dt,
\end{aligned}
$$

или,

$$
\begin{aligned}
&\overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) = \\
&= \int_{t_0}^{t}\left(-a(x,t)\left(\frac{\overline{u}_\Delta(x+\Delta,t) - \overline{u}_\Delta(x,t)}{\Delta} - \frac{\partial\,{}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\partial x} \right) - \widetilde{c}^{(0)}_{2k2n} \right) dt,
\end{aligned}
\tag{4.71}
$$

где $\widetilde{c}^{(0)}_{2k2n} = f_\Delta(x,\overline{u}_\Delta,t) - f(x,t) - \widetilde{c}_{2k2n}$ – новый остаточный член. При этом согласно (4.70) и (4.68), с учетом предположения $2^{-1}\varepsilon \le \max_{G}\left|c_{2k2n}\right|$,

$$
\begin{aligned}
&\forall\,\varepsilon > 0,\quad \varepsilon \le 2\max_{G}\left|c_{2k2n}\right|,\ \exists\,\widetilde{\Delta}_\varepsilon = \text{const}:\ \forall\Delta,\ 0 < \Delta \le \widetilde{\Delta}_\varepsilon,\ \Delta = \text{const},\ \exists\,\widetilde{r}_\Delta : \\
&\max_{G}\left|\widetilde{c}^{(0)}_{2k2n}\right| \le 2.5\max_{G}\left|c_{2k2n}\right| \quad \forall r \ge \widetilde{r}_\Delta .
\end{aligned}
\tag{4.72}
$$

Тождественное преобразование (4.71) с учетом (4.56) влечет

$$
\begin{aligned}
\overline{u}_\Delta(x,t) - {}^{t}\overline{P}_{u_\Delta 2k2nr}(x,t) = \int_{t_0}^{t}\Bigg(-a(x,t)\Bigg(&\frac{\overline{u}_\Delta(x+\Delta,t) - \overline{u}_\Delta(x,t)}{\Delta} - D_{r-1} - \\
&- \frac{{}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x+\Delta,t) - {}^{t}\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\Delta} \Bigg) - \widetilde{c}^{(0)}_{2k2n} \Bigg) dt,
\end{aligned}
$$

или,

$$\overline{u}_{\Delta}(x,t) - {}^{t}\overline{P}_{u_{\Delta}2k2nr}(x,t) =$$

$$= \int_{t_0}^{t} \left(- a(x,t)\Delta^{-1}\left(\overline{u}_{\Delta}(x+\Delta,t) - \overline{u}_{\Delta}(x,t) - \left({}^{t}\overline{P}_{u_{\Delta}2k2n(r-1)}(x+\Delta,t) - {}^{t}\overline{P}_{u_{\Delta}2k2n(r-1)}(x,t) \right) \right) + \overline{c}_{2k2n} \right) dt,$$

где $\overline{c}_{2k2n} = a(x,t)D_{r-1} - \widetilde{c}^{(0)}_{2k2n}$. Ввиду $\left| a(x,t)D_{r-1} \right| \le M\varepsilon \ \ \forall r \ge R_{\max}$, в предположении $M\varepsilon \le \left| \widetilde{c}^{(0)}_{2k2n} \right|$, будет выполняться $\max\limits_{G} \left| \overline{c}_{2k2n} \right| \le 3.5 \max\limits_{G} \left| c_{2k2n} \right|$. Аналогично предыдущему, предположение относительно ε повлечет изменение ограничений Δ и r. Именно, с учетом (4.72),

$$\forall \varepsilon > 0,\ \varepsilon \le M^{-1} \left| \widetilde{c}^{(0)}_{2k2n} \right|,\ \exists \overline{\Delta}_{\varepsilon} = \text{const},\ 0 < \overline{\Delta}_{\varepsilon} \le \min\left(\Delta_{\varepsilon}, \widetilde{\Delta}_{\varepsilon}, \Delta_{\min}, C_0^{-1} M^{-1} \max_{G} |c_{2k2n}| \right):$$

$$\forall \Delta,\ 0 < \Delta \le \overline{\Delta}_{\varepsilon},\ \Delta = \text{const},\ \exists \overline{r}_{\Delta},\ \overline{r}_{\Delta} \ge \max(r_{\Delta}, \widetilde{r}_{\Delta}, R_{\max}),$$

такие, что $\forall r \ge \overline{r}_{\Delta}$ выполняется

$$\forall\ (x,t) \in G_{ij}, (x+\Delta,t) \in G_{\Delta ij}:$$

$$\left| \overline{u}_{\Delta}(x,t) - {}^{t}\overline{P}_{u_{\Delta}2k2nr}(x,t) \right| \le 2M\Delta^{-1} \int_{t_0}^{t} \left(\max_{G_{ij}} \left| \overline{u}_{\Delta}(x,t) - {}^{t}\overline{P}_{u_{\Delta}2k2n(r-1)}(x,t) \right| + \overline{c}_{2k2n} \right) dt,$$

и, в тех же условиях,

$$\max_{G_{ij}} \left| \overline{u}_{\Delta}(x,t) - {}^{t}\overline{P}_{u_{\Delta}2k2nr}(x,t) \right| \le 2M\Delta^{-1} \int_{t_0}^{t} \left(\max_{G_{ij}} \left| \overline{u}_{\Delta}(x,t) - {}^{t}\overline{P}_{u_{\Delta}2k2n(r-1)}(x,t) \right| + \overline{c}_{2k2n} \right) dt.$$

Остается повторить проделанные ранее рассуждения, чтобы с точностью до обозначений, значений констант и постоянных множителей вывести аналог соотношения (4.66), в котором $u_{\Delta}(x,t) \equiv u(x,t)\ \ \forall\ (x,t) \in G$ из (4.23), ${}^{t}\overline{P}_{u_{\Delta}2k2nr}(x,t)$ из (4.69) с точностью до обозначения совпадает с ${}^{t}\overline{P}_{u2k2nr}(x,t)$ из (4.28). Отсюда рассуждения, проделанные с данными видоизменениями для ${}^{t}\overline{P}_{u_{\Delta}2k2nr}(x,t)$ из (4.69) и $u_{\Delta}(x,t)$ из (4.30), сохраняются для ${}^{t}\overline{P}_{u2k2nr}(x,t)$ из (4.28) и $u(x,t)$ из (4.23). Сохраняются также описанные выше ограничения значений констант. Однако они дополнительно скорректируются при введении аналога соотношений (4.36), (4.37) путем выбора Q для нового остаточного

члена $\overline{c}_{2k2n}$. Чтобы не усложнять обозначений, ниже данные ограничения неявно подразумеваются, но не детализируются. С этой оговоркой, в условиях леммы 4.2, выполняется соотношение

$$\forall \varepsilon > 0 \;\; \exists r_{\Delta\varepsilon} : \max_{\forall (x,t)\in G_{ij}, \forall i,j:\, G_{ij}\in G} \left| u(x,t) - {}^{t}\overline{P}_{u2k2nr}(x,t) \mid_{(x,t)\in G_{ij}} \right| \le \varepsilon \;\; \forall r \ge r_{\Delta\varepsilon}, \qquad (4.73)$$

где $u(x,t)$ – решение уравнения (4.23), ${}^{t}\overline{P}_{u2k2nr}(x,t)$ из (4.28); по построению $r_{\Delta\varepsilon}$ зависит от ε, Δ и $\Delta_{\varepsilon 0} = \text{const}$: $\forall \Delta,\, 0 < \Delta \le \Delta_{\varepsilon 0}$, $\Delta = \text{const}, \Delta$ из (4.29); $\Delta_{\varepsilon 0}$ существует в качестве параметра условий сходимости (4.69), на основе которых доказывается сходимость (4.28).

Имеет место

Теорема 4.2. Пусть в области G из (4.1) – (4.3) зафиксировано 2^{2k} подобластей. Пусть в каждой подобласти G_{ij} уравнение (4.24) рассматривается с теми же начальными условиями на границе с $G_{i(j-1)}$, с которыми выполняются итерации (4.28), при этом они сохраняются с изменением номера итерации: $\overline{u}(x,t_0) = \varphi_{ij}(x,t_0)$, ${}^{t}\overline{P}_{u2k2nr}(x,t_0) = \varphi_{ij}(x,t_0)$, $r = 0, 1, \ldots$, $t_0 = t_0(i,j)$. Тогда кусочно-интерполяционное приближение с итерационным уточнением (4.28) решения задачи (4.23) равномерно сходится в G к решению $u(x,t)$ с оценкой (4.73). Относительно равномерной и кусочной непрерывности последовательных приближений (4.28) без принципиальных изменений формулируется аналог утверждения леммы 4.4, данного относительно приближений (4.61).

4.5. Приближение частных производных при моделировании переноса. Пусть выполнены условия леммы 4.2 и теоремы 4.2. Тогда

$$\left| \frac{\partial u(x,t)}{\partial x} - \frac{\partial\, {}^{t}\overline{P}_{u2k2nr}(x,t)}{\partial x} \right| \le a_1 + a_2 + a_3,$$

где

$$a_1 = \left| \frac{\partial u(x,t)}{\partial x} - \frac{u(x+\Delta,t) - u(x,t)}{\Delta} \right|,$$

$$a_2 = \left| \frac{u(x+\Delta,t) - u(x,t)}{\Delta} - \frac{{}^t\overline{P}_{u2k2nr}(x+\Delta,t) - {}^t\overline{P}_{u2k2nr}(x,t)}{\Delta} \right|,$$

$$a_3 = \left| \frac{{}^t\overline{P}_{u2k2nr}(x+\Delta,t) - {}^t\overline{P}_{u2k2nr}(x,t)}{\Delta} - \frac{\partial\, {}^t\overline{P}_{u2k2nr}(x,t)}{\partial x} \right|.$$

Слагаемое a_1 оценивается из (4.29), для оценки a_3 можно повторить рассуждения, проделанные для (4.56) – (4.60). При этом $\Delta = \text{const}$ можно считать столь малым, что $a_1 \le 3^{-1}\varepsilon_0$ и $a_3 \le 3^{-1}\varepsilon_0$. Кроме того, $a_2 \le 2\Delta^{-1} \max\limits_{\forall (x,t)\in G_{ij}} \left| u(x,t) - {}^t\overline{P}_{u2k2nr}(x,t) \right|$. В (4.73) для такого Δ ничто не исключает $\varepsilon \le 6^{-1}\Delta\, \varepsilon_0$, при необходимости можно указать соответственные Δ_ε и $r_{\Delta\varepsilon}$. Тогда $a_2 \le 3^{-1}\varepsilon_0$. В неравенствах можно перейти к максимуму по $\forall\, i,j: G_{ij} \in G$. В результате

$$\forall\, \varepsilon_0 > 0 \;\; \exists\, \Delta_{\varepsilon_0} = \text{const}\,, \exists r_{\Delta\varepsilon_0}:$$
$$\max_{\forall\,(x,t)\in G_{ij}, \forall i,j:\, G_{ij}\in G} \left| \frac{\partial u(x,t)}{\partial x} - \frac{\partial\, {}^t\overline{P}_{u2k2nr}(x,t)}{\partial x} \Big|_{(x,t)\in G_{ij}} \right| \le \varepsilon_0 \quad \forall r \ge r_{\Delta\varepsilon_0}. \tag{4.74}$$

Производная $\dfrac{\partial u}{\partial t}$ приближается с оценкой, зависящей от размера подобласти. Из (4.23) и (4.69)

$$\left| \frac{\partial u}{\partial t} - \frac{\partial\, {}^t\overline{P}_{u_\Delta 2k2nr}(x,t)}{\partial t} \right| \le M \left| \frac{\partial u}{\partial x} - \frac{\partial\, {}^t\overline{P}_{u_\Delta 2k2n(r-1)}(x,t)}{\partial x} \right| + c\, 2^{-k(2n+1)} h^{2n+1},\ r = 1, 2, \ldots,$$

где учитывается (4.70) и $c = 2C$. С подстановкой (4.74), $\left| \dfrac{\partial u}{\partial t} - \dfrac{\partial\, {}^t\overline{P}_{u_\Delta 2k2nr}(x,t)}{\partial t} \right| \le M\, \varepsilon_0 + c\, 2^{-k(2n+1)} h^{2n+1} \quad \forall r \ge r_{\Delta\varepsilon_0}$. Здесь и в (4.74) вместо ε_0 можно взять $\min(\varepsilon_0, M^{-1}\varepsilon_0)$, соответственно скорректировав Δ_{ε_0} и $r_{\Delta\varepsilon_0}$. Тогда

$\forall \varepsilon_0 > 0 \exists\, \Delta_{\varepsilon_0} = \text{const}$, $\exists r_{\Delta\varepsilon_0}$:

$$\max_{\forall (x,t)\in G_{ij}, \forall i,j:\, G_{ij}\in G} \left| \frac{\partial u}{\partial t} - \frac{\partial\, {}^{t}\overline{P}_{u_\Delta 2k2n\, r}(x,t)}{\partial t} \right| \le \varepsilon_0 + c\, 2^{-k(2n+1)} h^{2n+1},\ c = \text{const},\ \forall r \ge r_{\Delta\varepsilon_0}.$$

4.6. Обработка данных квазилинейной модели переноса. Пусть рассматривается задача

$$\frac{\partial u}{\partial t} + a(u,x,t)\,\frac{\partial u}{\partial x} = f(x,t),\ \ u(x,0) = \varphi(x), \tag{4.75}$$

где принимается, что $f(x,t)$ – заданная функция в области $\{(x,t) \mid x \in R, t \ge 0\}$, $\varphi(x)$ – заданная функция $x \in R$, $a(u,x,t)$ – функция, заданная в области $\{\, |u(x,t) - u(x,0)| \le \widetilde{B}; (x,t) \mid x \in R, t \ge 0\}$, $\widetilde{B}$ – некоторая постоянная, выбор которой оговаривается аналогично условию I. Изложенный выше метод рассматривается в предположениях I – III, с тем изменением, что $a(u,x,t)$ задана в области $\widetilde{R}_u = \{\, |u(x,t) - u(x,0)| \le \widetilde{B}; (x,t) \in G\}$, и относительно (4.75) приняты все предположения, сделанные относительно (4.23). Из (4.75)

$$u(x,t) = u(x,t_0) + \int_{t_0}^{t} f(x,t) - a(u,x,t)\,\frac{\partial u(x,t)}{\partial x}\, dt,\ \ u(x,0) = \varphi(x).$$

Аналогично предыдущему, определяются последовательности

$$\overline{\Psi}^{\,ij}_{\partial u\, 2k2n\,(r-1)}(z,w) = f(x,t) - a({}^{t}\overline{P}_{u2k2n\,(r-1)}(x,t),x,t)\,\frac{\partial\, {}^{t}\overline{P}_{u2k2n\,(r-1)}(x,t)}{\partial x} + c_{2k2n},$$

$${}^{t}\overline{P}_{u2k2n\,r}(x,t) = {}^{t}\overline{P}_{u2k2n\,r}(x,t_0) + \int_{t_0}^{t} \overline{\Psi}^{\,ij}_{\partial u\, 2k2n\,(r-1)}(z,w)\ dt + {}^{t}c_{u2k2n},$$

где z, w из (4.20). Отсюда,

$${}^{t}\overline{P}_{u2k2n\,r}(x,t) = {}^{t}\overline{P}_{u2k2n\,r}(x,t_0) +$$
$$+ \int_{t_0}^{t} \left(f(x,t) - a({}^{t}\overline{P}_{u2k2n\,(r-1)}(x,t),x,t)\,\frac{\partial\, {}^{t}\overline{P}_{u2k2n\,(r-1)}(x,t)}{\partial x} \right) dt + {}^{t}c_{u2k2n}.$$

Рассматривается вспомогательная задача

$$\frac{\partial u_\Delta}{\partial t} + a(u,x,t)\,\frac{u_\Delta(x+\Delta,t)-u_\Delta(x,t)}{\Delta} = f_\Delta(x,t),\;\; u_\Delta(x,0)=\varphi(x), \tag{4.76}$$

где $(x,t)\in G$, $(x+\Delta,t)\in G_\Delta$,

$$f_\Delta(x,t) = f(x,t) - a(u,x,t)\left(\frac{\partial u}{\partial x} - \frac{u(x+\Delta,t)-u(x,t)}{\Delta}\right),\;\; u_\Delta(x,t)\equiv u(x,t)$$ из (4.75)

$\forall\,(x,t)\in G$, $\Delta=\text{const}$ произвольно выбрано в соответствии с (4.29). В G_{ij} при $t_0=t_0(i,j)$ (4.75) преобразуется к виду

$$\begin{aligned} &u_\Delta(x,t) = u_\Delta(x,t_0) + \\ &+\int_{t_0}^{t}\left(f_\Delta(x,t) - a(u,x,t)\,\frac{u_\Delta(x+\Delta,t)-u_\Delta(x,t)}{\Delta}\right)dt,\; u_\Delta(x,t_0)=\varphi_{ij}(x), \end{aligned} \tag{4.77}$$

где $(x,t)\in G_{ij}$, $(x+\Delta,t)\in G_{\Delta ij}$. Строится последовательность

$$\begin{aligned} {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t) = {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t_0) + \int_{t_0}^{t}\Bigg(& f_\Delta(x,t) - a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\times \\ &\times\frac{{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x+\Delta,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\Delta}\Bigg)dt + {}^t c_{u\,2k2n}. \end{aligned} \tag{4.78}$$

Пусть в (4.77) и (4.78) начальные условия одинаковы: $\overline{u}_\Delta(x,t_0) = {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t_0)$, $r=0,1,\ldots$, $t_0=t_0(i,j)$, решение $\overline{u}_\Delta(x,t)$ отмечается чертой. Тогда

$$\begin{aligned} \overline{u}_\Delta(x,t) - {}^t\overline{P}_{u_\Delta 2k2n\,r}(x,t) = \int_{t_0}^{t}\Bigg(& -a(\overline{u}_\Delta,x,t)\frac{\overline{u}_\Delta(x+\Delta,t)-\overline{u}_\Delta(x,t)}{\Delta} + \\ &+ a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\frac{{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x+\Delta,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)}{\Delta}\Bigg)dt + {}^t c_{u\,2k2n}. \end{aligned} \tag{4.79}$$

Выражение под знаком интеграла преобразуется в сумму A_1+A_2, где

$$A_1 = \Delta^{-1}\left(a(\overline{u}_\Delta,x,t)\;\overline{u}_\Delta(x,t) - a({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t)\;{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right),$$

$$A_2 = \Delta^{-1}\left(-a(\overline{u}_\Delta,x,t)\;\overline{u}_\Delta(x+\Delta,t) + a({}^t\overline{P}_{u_\Delta 2k2n(r-1)}(x,t),x,t)\;{}^t\overline{P}_{u_\Delta 2k2n(r-1)}(x+\Delta,t)\right).$$

В A_1 можно добавить и вычесть $\Delta^{-1}a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\overline{u}_\Delta(x,t)$. Отсюда

$$\left|A_1\right| \le \Delta^{-1}\left(\left|\overline{u}_\Delta(x,t)\right|\left|a(\overline{u}_\Delta,x,t)-a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\right|+\right.$$
$$\left.+\left|a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\right|\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|\right).$$

Учитывая аналог замечания 4.2, можно применить условие Липшица к функции $a(\overline{u}_\Delta,x,t)$:

$$\left|A_1\right| \le \Delta^{-1}\left(\widetilde{L}\left|\overline{u}_\Delta(x,t)\right|\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|+\right.$$
$$\left.+\left|a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\right|\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|\right),\ \widetilde{L}=\text{const}.$$

В результате

$$\left|A_1\right| \le \Delta^{-1}\left(\overline{c}\,\widetilde{L}\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|+\widetilde{M}\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|\right),$$

где $\overline{c}=\max\limits_G\left|\overline{u}_\Delta(x,t)\right|$, $\overline{c}=\text{const}$, $\widetilde{M}=\max\limits_G\left|a(\overline{u},x,t)\right|$, $\widetilde{M}=\text{const}$. Отсюда

$$\left|A_1\right| \le \widetilde{c}\max_{G_{ij}}\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|,\ \widetilde{c}=2\Delta^{-1}\max\left(\overline{c}\widetilde{L},\widetilde{M}\right),\ \widetilde{c}=\text{const}.$$

В A_2 можно добавить и вычесть $\Delta^{-1}a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\overline{u}_\Delta(x+\Delta,t)$. Тогда

$$\left|A_2\right| \le \Delta^{-1}\left(\left|-a(\overline{u}_\Delta,x,t)\,\overline{u}_\Delta(x+\Delta,t)+a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\overline{u}_\Delta(x+\Delta,t)\right|+\right.$$
$$\left.+\left|a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\right|\left|\overline{u}_\Delta(x+\Delta,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x+\Delta,t)\right|\right).$$

С применением условия Липшица,

$$\left|A_2\right| \le \Delta^{-1}\left(\widetilde{L}\left|\overline{u}_\Delta(x+\Delta,t)\right|\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|+\right.$$
$$\left.+\left|a\left({}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t),x,t\right)\right|\left|\overline{u}_\Delta(x+\Delta,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x+\Delta,t)\right|\right).$$

Как и выше,

$$\left|A_2\right| \le \widetilde{c}\times\max_{G_{ij}}\left|\overline{u}_\Delta(x,t)-{}^t\overline{P}_{u_\Delta 2k2n\,(r-1)}(x,t)\right|,\ \widetilde{c}=2\Delta^{-1}\max\left(\overline{c}\,\widetilde{L},\widetilde{M}\right),\ \widetilde{c}=\text{const}.$$

Отсюда и из (4.79)

$$\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}\,2k\,2nr}(x,t)\right|\le\int_{t_0}^{t}\left(2\widetilde{c}\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n\,(r-1)}(x,t)\right|+\left|c_{2k\,2n}\right|\right)dt\,.$$

Ввиду произвольности $(x,t)\in G_{ij}$,

$$\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}\,2k\,2nr}(x,t)\right|\le\int_{t_0}^{t}\left(2\widetilde{c}\max_{G_{ij}}\left|\overline{u}_{\Delta}(x,t)-{}^{t}\overline{P}_{u_{\Delta}\,2k\,2n\,(r-1)}(x,t)\right|+\left|c_{2k\,2n}\right|\right)dt\,. \quad (4.80)$$

На основе (4.80) можно рассматривать преобразования, аналогичные выполненным для линейного уравнения, включая (4.36), (4.40), а также переход от (4.30) к (4.23), в данном случае – от (4.76) к (4.75).

Если в (4.75) вместо $f(x,t)$ рассматривать $f(u,x,t)$, то в (4.79) в качестве слагаемого под знаком интеграла появится разность $f_{\Delta}(\overline{u},x,t)-f_{\Delta}\left({}^{t}\overline{P}_{u_{\Delta}\,2k\,2n(r-1)}(x,t),x,t\right)$, модуль которой оценивается по условию Липшица. Это изменит значение константы в (4.80), но не исключает оценку сходимости метода.

4.7. Численный эксперимент. Изложенный метод реализован программно. Реализация содержит отклонения от исходного описания. Именно, не использован переход к описанному треугольнику области G. Ниже алгоритм реализации описан для области G без ее разбиения на подобласти, при $k_x=k_t=0$ в (4.2), (4.3); при $k_x,k_t\in\{1,2,\dots\}$ алгоритм выполняется аналогично для каждой подобласти G_{ij} из (4.2).

Интерполирование выполняется на основе (4.7) с переводом в форму (4.9) без удвоения степени полинома. Аналог (4.28)

$$\Psi_{\partial u\,n\,(r-1)}(z,w)\approx f(x,t)-a(x,t)\,\frac{\partial\,{}^{t}P_{u\,n\,(r-1)}(x,t)}{\partial x},$$

$${}^{t}P_{u\,n\,r}(x,t)={}^{t}P_{u\,n\,r}(x,t_0)+\int_{t_0}^{t}\Psi_{\partial u\,n\,(r-1)}(z,w)\,dt\,,$$

служит для итерационного уточнения, коэффициенты преобразуются с учетом (4.20) – (4.22). Полученное в результате r таких итераций (конечное число итераций определяется с использованием описываемой в дальнейшем интегральной невязки) полиномиальное приближение искомого решения ${}^{t}P_{u\,n\,r}(x,t)$, обозначаемое ${}^{t}P_{u\,n}(x,t)$, интерполирует решение задачи (4.23) в нижней треугольной части рассматриваемой области G:

$$u(x,t) \approx {}^{t}P_{u\,n\,r}(x,t) = {}^{t}P_{u\,n}(x,t),\ (x,t) \in G'$$

где $G' = \{(x,t) \mid x \in [a,b],\ 0 \le t \le T(b-x)/(b-a)\}$.

Экстраполяция приближения решения для оставшейся части области $G \setminus G'$ согласно экспериментам [126, 127, 151] влечет избыточную погрешность и как следствие – снижение точности приближения в целом в области G.

С целью сохранения сравнительно высокой точности приближения в целом в области G, интерполяция с итерационным уточнением выполняется также в областях, получающийся сдвигом G вдоль OX влево (G^{left}) и вправо (G^{right}). Построенные приближения обозначаются ${}^{t}P_{un}^{left}(x,t)$ и ${}^{t}P_{un}^{right}(x,t)$ соответственно. Сдвиг выполняется на некоторую фиксированную длину, которая кратна расстоянию между узлами интерполяции (в программной реализации использованы значения сдвига $k_{left} = k_{right} = h_x \lfloor n/2 \rfloor$, где n – степень интерполяционного полинома). Для областей G^{left} и G^{right}, также как и для области G, решение задачи (4.32) существует и единственно, поскольку все эти области содержатся в полуплоскости $\{(x,t) \mid x \in R, t \ge 0\}$, в которой в общем случае рассматривается решение исследуемой задачи.

Таким образом, получаются три приближения решения: исходное в области G и еще два, полученные в областях G^{left} и G^{right}. Данные области пересекаются, характер пересечения наглядно выражен на рис. 4.1.

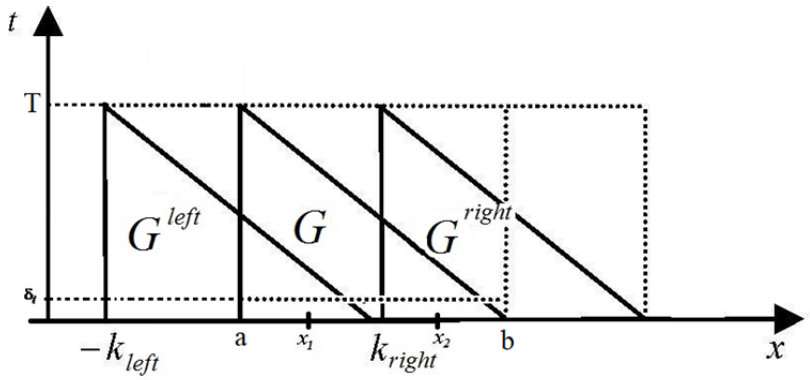

Рис. 4.1. Пересечение областей G, G^{left} и G^{right}

Согласно численному эксперименту наименее точные приближения оказывались вдоль сторон треугольника, образованного узлами интерполяции (рис. 4.1). Обозначим левую и правую границу центральной части G по координате x соответственно x_1 и x_2 (рис. 4.1) (в программной реализации использованы следующие значения границ: $x_1 = a + h_x[n/4]$, $x_2 = b - h_x[n/4]$). Тогда при $x < x_1$ за приближение принимается полином ${}^tP_{un}^{left}(x,t)$, построенный для G^{left}. При $x_1 \le x < x_2$ за приближение принимается полином ${}^tP_{un}(x,t)$, построенный для G. При $x \ge x_2$ за приближение принимается полином ${}^tP_{un}^{right}(x,t)$, построенный для G^{right}.

Полный процесс приближения со сдвигами вдоль оси OX циклически повторяется со сдвигом по вертикали вдоль оси OT. Окончательные полиномиальные приближения выбираются из десятой части (значение δ_t на рис. 4.1), отсчитанной от нижнего основания вертикально сдвигаемой прямоугольной области («нижний прямоугольный слой»), что устойчиво повышает точность приближения. В процессе смещения вдоль OT полученные на верхней границе текущего нижнего прямоугольного слоя приближения решения принимаются за начальные условия для следующего за ним нижнего прямоугольного слоя, причем с учетом $x < x_1$, $x_1 \le x < x_2$, $x \ge x_2$. Контроль точности привязан к текущему нижнему прямоугольному слою. Степень

полинома выбирается на основе минимизации аналога невязки (ниже просто невязки) специального вида. Невязка

$$S_n = \left| s_n - s_{n-1} \right|$$

определяется из соотношений

$$s_n = \int_a^{x_1} dx \int_{t_k}^{t_{k+1}} {}^t P_{un}^{left}(x,t)\,dt + \int_{x_1}^{x_2} dx \int_{t_k}^{t_{k+1}} {}^t P_{un}(x,t)\,dt + \int_{x_2}^{b} dx \int_{t_k}^{t_{k+1}} {}^t P_{un}^{right}(x,t)\,dt ,$$

$$s_{n-1} = \int_a^{x_1} dx \int_{t_k}^{t_{k+1}} {}^t P_{u(n-1)}^{left}(x,t)\,dt + \int_{x_1}^{x_2} dx \int_{t_k}^{t_{k+1}} {}^t P_{u(n-1)}(x,t)\,dt + \int_{x_2}^{b} dx \int_{t_k}^{t_{k+1}} {}^t P_{u(n-1)}^{right}(x,t)\,dt ,$$

и вычисляется в текущем (k-м) нижнем прямоугольном слое. Здесь x_i – соответственные границы сдвигов вдоль OX, t_k, t_{k+1} – временные границы k-го нижнего прямоугольного слоя. Геометрически «невязка» представляет модуль разности объемов фигур с боковыми поверхностями и нижними основаниями параллелепипедов, ограниченных сверху криволинейными поверхностями, отображающими кусочно-интерполяционные приближения решения $u(x,t)$ посредством полиномов степени n и $n-1$ соответственно. Сходная форма невязки используется при выборе числа итераций r, для каждой разновидности полиномов ${}^t P_{un}^{left}(x,t)$, ${}^t P_{un}^{right}(x,t)$ и ${}^t P_{un}(x,t)$ это число выбирается отдельно. Для ${}^t P_{un}(x,t)$ вычисляется

$$I_r = \left| i_r - i_{r-1} \right|,\ r = \overline{r_{\min}, r_{\max}},$$

где

$$i_r = \int_a^b dx \int_{t_k}^{t_{k+1}} {}^t P_{un\,r}(x,t)\,dt ,\ i_{r-1} = \int_a^b dx \int_{t_k}^{t_{k+1}} {}^t P_{un\,(r-1)}(x,t)\,dt .$$

Соответствующие значения невязки для ${}^t P_{un}^{left}(x,t)$ и ${}^t P_{un}^{right}(x,t)$ вычисляются аналогично. Значения n и r, при которых S_n и I_r принимают наименьшие значения фиксируются.

При компьютерной реализации для вычисления значений невязки S_n и I_r применяется модификация, заключающаяся в разбиении области интегрирования на подобласти и суммировании значений интегралов в каждой из подобластей. Более подробно, при каждом вычислении значения повторного интеграла текущая область интегрирования обозначается $\beta = [a,b]\times[c,d] = \{(x,t) \mid x \in [a,b],\ t \in [c,d]\}$ и разбивается на подобласти $\beta_{\ell m}$, объединение которых покрывает β:

$$\beta = \bigcup_{m=0}^{K_t-1} \bigcup_{\ell=0}^{K_x-1} \beta_{\ell m},$$

где $\beta_{\ell m} = \{ (x,t) \mid x \in [x_\ell, x_{\ell+1}],\ t \in [t_m, t_{m+1}] \}$ (в компьютерной реализации число подобластей – $K_x = 8$, $K_t = 4$). При этом в качестве итогового значения повторного интеграла по аддитивности принимается сумма значений повторного интеграла во всех подобластях $\beta_{\ell m}$, $\ell = 0, K_x - 1$, $m = 0, K_t - 1$.

Программная реализация метода представлена в листинге 4.4. На вход программы поступают функции $a(x,t)$, $f(x,t)$, $\varphi(x)$ из (4.23) и границы области решения. Помимо этого, на вход программы поступают значения констант *Nmin* и *Nmax*, определяющие границы варьируемой степени интерполяционного полинома с целью программного выбора наилучшей степени полинома на каждом «нижнем прямоугольном слое» с использованием невязки S_n, и значения констант *K_iter_min*, *K_iter_max*, определяющие границы изменения количества итераций с целью программного выбора наилучшего количества итераций для каждого интерполяционного полинома с использованием невязки I_r. Условия воспроизведения итераций и вычисления их количества в пределах границы выделены в программе *VPI_IVP_FOR_PDE*, приведенной ниже в листинге 4.4, соответственными комментариями.

Непосредственно ниже приводится код программы кусочно-интерполяционного приближения решения задачи Коши с итерационным

уточнением для линейного уравнения переноса. Конкретные значения входных параметров, приведенные в программе *VPI_IVP_FOR_PDE*, представленной в листинге 4.4, соответствуют задаче

$$u'_t + u'_x = 0,\ \ u(x,0) = \sin(x). \tag{4.81}$$

Результат работы программы анализируется в приводимом ниже описании численного эксперимента.

Листинг 4.4

```
program VPI_IVP_FOR_PDE; // Ut+a(x,t)*Ux = f(x,t), u(x,0)=IC(x)
{$APPTYPE CONSOLE}
uses SysUtils, DateUtils; const nn=15;
a_Gl=0; b_Gl=1; c_Gl=0; d_Gl=1; //Размеры области
kk=0;//число, определяющее количество подобластей разбиения 2^2k
Nmin=9; Nmax=14; // границы вариации степени интерполяционного полинома
K_iter_min=3; K_iter_max=50;  // границы вариации кол-ва итераций
outputGl_x=10; outputGl_t=10;//кол-во точек вывода
type
 Mass1= array[-nn..nn] of extended;
 Mass2= array[-nn..nn,-nn..nn] of extended;
 Mass2_int = array[0..nn,0..nn] of integer; Mass4_=^Mass4;
 Mass4= array[0..nn,0..nn,0..nn,0..nn] of extended; Mass5_=^Mass5;
 Mass5= array[0..nn,0..nn,0..nn,0..nn,0..nn] of extended;
 Mass7= array[0..500,1..3] of Mass2;
 Mass8_=^Mass8; Mass8=array[-16..16] of Mass7;
var
      minN_G,oprpol,sdvigr,sdvigl,kl,kr,Nsloya, N_G: shortint;
                                    podT,podtGl,Ksl: longint;
   hx,ht,hhx,hht,c00,d00,x1,x2,a0,c0,x1_t,x2_t,line: extended;
                   x1_,x2_,x1_prevN,x2_prevN,Minsum: extended;
 iterMinC,iterMinL,iterMinR,iterMin_C,iterMin_L,iterMin_R,fake:integer;
                                hour,minut,sec,msec: word;
                                      tnach,tkonech: extended;
                                   x,xx,t,factorial: Mass1;
                                    U,C,CX,Fi,C1,CT: Mass2;
                         C_last_C,C_last_R,C_last_L: Mass2;
              C_after,C_last_L_,C_last_C_,C_last_R_: Mass2;
       C_last_L_prevN,C_last_C_prevN,C_last_R_prevN: Mass2;
                                              MasPolT: Mass7;
                                                MasGl: Mass8_;
                                          U0,U0L,U0R: Mass1;
                                                DD_G: Mass4_;
function f(x,t: extended): extended; begin f: = 0 end;
function a(x,t: extended): extended; begin a: = 1 end;
function IC(x:extended):extended; //начальное условие
 begin IC:=sin(x) end;
//точное аналитическое решение для вывода погрешности
function Solution(x,t:extended):extended; begin Solution:=sin(x-t)end;
//Вычисление коэффициентов полиномов
procedure Viet2(var DD:Mass4_); var k,l,i,j,m_: shortint;d,e: Mass2_int;
begin e[1,1]:=1; e[1,0]:=0;
 for k:=2 to nn do begin e[k,0]:=-e[k-1,0]*(k-1);
  for l:=1 to k-1 do e[k,k-l]:=e[k-1,k-l-1]-e[k-1,k-l]*(k-1);
  e[k,k]:=e[k-1,k-1] end;
 for k:=1 to nn do for l:=0 to k do d[l,k]:= e[k,l];
```

```
//Вычисление коэффициентов для двумерного полинома
for m_:=1 to nn do for k:=0 to m_ do
for i:=0 to k do for j:=0 to m_-k do begin
  if (k=0) then Dd[m_,k,i,j]:=d[j,m_-k]
   else if (m_-k=0) then Dd[m_,k,i,j]:= d[i,k]
    else Dd[m_,k,i,j]:=d[i,k]*d[j,m_-k]
  end end;
//Вычисление конечных разностей
procedure Dual_finite_differences(UU:Mass2; N1:integer; var dz:mass5_);
var i,j,k,m:shortint;
begin
 for i:=0 to N1-1 dofor j:=0 to N1-1-i do begin
   dz[1,1,0,i,j]:=UU[i+1,j]-UU[i,j]; dz[1,0,1,i,j]:=UU[i,j+1]-UU[i,j]; end;
 for m:=2 to N1 do for k:=0 to m do
  for i:=0 to N1-m do for j:=0 to N1-i-m do
    if m-k<>0 then dz[m,k,m-k,i,j]:=dz[m-1,k,m-k-1,i,j+1]-dz[m-1,k,m-k-1,i,j]
    else dz[m,k,m-k,i,j]:=dz[m-1,k-1,m-k,i+1,j]-dz[m-1,k-1,m-k,i,j]
end;
procedure preparation; var i:shortint;
begin New(DD_G);New(MasGl);Viet2(DD_G);
factorial[0]:=1; for i:=1 to nn do factorial[i]:=i*factorial[i-1]; end;
procedure Sdvig(N:integer);
 begin kl:=trunc(N/2); kr:=trunc(N/2);
  sdvigl:=trunc(N/4); sdvigr:=trunc(N/4); end;
//Вычисление коэффициентов полинома от двух переменных
procedure Two_dim_Newton(UU:mass2; N:integer; var C_:mass2);
var i,j,m_,k_:integer; Summa:extended; b_:mass2; dz_:Mass5_;
begin
 New(dz_); Dual_finite_differences(UU, N, dz_);
 for i:=0 to N do
 for j:=0 to i do b_[i,j]:=dz_[i,j,i-j,0,0]/factorial[j]/factorial[i-j];
 for i:=0 to N do
 for j:=0 to N-i do begin
   Summa:=0;
   for m_:=j to N    do
   for k_:=i to m_-j do Summa:= Summa + b_[m_,k_]*Dd_G[m_,k_,i,j];
   if (i=0) and (j=0) then C_[i,j]:=Summa+UU[0,0]  else C_[i,j]:=Summa;
  end;
 C_[-1,0]:=x[0]; C_[0,-1]:=t[0]; C_[-2,0]:=hhx; C_[0,-2]:=hht; C_[-1,-1]:=N;
 Dispose(dz_);
end;
//Подсчет значений полинома по коэффициентам и степени
function Polinom(xx,tt:extended; CC:mass2):extended;
var i,j,N:shortint;
    zp,wp,polin,polinom_z:extended;
begin
 zp:=(xx-CC[-1,0])/CC[-2,0]; wp:=(tt-CC[0,-1])/CC[0,-2];
 N:=trunc(CC[-1,-1]); polin:=CC[0,N];
 for i:=N-1 downto 0 do
  begin
   polin:=polin*wp; polinom_z:=CC[N-i,i];
   for j:=N-i-1 downto 0 do polinom_z:=polinom_z*zp+CC[j,i];
   polin:=polin+polinom_z;
  end;
 Result:=polin;
end;
procedure Utochnenie; var i,j:word;
 begin Two_dim_Newton(U, N_G, C);
//Вычисление массива коэффициентов для аппроксимации UX
for i:=0 to N_G-1 do for j:=0 to N_G-1 do CX[i,j]:=C[i+1,j]*(i+1)/hhx;
  CX[-1,0]:=x[0]; CX[0,-1]:=t[0]; CX[-2,0]:=hhx;
CX[0,-2]:=hht; CX[-1,-1]:=N_G-1;
```

```
  //Вычисление условий интерполяции
for i:=0 to N_G - 1 do for j:=0 to N_G - 1 do
   Fi[i,j]:=f(x[i],t[j])-a(x[i],t[j])*Polinom(x[i],t[j],CX);
  Two_dim_Newton(Fi, N_G-1, CT);
  for i:=0 to N_G-1 do for j := 0 to N_G - 1 do C1[i,j]:= CT[i,j]/(j+1);
 C1[-1,0]:=x[0]; C1[0,-1]:=t[0]; C1[-2,0]:=hhx; C1[0,-2]:=hht; C1[-1,-1]:=N_G-1;
end;
procedure Nachusl; var i,j:shortint;
begin case OprPol of
 0:begin
   if podT=0 then for j:=0 to N_G   do for i:=0 to N_G-j do u[i,j]:=IC(x[i])
   else begin
    for i:=0 to N_G do if x[i]<x1 then U0[i]:=Polinom(x[i],t[0],C_last_L)
     else if x[i]<x2 then U0[i]:=Polinom(x[i],t[0],C_last_C)
     else U0[i]:= Polinom(x[i],t[0],C_last_R);for i:=0 to N_G do u[i,0]:= U0[i];
  for j:= 1 to N_G do for i:=0 to N_G-j do U[i,j]:=Polinom(x[i],t[j], C_last_C)
   end; end;
 -1:begin
     if podT =0 then for j:=0 to N_G do for i:= 0 to N_G-j do u[i,j]:= IC(x[i])
     else begin
       for i:=0 to N_G do if x[i]<x1 then
         begin u[i,0]:=Polinom(x[i],t[0],C_last_l);U0L[i]:=u[i,0] end
        else begin u[i,0]:=Polinom(x[i],t[0],C_last_C);U0L[i]:=u[i,0] end;
       for j:=1 to N_G   do for i:=0 to N_G-j do u[i,j]:=u[i,0];
    end; end;
  1:begin
     if podT= 0 then for j:=0 to N_G do for i:= 0 to N_G-j do u[i,j]:= IC(x[i])
     else begin
   for i:=0 to N_G do for j:=0 to N_G-i do u[i,j]:=Polinom(x[i],t[j],C_last_R);
   for i:=0 to N_G do if x[i]<x2 then
       begin u[i,0]:= Polinom(x[i],t[0],C_last_C);U0R[i]:=u[i,0] end
        else begin u[i,0]:=Polinom(x[i],t[0],C_last_R); U0R[i]:=u[i,0] end;
 end; end;end; end;
//Вычисление значения для «невязки»
function integral(a,b,c,d:extended; CPol:mass2):extended;
var N,i,j:shortint; x0,t0:extended; CPol2:mass2;
     hx_int,ht_int,x_int,t_int,int,sum_int:extended; a_,b_,c_,d_:extended;
begin hx_int:=(b-a)/8; ht_int:=(d-c)/4; sum_int:=0; a_:=a; b_:=a_+hx_int;
  repeat c_:=c; d_:=c_+ht_int;
  repeat N:=trunc(CPol[-1,-1]); x0:=CPol[-1,0]; t0:=CPol[0,-1];CPol2:=Cpol;
   for i:=0 to N do for j:=0 to N do CPol2[i,j]:=CPol[i,j]/((i+1)*(j+1));
   Int:=(d-t0)*(b-x0)*Polinom(b,d,CPol2)-(c-t0)*(b-x0)*Polinom(b,c,CPol2)-(d-
t0)*(a-x0)*Polinom(a,d,CPol2)+(c-t0)*(a-x0)*Polinom(a,c,CPol2);
   Sum_int:=sum_int+Int; c_:=c_+ht_int; d_:=c_+ht_int;
  until abs(d_-d)<=1e-10;
   a_:=a_+hx_int; b_:=a_+hx_int;
 until abs(b_-b)<=1e-10;
 integral:=Sum_int;
end;
procedure IterUt(flag:boolean;Kit:integer;U00:Mass1;var C_last:Mass2;
var iterMin:integer); //Итерационное полиномиальное уточнение
var Int_after_it,Int_it,nev_it,MinSum_it:extended; i,j:shortint; iter:word;
begin MinSum_it:=100;
for iter:= 1 to Kit do begin
   Utochnenie;
   for i:= 0 to N_G do for j:= 0 to N_G do
    if podT=0 then U[i,j]:= Polinom(x[i],t[j],C1)*(t[j]-t[0])+IC(x[i])
    else  U[i,j]:= Polinom(x[i],t[j],C1)*(t[j]-t[0])+U00[i];
   if flag then begin
     two_dim_Newton(U, N_G, C_after);
     Int_after_it:=integral(x[0],x[N_G],t[0],t[0]+line, C_after);
     Int_it:=integral(x[0],x[N_G],t[0],t[0]+line,C);
```

```
     nev_it:=abs(Int_it-Int_after_it);
     if nev_it<MinSum_it then
      begin MinSum_it:=nev_it; iterMin:=iter; C_last:=C end;
  end end;
 if flag=false then C_last:=C; C_last[-1,0]:=x[0]; C_last[0,-1]:=t[0];
 C_last[-2,0]:=hhx;  C_last[0,-2]:=hht;  C_last[-1,-1]:=N_G;
end;
procedure NachuslNewLevo; var i,j:shortint;
begin dec(Nsloya);
 if podT=0 then begin
  for j:=0 to N_G do for i:=0 to N_G-j do u[i,j]:=IC(x[i]); end
 else begin
 for i:=kl to N_G do for j:=1 to N_G-i do u[i,j]:=Polinom(x[i],t[j],C_last_C_);
  for i:=0 to kl-1 do for j:=0 to N_G-i do begin
  podtGl:=trunc(t[j]/line);
     if (t[j]<>0) and (frac(t[j]/line)=0) then podtGl:=podtGl-1;
     x1_t:=MasGl[Nsloya,podTGl,1][-3,0]; x2_t:=MasGl[Nsloya,podTGl,1][-4,0];
     if x[i]<x1_t then u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,1])
     else
      if x[i]<x2_t then u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,2])
      else u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,3])
    end;
   for i:=0 to N_G do if x[i]<x1 then begin
      podtGl:=trunc(t[0]/line);
      if (t[0]<>0) and (frac(t[0]/line)=0) then dec(podtGl);
      x1_t:=MasGl[Nsloya,podTGl,1][-3,0]; x2_t:=MasGl[Nsloya,podTGl,1][-4,0];
      if x[i]<x1_t then u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,1])
      else
       if x[i]<x2_t then u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,2])
       else u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,3]);
      U0L[i]:=u[i,0] end
    else begin u[i,0]:=Polinom(x[i],t[0],C_last_C); U0L[i]:=u[i,0] end;
  end; inc(Nsloya); end;
procedure solve(flag:boolean;Nsl:integer);var i,j:shortint;
begin Sdvig(N_G); hhx:=hx/N_G; hht:=(d00-c00)/N_G;
 for i:=-N_G to N_G do x[i]:=a0 + i*hhx;
 for j:=1 to N_G do t[j]:= t[0] + j*hht;
//Центральный полином
 OprPol:=0; Nachusl; if flag then IterUt(true,K_iter_max,U0,C_last_C_,iterMinC)
 else IterUt(false,iterMin_C,U0,C_last_C_,fake);
//Сдвиг влево
 for i:=kl to N_G do xx[i]:= x[i-kl];
 for i:=1  to kl  do xx[kl-i]:=x[0]-i*hhx;
 for i:=0 to N_G do  x[i]:= xx[i];
 for i:=-1 downto -N_G do  x[i]:= x[0]-abs(i)*hhx;
 if (Nsl=-Ksl div 2) then begin OprPol:=-1; Nachusl end else NachuslNewLevo;
 if flag then IterUt(true,K_iter_max,U0L,C_last_L_,iterMinL)
 else IterUt(false,iterMin_L,U0L,C_last_L_,fake);
 for i:=0 to N_G do x[i]:= a0 + i*hhx;
//Сдвиг вправо
 for i:=0 to N_G-kr do xx[i]:= x[i+kr];
 for i:=0 to kr-1   do xx[N_G-i]:= x[N_G]+(kr-i)*hhx;
 for i:=0 to N_G    do x[i]:= xx[i];
 OprPol:=1; Nachusl;
 if flag then IterUt(true,K_iter_max,U0R,C_last_R_,iterMinR)
 else IterUt(false,iterMin_R,U0R,C_last_R_,fake);
 for i:=0 to N_G do x[i]:= a0 + i*hhx;
end;
procedure ViborN;
var Int1,Int2,Int3,Int1_prevN,Int2_prevN,Int3_prevN:extended;
 nev,Int_prevN, Int:extended;
begin
```

```
 x1_:=x[sdvigl]; x2_:=x[N_G-sdvigr];
 Int1:=integral(x[0],x1_,t[0],t[0]+line, C_last_L_);
 Int2:=integral(x1_,x2_,t[0],t[0]+line, C_last_C_);
 Int3:=integral(x2_, x[N_G],t[0],t[0]+line, C_last_R_);
 Int1_prevN:=integral(x[0],x1_prevN,t[0],t[0]+line, C_last_L_prevN);
 Int2_prevN:=integral(x1_prevN,x2_prevN,t[0],t[0]+line, C_last_C_prevN);
 Int3_prevN:=integral(x2_prevN, x[N_G],t[0],t[0]+line, C_last_R_prevN);
 Int:=abs(Int1+Int2+Int3);
 Int_prevN:=abs(Int1_prevN+Int2_prevN+Int3_prevN);
 nev:=abs(Int-Int_prevN)/(abs(Int_prevN)+1e-16);
 if nev<MinSum then begin MinSum:=nev; MinN_G:=N_G; iterMin_C:=iterMinC;
iterMin_L:=iterMinL;iterMin_R:=iterMinR; end;
end;
procedure RD; var Dteck:extended;
begin x[0]:=a0; t[0]:=c0;podT:=0; c00:=c0; d00:=c00+ht;
Dteck:=d_Gl+ht*(Ksl-Nsloya); writeln('sl_X= ',Nsloya,' Dteck= ',Dteck:7:5);
while (t[0]+line-Dteck) <=1e-15 do begin
  N_G:=Nmin; MinSum:=100;
  repeat
   dec(N_G); solve(true,Nsloya); // Решение для степени N-1
// Сохранение коэффициентов полиномов для степени N-1
C_last_L_prevN:=C_last_L_; C_last_C_prevN:=C_last_C_; C_last_R_prevN:=C_last_R_;
  x1_prevN:=x[sdvigl]; x2_prevN:=x[N_G-sdvigr];
   inc(N_G); solve(true,Nsloya); // Решение для степени N
ViborN; inc(N_G);
  until N_G>Nmax;
  iterMin_L:=iterMinL; iterMin_C:=iterMinC;iterMin_R:=iterMinR;
  writeln('sloy po X = ',Nsloya:2,'    sloy po T = ',podT:2,'   N= ',MinN_G:3,'
iter= ',iterMin_L:3,' ',iterMin_C:3,' ',iterMin_R:3);
  N_G:=MinN_G;
// Решиние с оптимальной степенью полинома
  if iterMin_C<K_iter_min then iterMin_C:=K_iter_min;
  if iterMin_L<K_iter_min then iterMin_L:=K_iter_min;
  if iterMin_R<K_iter_min then iterMin_R:=K_iter_min;
  solve(false,Nsloya); x1:=x[sdvigl]; x2:=x[N_G-sdvigr];
//Сохранение массивов для следующих слоев
C_last_L :=C_last_L_; C_last_C :=C_last_C_; C_last_R :=C_last_R_;
C_last_L[-3,0]:=x1; C_last_L[-4,0]:=x2;
  MasPolT[podT,1]:=C_last_L; MasPolT[podT,2]:=C_last_C;
  MasPolT[podT,3]:=C_last_R; c00:=c00+hht; d00:=c00+ht; t[0]:=t[0]+line;
  inc(podT); end;
MasGl[Nsloya]:=MasPolT;
end;
procedure outputGl; // Вывод результата
 var Nsl,j,i:shortint; Sum_err, err,xpr,tpr:extended;
begin Sum_err:=0; writeln;writeln('Output of results');
 for j:=0 to outputGl_t do begin
   tpr:=c_Gl+j*(d_Gl-c_Gl)/outputGl_t; podtGl:=trunc((tpr-c_Gl)/line);
   if (tpr<>c_Gl) and (abs(frac(tpr/line))<1e-18)  then dec(podtGl);
   writeln; writeln(' podT=',podtGl); write('press enter...');readln;
   for i:=0 to outputGl_x do begin
     xpr:=a_Gl+i*(b_Gl-a_Gl)/outputGl_x; Nsl:=trunc((xpr-a_Gl)/hx)+1;
     if (xpr<>a_Gl) and (abs(frac(xpr/hx))<1e-18) then Nsl:=Nsl-1;
     x1:=MasGl[Nsl,podTGl,1][-3,0]; x2:=MasGl[Nsl,podTGl,1][-4,0];
     if xpr<x1 then
      err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,1])-solution(xpr,tpr))
     else if xpr<x2 then
      err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,2])-solution(xpr,tpr))
      else err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,3])-solution(xpr,tpr));
writeln(x1:7:4,' ',x2:7:4,' Nsl=',Nsl,' t=',tpr:7:4,' x=',xpr:7:4,'  ',err);
    Sum_err:=Sum_err+err; end; end;
  writeln;writeln('Sredn_err= ',Sum_err/(outputGl_x*outputGl_t));
```

```
end;
procedure RD_GL;
 begin Ksl:=trunc(exp(kk*ln(2))); a0:=a_Gl; c0:=c_Gl; Nsloya:=-Ksl div 2;
  hx:=(b_Gl-a_Gl)/exp(kk*ln(2)); ht:=hx/2; line:=ht/10;
  a0:=a_Gl-(abs(Nsloya)+1)*hx;
  repeat RD; inc(Nsloya); a0:=a0+hx; until Nsloya>Ksl; end;
begin preparation; writeln('The calculation process is in progress.');
writeln('Please wait...'); tnach:=Gettime; RD_GL; tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec);
writeln; writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec);
outputGl; Dispose(DD_G);Dispose(MasGl); readln;end.
```

Согласно эксперименту программа *VPI_IVP_FOR_PDE* позволяет получить приближенное кусочно-интерполяционное решение задачи (4.23), характеризующееся сравнительно высокой точностью при небольших размерах прямоугольной области G. Данное утверждение детализируется непосредственно ниже при описании результатов численного эксперимента.

4.8. Данные результатов эксперимента. Эксперимент составляют примеры задач, имеющих точные решения, сравнением с ними определяется погрешность приближенных решений.

Пример 4.1. Задача Коши (4.81), имеющая точное решение $u(x,t)=\sin(x-t)$, рассматривается как модель для сравнения численных методов [168]. Ее приближенное решение ниже дано в $G=\{(x,t)\,|\,x\in[0,1],t\in[0,1]\}$. В табл. 4.1 приводятся значения абсолютной погрешности кусочно-интерполяционного решения с итерационным уточнением в 10 равномерно распределенных точках на отрезке $x\in[0,1]$ при значениях $t=0.1$ и $t=1.0$ (степень интерполяционного полинома $n=13$, число итераций – $r=50$, область G не делится на подобласти). В первом столбце таблицы – время решения на персональном компьютере (для сравниваемых методов время на том же компьютере дано в табл. 4.3, 4.4).

Таблица 4.1

Абсолютная погрешность приближенного решения задачи (4.81)

11 *s*	$h_x\approx 0.08$	$h_t\approx 0.04$	$t=0.1$	4.7**e-20**	2.7**e-20**	9.5**e-20**	…	2.2**e-19**	1.1**e-19**
			$t=1.0$	1.6**e-19**	6.5**e-19**	7.6**e-19**	…	6.8**e-21**	3.0**e-19**

Пример 4.2. Задача Коши

$$u'_t + e^{-t}u'_x = e^t x, \;\; u(x,0) = x + 1 + \sin(x+1), \tag{4.82}$$

имеет решение $u(x,t) = e^t x + 1 - t + \sin(x + e^{-t})$. Приближенное решение построено в квадрате G из примера 4.1. Табл. 4.2 содержит абсолютную погрешность предложенного метода в точках, распределенных как в примере 4.1 ($n = 13$, $r = 25$, число подобластей G_{ij} равно 4).

Таблица 4.2

Абсолютная погрешность приближенного решения задачи (4.82)

23 *s*	$h_x \approx 0.04$	$h_t \approx 0.02$	$t = 0.1$	5.4**e-19**	0.0**e+00**	2.2**e-19**	…	4.3**e-19**	0.0**e+00**
			$t = 1.0$	1.8**e-18**	3.3**e-19**	9.8**e-19**	…	6.5**e-19**	5.6**e-18**

В системах компьютерной математики решение рассматриваемых уравнений опирается на граничные условия. Для единства условий сравнения ниже даны решения начально-краевых задач в системах *MathCAD*, *Maple* и предложенным методом без его принципиальных изменений. В табл. 4.3, 4.4 строка (1) соответствует предложенному методу, (2) – функции *pdesolve* программы *MathCAD* (выбор разностного метода отвечает наилучшему приближению), (3) – *pdsolve* программы *Maple*. Соотношение шагов удовлетворяет условию сходимости метода сеток для уравнений гиперболического типа.

Пример 4.3. Рассматривается начально-краевая задача

$$u'_t + u'_x = 3\cos(x + 2t), \; u(x, 0) = \sin(x), \; u(0, t) = \sin(2t). \tag{4.83}$$

Погрешность ее решения в области из примера 4.1 сравнивается в десяти равномерно распределенных точках на отрезке $x \in [0, 1]$ при значении $t = 1$. В предложенном методе $n = 13$, $r = 20$, область G не делится на подобласти.

Таблица 4.3

Абсолютная погрешность приближенного решения задачи (4.83)

(1)	4 *s*	$h_x \approx 8.0 \times 10^{-2}$	$h_t \approx 4.0 \times 10^{-2}$	1.1**e-19**	1.1**e-19**	5.4**e-19**	…	5.8**e-19**	4.1**e-20**
(2)	14 *s*	$h_x \approx 3.3 \times 10^{-4}$	$h_t \approx 1.7 \times 10^{-4}$	6.1**e-07**	4.0**e-08**	6.6**e-08**	…	1.3**e-08**	2.2**e-07**
(3)	267 *s*	$h_x = 8.0 \times 10^{-5}$	$h_t = 4.0 \times 10^{-5}$	1.2**e-10**	2.0**e-10**	2.6**e-10**	…	4.2**e-11**	1.5**e-10**

Пример 4.4. В [169] рассматривается нелинейная краевая задача

$$A_2\frac{\partial u}{\partial t}+A_1\frac{\partial u^{m+1}}{\partial x}=A_3u^k+A_4e^{A_5u}+E(x,t),\ u(x,0)=e^{x/b},\ u(0,t)=e^t. \qquad (4.84)$$

Если решение взято в виде e^{t+y}, $y=x/b$, то источник E принимает вид

$$E(x,t)=e^{t+y}(A_2+A_1b^{-1}(m+1)e^{m(t+y)})-A_3e^{k(t+y)}-A_4e^{A_5e^{t+y}}.$$

В качестве опорных согласно [169] взяты значения: $m=0$, $A_1=A_2=1$, $A_4=A_5=0$ и $b=1$. В табл. 4.4 приводится погрешность приближенного решения при различных значениях параметров k и A_3 (первая строка таблицы) в десяти равномерно распределенных точках из примера 4.3. Остальные обозначения соответствуют табл. 4.3. В предложенном методе $n=13$, $r=20$, $r=30$, $r=35$ соответственно параметрам данных задач, деление G на подобласти не выполнялось.

Таблица 4.4

Абсолютная погрешность приближенного решения задачи (4.84)

$k=1,\ A_3=0$									
(1)	3 *s*	$h_x\approx 8.0\times10^{-2}$	$h_t\approx 4.0\times10^{-2}$	0.0**e+00**	1.5**e-18**	8.7**e-19**	…	0.0**e+00**	1.7**e-18**
(2)	24 *s*	$h_x=2.5\times10^{-4}$	$h_t\approx 1.3\times10^{-4}$	3.7**e-07**	1.4**e-06**	1.8**e-07**	…	1.5**e-07**	3.3**e-07**
(3)	267 *s*	$h_x=8.0\times10^{-5}$	$h_t=4.0\times10^{-5}$	3.6**e-10**	7.3**e-10**	1.1**e-09**	…	3.7**e-09**	4.3**e-09**

$k=1,\ A_3=-1$									
(1)	7 *s*	$h_x\approx 8.0\times10^{-2}$	$h_t\approx 4.0\times10^{-2}$	1.5**e-18**	1.1**e-18**	8.7**e-19**	…	2.2**e-18**	0.0**e+00**
(2)	25 *s*	$h_x=2.5\times10^{-4}$	$h_t\approx 1.3\times10^{-4}$	1.1**e-06**	9.9**e-07**	1.8**e-07**	…	5.0**e-07**	4.8**e-07**
(3)	284 *s*	$h_x=8.0\times10^{-5}$	$h_t=4.0\times10^{-5}$	6.1**e-10**	1.2**e-09**	1.7**e-09**	…	4.9**e-09**	5.5**e-09**

$k=2,\ A_3=-1$									
(1)	8 *s*	$h_x\approx 8.0\times10^{-2}$	$h_t\approx 4.0\times10^{-2}$	6.5**e-19**	4.3**e-19**	8.7**e-19**	…	5.2**e-18**	3.5**e-18**
(2)	107 *s*	$h_x=2.5\times10^{-4}$	$h_t\approx 1.3\times10^{-4}$	5.8**e-07**	8.6**e-07**	6.1**e-07**	…	9.9**e-07**	1.0**e-06**
(3)	3074 *s*	$h_x=8.0\times10^{-5}$	$h_t=4.0\times10^{-5}$	1.4**e-09**	2.4**e-09**	3.0**e-09**	…	6.4**e-09**	7.1**e-09**

В [169] приводится относительная погрешность 2.21% при $t=1$. Предложенный метод в этом случае дает $2.02\times10^{-16}\,\%$.

Пример 4.5. Рассматривается задача Коши [26]

$$\frac{\partial u}{\partial t}+\frac{\partial u}{\partial x}=0,\ u(x,0)=\begin{cases}0, & x\in(-\infty,10)\cup(30,+\infty),\\ u_0(x), & x\in[10,30],\end{cases}$$

где $u_0(x)=2^{-1}-2^{-1}\cos(10^{-1}\pi(x-10))$. Точное решение имеет вид:

$$u(x,t)=\begin{cases}0, & (x-t<10)\vee(x-t>30),\\ u_0(x-t), & 10\le x-t\le 30.\end{cases}$$

Область приближения $G=\{(x,t)\,|\,x\in[0,40],t\in[0,1]\}$. В точке с абсциссой $x=21$, соответствующей пику волны при $t=1$, абсолютная погрешность рекурсивной 5-точечной разностной схемы (*MathCAD*) составляет 6.15×10^{-7}, разностной схемы Кранка-Николсон (*Maple*) – 2.40×10^{-14}, предложенный метод в этой точке приближает решение с погрешностью 2.71×10^{-19}. Кусочная непрерывность приближения позволяет сравнительно адекватно отразить характер изменения решения в окрестности амплитуды (рис. 4.2).

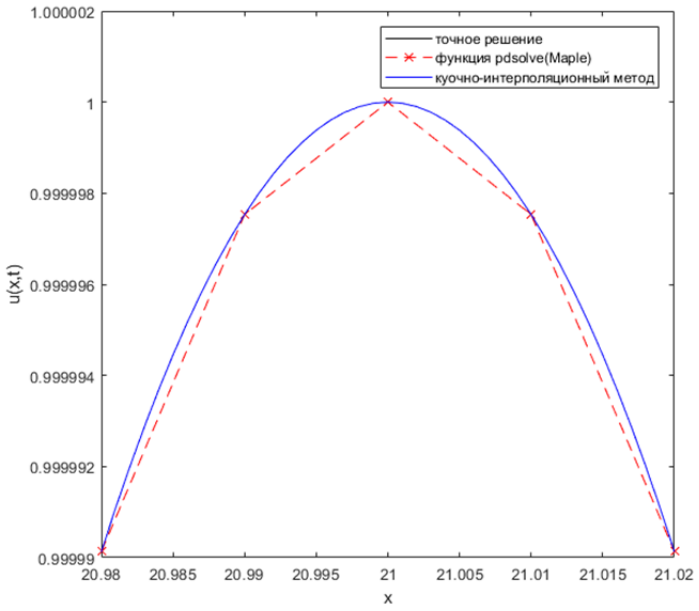

Рис. 4.2. Сравнение графиков точного решения задачи из примера 4.5 и его приближений по кусочно-интерполяционному методу и с помощью функции pdsolve (Maple) в окрестности амплитуды при $t=1$

Вследствие высокой точности и непрерывности график кусочно-интерполяционного приближения визуально не отличим от графика точного решения в выбранном на рис. 4.2 масштабе отображения. В начале и в конце волны погрешность можно снизить, окружив эти точки узкой областью с большим количеством подобластей. Так, в точке $x=11.2$, соответствующей окрестности начала волны при $t=1$, абсолютная погрешность кусочно-интерполяционного приближения составит 6.59×10^{-14}, погрешность приближения в этой же точке по разностной схеме Кранка-Николсон – 8.29×10^{-10}.

В аналогичных исследованиях точность данных численных экспериментов, как правило, не превосходят точность в строках (2), (3) табл. 4.3, 4.4. Так, в [170] строится решение гиперболического уравнения второго порядка с помощью полиномов Тейлора от двух переменных. Абсолютная погрешность этого метода в квадрате из примера 4.1 при $t=0$ составляет $3.97\times10^{-21`}$, но с ростом t растет до 7.61×10^{-9}. В [171] эта же задача решается на основе кубической тригонометрической сплайн интерполяции. В $G=\{(x,t)\mid x\in[0,2\pi], t\in[0,3]\}$ абсолютная погрешность приближения не ниже 5.43×10^{-7}. В [172] строится кусочно-интерполяционное решение уравнения переноса с помощью кубической B-сплайн квазиинтерполяции, ее абсолютная погрешность в квадрате из примера 4.1 не ниже 3.01×10^{-10} при $t=1$.

Сравнительная точность предложенного метода достигается за счет итерационного уточнения (не применяемого в работах, с которыми проводилось сравнение). Его реализация опирается на аналитический вид первообразной и производной от интерполяционного полинома, преобразованного в форму алгебраического полинома с числовыми коэффициентами. Для снижения погрешности существенны также сдвиги области приближения по горизонтали и вертикали, с их помощью обходится зона погрешности экстраполяции за пределами треугольного расположения узлов интерполяционного полинома.

Замечание 4.4. В табл. 4.1 – 4.4 время решения задачи предложенным методом дано без использования программного выбора степени полинома и числа итераций. С программным выбором n и r время существенно возрастает, наиболее долго решается задача (4.82) – $2\ \text{min}\ 41\ s$. В целом, с таким выбором время решения занимает промежуточное значение между *MathCAD* и *Maple* в границах погрешности, приведенной в таблицах.

Алгоритм выбора параметров можно выполнить параллельно по всем сравниваемым значениям невязки S_n одновременно для всех степеней полиномов n (в силу взаимной независимости сравниваемых значений), аналогично, по всем I_r для всех значений r. По параллельно найденным значениям без труда определяются параметры минимизации погрешности при наименьшей временной сложности. Вместе с тем вычислительный алгоритм решения рассматриваемой задачи Коши является взаимно независимым по временным слоям подобластей $\bigcup_{j=0}^{2^k-1} G_{ij}$, $i=\text{const}$, поэтому решение распараллеливается по всем номерам слоев $i\in\overline{0,\,2^k-1}$, ускоряя последовательный алгоритм пропорционально 2^k. Распараллеливание допускает также алгоритм вычисления коэффициентов и значения интерполяционного полинома Ньютона. Для полинома от одной переменной временная сложность оценивается в [132], для полинома от двух переменных – в [150].

4.9. Об аналогах обработки данных в ИВС для интегро-дифференциальных моделей периодических процессов. Интерполяционным полиномом (4.7), преобразованным к виду (4.9), приближаются частные производные решения уравнения гиперболического типа второго порядка и выше, итерационное уточнение в этом случае выполнимо с помощью первообразной соответственной кратности. Для линейного гиперболического уравнения второго порядка такой подход реализован в [150]. Формально

рассмотренный метод переносится на уравнения порядка выше первого на основе их преобразования в систему уравнений первого порядка. Пусть, например, рассматривается задача Коши для уравнения гиперболического типа

$$a u_{xx} - b u_{tt} + c u_x + d u_t + g u = f ,$$

где a, b, c, d, g, f – заданные функции независимых переменных x и t, $ab > 0$. Пусть приближенное решение ищется в прямоугольнике G из (4.1) – (4.3) при начальных условиях

$$u\big|_{t=0} = \varphi(x), \;\; u_t\big|_{t=0} = \psi(x), \quad a \le x \le b ,$$

где φ и ψ – заданные функции $x \in R$. Замена переменных влечет

$$u_x = y, \;\; u_t = z, \;\; a y_x - b z_t + c y + d z + g u = f ,$$

$$u\big|_{t=0} = \varphi(x), \;\; y\big|_{t=0} = \varphi'(x), \;\; z\big|_{t=0} = \psi(x), \quad a \le x \le b .$$

В каждой подобласти значения y и z из первых двух уравнений можно интерполировать полиномами (4.7) спреобразованием к виду (4.9). Для решения $u = g^{-1}(f - a\, y_x + b z_t - c y - d z)$ в качестве приближений частных производных подставляются соответственные частные производные преобразованных полиномов. Кроме того, используются полиномиальные приближения y, z. Процесс можно циклически повторять до достижения искомой точности. Вопрос о сходимости при данном подходе требует отдельного исследования.

Аналогично итерационному уточнению можно строить последовательность приближений решения интегро-дифференциальных уравнений. Пусть, например, рассматривается задача Коши для линейного уравнения с интегральным оператором типа Вольтерры

$$y'_x = A(x)\, y(x) + \int_a^x K(x, s)\, y(s)\, ds + f(x), \;\; y(a) = y_0 . \qquad (4.85)$$

Предполагается, что $A(x)$ и $f(x)$ непрерывны при $a \le x \le b$, $K(x,s)$ непрерывно при $a \le s \le x \le b$. В качестве нулевого приближения принимается $y_0(s) \equiv y_0$. С подстановкой $y_0(s)$ подынтегральную функцию можно интерполировать полиномом (4.7) с преобразованием к (4.9): $\Psi_{n0}(z,w) \approx K(x,s)\,y_0(s)$, где $z = h_x^{-1}(x - x_0)$, $w = h_s^{-1}(s - s_0)$ (здесь и ниже не индексируется подобласть и соответственные ей выражения). Тогда $y'_x \approx A(x)\,y_0 + \int_a^x \Psi_{n0}(z,w)\,ds + f(x)$. Если $s_0 = x_0 = a$, то

$$\int_a^x \sum_{\ell=0}^{n} \sum_{m=0}^{n-\ell} a_{\ell m}\, z^{\ell}\, w^{m}\, ds = h_s \sum_{\ell=0}^{n} \sum_{m=0}^{n-\ell} \frac{a_{\ell m}}{m+1} \frac{(x-a)^{\ell+m+1}}{h_x^{\ell}\, h_s^{m+1}}.$$

Правую часть приближения y'_x можно интерполировать полиномом Ньютона от одной переменной с равноотстоящими узлами, преобразовав его согласно (2.4) – (2.11) к виду алгебраического полинома с числовыми коэффициентами, $p_{(n-1)0}(t) = \sum_{\ell=0}^{n-1} a_{\ell}\, t^{\ell}$, $t = h^{-1}(x-a)$, h – шаг интерполяции. В результате $p_{(n-1)0}(t) \approx A(x)\,y_0 + h_s \sum_{\ell=0}^{n} \sum_{m=0}^{n-\ell} \frac{a_{\ell m}}{m+1} \frac{(x-a)^{\ell+m+1}}{h_x^{\ell}\, h_s^{m+1}} + f(x)$. Первообразная от $p_{(n-1)0}(t)$ принимается за первое приближение: $y_1(x) = y_0 + \int_a^x p_{(n-1)0}(h^{-1}(x-a))\,dx$, или, $y_1(x) = y_0 + h \sum_{\ell=0}^{n-1} \frac{a_{\ell}}{\ell+1} (h^{-1}(x-a))^{\ell+1}$.

Циклическое повторение описанного процесса с рекуррентными подстановками влечет

$$y_{r+1}(x) \approx y_0 + \int_a^x \left(A(x)\,y_r(x) + \int_a^x K(x,s)\,y_r(s)\,ds + f(x) \right) dx, \quad r = 0,1,\ldots$$

(знак $\approx$ ставится, поскольку не учитываются остаточные члены интерполяции), отсюда $y(x) \approx y_{r+1}(x)$. Сходимость метода требует отдельного исследования.

Задача (4.85) сводится к интегральному уравнению Вольтерры второго рода

$$y(x) = \lambda \int_a^x K(x, s)\, y(s)\, ds + f(x),$$

где $K(x, s)$ и $f(x)$ – непрерывные функции при $a \le s \le x \le b$. Метод последовательных приближений для этого уравнения сходится к единственному непрерывному решению. Если в качестве нулевого приближения взять $y_0(s) \equiv 0$, то $y_1(x) = f(x)$. Подынтегральную функцию с подстановкой y_1 можно интерполировать полиномом (4.7) и преобразовать его к виду (4.9): $\Psi_{n1}(z, w) \approx K(x, s)\, y_1(s)$, $z = h_x^{-1}(x - x_0)$, $w = h_s^{-1}(s - s_0)$. Отсюда $y(x) \approx \lambda \int_a^x \Psi_{n1}(z, w)\, ds + f(x)$. Если $x_0 = s_0 = a$, то

$$\int_a^x \Psi_{n1}(z, w)\, ds = h_s \sum_{\ell=0}^{n} \sum_{m=0}^{n-\ell} \frac{a_{\ell m}}{m+1} \frac{(x-a)^{\ell+m+1}}{h_x^{\ell}\, h_s^{m+1}}.$$

Правую часть приближения $y(x)$ можно интерполировать полиномом Ньютона от одной переменной, преобразовав его к виду $p_{n1}(t) = \sum_{\ell=0}^{n-1} a_\ell t^\ell$. Тогда $p_{n1}(t) \approx \lambda\, h_s \sum_{\ell=0}^{2n} \sum_{m=0}^{2n-\ell} \frac{a_{\ell m}}{m+1} \frac{(x-a)^{\ell+m+1}}{h_x^{\ell}\, h_s^{m+1}} + f(x)$, $t = h^{-1}(x - a)$, и в качестве второго приближения принимается $y_2(x) = p_{n1}(t)$. В продолжение процесса, с рекуррентными подстановками, получится

$$y_{r+1}(x) \approx \lambda \int_a^x K(x, s)\, y_r(s)\, ds + f(x), \quad r = 1, 2, \ldots,$$

откуда $y(x) \approx y_{r+1}(x)$. Сходимость в данном случае можно анализировать, сопоставляя $y_r(x)$ со сходящейся последовательностью приближений, каждое из которых имеет точное аналитическое выражение.

Для неоднородного уравнения Фредгольма 2-го рода $y(x) = \lambda \int_a^b K(x, s)\, y(s)\, ds + f(x)$ метод последовательных приближений сходится лишь при условии $|\lambda| < M^{-1}(b-a)^{-1}$, где $M = \sup_{x,s \in [a,b]} |K(x,s)|$. Поэтому аналог рассмотренной схемы допустимо строить только при выполнении этого условия.

Таким образом, в главе разработан, обоснован и программно реализован метод варьируемой кусочно-интерполяционной обработки данных модели переноса с итерационным уточнением. Дано приближенное решение задачи Коши для уравнения переноса, характеризующееся относительно высокой точностью и кусочно-непрерывным характером приближения. В случае линейного уравнения переноса метод равномерно сходится к решению в прямоугольной области. Приближение равномерно непрерывно в данной области, кусочно-непрерывно – при ее делении на подобласти. Метод распространяется на модели других классов, позволяет уточнять результаты обработки данных, детализировать и повысить качество моделирования процесса переноса. В главе также представлены аналоги предложенного метода обработки данных для моделей на основе интегральных и интегро-дифференциальных уравнений.

4.10. Выводы

1. Предложена варьируемая кусочно-интерполяционная обработка данных модели переноса, которая отличается от известных применением интерполяционного полинома Ньютона для двух переменных, программно преобразуемого в алгебраический полином с числовыми коэффициентами. Варьируемая интерполяция данных выполняется в каждой прямоугольной подобласти, на которые в зависимости от значений параметров автоматически делится исходная прямоугольная область. На этой основе получается гладкое приближение данных в каждой прямоугольной подобласти, которое отличает предложенный метод от известных, кроме того, метод отличается значительно более высокой точностью.

2. На основе алгебраической формы интерполяционного полинома Ньютона от двух переменных построено итерационное уточнение обработанных данных модели переноса, которое сходно с двумерным аналогом последовательных приближений Пикара. В результате относительно известных методов достигается дополнительное повышение точности кусочно гладкого аналитического приближения данных на несколько десятичных порядков.

3. Показана сходимость кусочно-интерполяционной обработки данных модели переноса, даны оценки скорости сходимости. Для случая прямоугольной области оценивается скорость сходимости итерационного уточнения. Для квазилинейной модели переноса предложена схема вывода аналогичных оценок.

4. Выполнена алгоритмизация и программная реализация обработки данных модели переноса, реализован автоматический выбор параметров, обеспечивающий наибольшую точность при наименьшем времени обработки. Программная реализация отличается устойчивостью и отмеченной точностью обработки, что в сочетании с кусочной гладкостью приближения данных позволяет детализировать и повысить качество моделирования процесса переноса.

5. Представлен численный эксперимент, согласно которому достигается превышение точности известных методов моделирования переноса на несколько десятичных порядков, что позволяет улучшить качество моделирования волновых процессов в ИВС.

6. Построены аналоги предложенного метода обработки данных для других классов моделей с частными производными, а также для моделей на основе интегральных и интегро-дифференциальных уравнений.

ГЛАВА 5. РАЗНОВИДНОСТИ МОДЕЛЕЙ ПЕРИОДИЧЕСКИХ ПРОЦЕССОВ В ИВС С ПРИМЕНЕНИЕМ КУСОЧНО-ИНТЕРПОЛЯЦИОННЫХ МЕТОДОВ ОБРАБОТКИ ДАННЫХ

В главе разработан комплекс программ обработки данных в ИВС на основе кусочно-интерполяционного метода для моделей жестких и нежестких задач, включающий автоматизированный и пользовательский выбор параметров для адаптации к классам моделей. С помощью комплекса выполнено моделирование периодических автоколебательных реакций, автоколебаний в задачах радиоэлектроники. Представлен комплекс программ и выполнено моделирование в ИВС движения КА, рассчитано уточненное время вывода КА на устойчивую периодическую орбиту. Выполнено моделирование возмущенного движения ИСЗ, выведенного на низкую околоземную орбиту. Для повышения качества программной обработки данных моделей выполнено построение библиотеки стандартных программ и вычисления специальных функций в ИВС на основе кусочно-интерполяционного метода. Обработка данных использует многообразие методов, в числе которых вычисление интеграла. Разработанный кусочно-интерполяционный метод используется для приближенного вычисления интеграла. Коэффициенты квадратурных формул не зависят от интегрируемой функции, отрезка интегрирования, сохраняются в разделе констант программы. Программный интерфейс стандартизируется до задания интегрируемой функции, отрезка интегрирования (как вариант может включать степень полинома и число подынтервалов). Метод распространяется на приближение первообразной функции. Одновременно с минимизацией погрешности минимизируется время вычисления интегралов и первообразных. Предложенный комплекс программ позволяет выявить неизвестные физические, химические, механические и другие свойства моделируемого процесса, а также существенно уточнить параметры модели.

Следует отметить, что кусочно-интерполяционное приближение для различных классов задач, включая задачи с неустойчивыми решениями и

жесткие задачи, выполняется на основе единой программы, переменные параметры которой настраиваются в зависимости от условий задачи. При необходимости повышения быстродействия метода, в частности при моделировании в реальном времени, автоматический программный подбор параметров может быть заменен пользовательским подбором и фиксированием значений параметров, что дополнительно исследовано в главе. В целом, не меняется метод, его алгоритмизация и реализующая программа.

На основе кусочной интерполяции с использованием интерполяционного полинома Лагранжа, преобразованного к виду алгебраического полинома с числовыми коэффициентами, в главе строится метод кусочно-интерполяционной обработки данных дифференциальных моделей. Алгоритм построения метода аналогичен представленному в главе 3 для интерполяционного полинома Ньютона. Представлено математическое описание метода, его алгоритмизация и программная реализация. Анализируется вычислительная устойчивость, границы погрешности и временная сложность для модельных задач небесной механики, задачи Коши для дифференциальных уравнений Бесселя и Гаусса. Описан численный эксперимент, исследуется возможность уменьшения погрешности без автоматического выбора параметров, изложенного в главе 3, с целью снижения трудоемкости метода. В случае вычисления специальной или стандартной функции, в частности, функции Бесселя и гипергеометрической функции, предложенный метод позволяет достигать временной сложности, измеряемой числом шагов схемы Горнера, что не достижимо с помощью известных разностных методов.

Материал главы соответствует содержанию публикаций автора [173 – 181].

5.1. Обработка данных в моделях химических и биохимических осцилляторов. Модели химических и биологичесих осцилляторов, как правило, приводят к задаче Коши для системы ОДУ вида

$$\frac{d\,\mathbf{y}}{d\,t} = \mathbf{f}(\mathbf{y}),\ \mathbf{y}(t_0) = \mathbf{y}_0, \tag{5.1}$$

где $\mathbf{y} = \left(y_1(t),\, y_2(t),\, \ldots,\, y_n(t)\right)$ – вектор концентраций, а вектор $\mathbf{f}(\mathbf{y}) = \left(f_1(\mathbf{y}),\, f_2(\mathbf{y}),\, \ldots,\, f_n(\mathbf{y})\right)$ описывает нелинейную реакционную кинетику или механизм, лежащий в основе колебаний, химических или биологических [59, 60]. При этом большинство задач являются жесткими или плохо обусловленными, например, модельная система Филда-Нойеса для реакции Белоусова-Жаботинского [59, 63], модель гликолиза [60, 182]. Однако имеют место и колебательные процессы, которые описываются нежестким или слабо жесткими моделями, например, модель Чумакова-Слинько для реакции окисления молекулярного водорода на поверхности никлевого или платиного катализатора [183], простейшая модель «брюсселятора» [35]. При численном приближении решения задачи (5.1), как было описано в главе 1, в зависимости от характера системы применяют различные методы интегрирования, при этом актуальными остаются проблемы повышения точности приближения и оптимального управления величиной шага интегрирования, а также создания единого метода инвариантного относительно вида системы.

5.1.1. Обработка числовых данных периодической реакции Белоусова-Жаботинского на модели Филда-Нойеса. В главе 1, п. 1.1.4, рассмотрены ключевые стадии реакции Белоусова-Жаботинского, являющейся одной из важных и часто применяемых для тестирования численных методов периодических реакций [3, 61]. Там же описана упрощенная модель колебательной реакции Белоусова-Жаботинского «орегонатор», предложенная Филдом и Нойесом [63], представлена безразмерная форма записи данной модели с параметрами, соответствующими устойчивому предельному циклу. Математическая модель с данными параметрами в виде

$$\left.\begin{aligned} &y_1' = 77.27\,(y_2 + y_1(1 - 8.375\cdot 10^{-6}\, y_1 - y_2)),\\ &y_2' = 77.27^{-1}(y_3 - y_2(1+y_1)),\ \ y_3' = 0.161(y_1 - y_3),\\ &y_1(0) = 1,\ \ y_2(0) = 2,\ \ y_3(0) = 3, \end{aligned}\right\} \qquad (5.2)$$

применялась в [3] при тестировании численных методов для решения жестких задач. На рис. 5.1 представлена визуализация результатов численного моделирования задачи (5.2) на интервале $t \in [0, 360]$ с применением кусочно-интерполяционного метода.

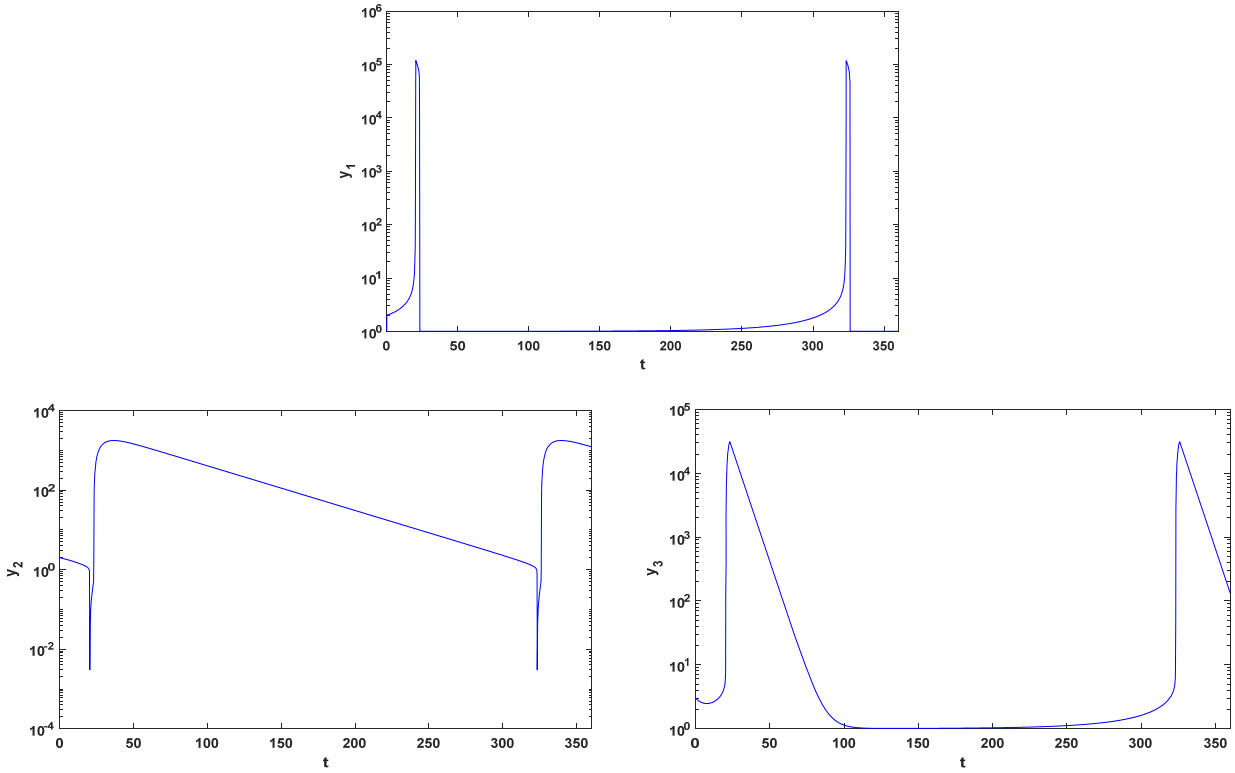

Рис. 5.1. Визуализация релаксационных автоколебаний значений компонентов системы (5.2) на основе кусочно-интерполяционного приближения решения на интервале $t \in [0, 360]$

Концентрации компонентов системы характеризуются периодическими высокоамплитудными быстрыми скачками. На рис. 5.2 представлены фазовые портреты системы (5.2), представляющие собой сложные предельные циклы. Характер и форма колебаний согласуются с результатами, представленными в работе [63].

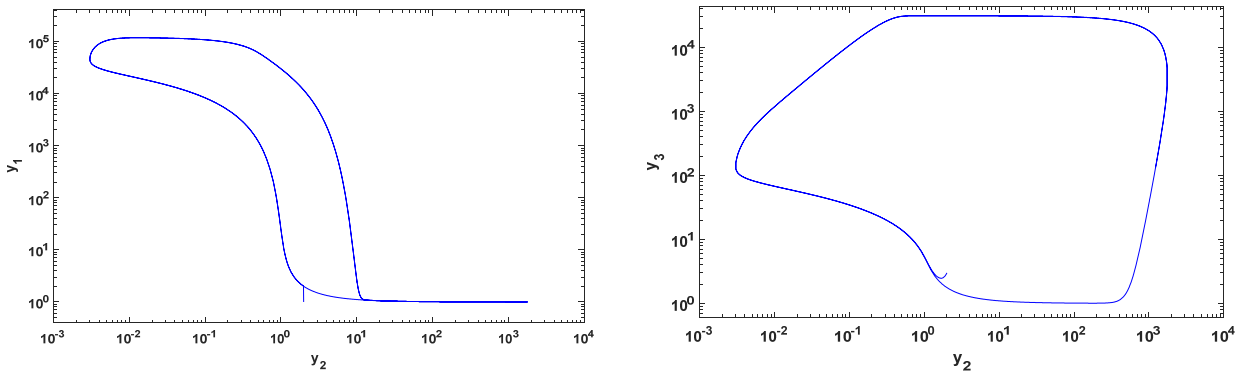

Рис. 5.2. Фазовые портреты системы (5.2), полученные на основе кусочно-интерполяционного приближения решения

Программный код кусочно-интерполяционного решения задачи (5.2) представлен в листинге 5.1.

Листинг 5.1

```
Program PP_BZH_listing51; {$APPTYPE CONSOLE} uses Math, SysUtils;
var kiter,koutput,k_:integer; Npol:byte;
 ynach1, ynach2, ynach3, Anach, Bkonech,velint:extended;s_rez:text; tt:longint;
function f1(x,y1,y2,y3:extended):extended;
begin f1:=77.27*(y2+y1*(1-8.375*1e-6*y1-y2)) end;
function f2(x,y1,y2,y3:extended):extended; begin f2:=(y3-(1+y1)*y2)/77.27 end;
function f3(x,y1,y2,y3:extended):extended; begin f3:=0.161*(y1-y3) end;
procedure RD(y1_nach,y2_nach,y3_nach,A_nach,B_konech,vel_int:extended;
n:byte;k,k_iter,k_output:integer;tt_:longint);
const nn=4;
type matr=array[0..nn,0..nn] of extended; vect=array[0..nn] of extended;
     matrC=array[-5..nn+1] of extended; matrAll= array[0..1200] of matrC;
var d:matr; CC1,CC2,CC3:matrC; xx:vect;
 a0,b0,h,x,y01,y02,y03,PogrFun1,PogrFun2,PogrFun3,PogrFun:extended;
 i,pod1,pod2:integer; kk,m:longint; hour,minut,sec,msec:word;
 tnach,tkonech:extended; Ck1,Ck2,Ck3,Ck11,Ck21,Ck31:matrAll;
//Рассчет коэффициентов полинома
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr;
begin e[1,1]:=1; e[1,0]:=0; for k:=2 to n do
  begin e[k,0]:=-e[k-1,0]*(k-1); for i:=1 to k-1 do
        e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1] end;
   for k:=1 to n do for i:=0 to k do d[i,k]:=e[k,i]
  end;
//вычисление конечных разностей
procedure Konech_Raznost(fy1,fy2,fy3:vect; n:byte; var dy1,dy2,dy3:matr);
var i,j:byte;
begin for j:=0 to n-1 do begin dy1[1,j]:=fy1[j+1]-fy1[j];
 dy2[1,j]:=fy2[j+1]-fy2[j]; dy3[1,j]:=fy3[j+1]-fy3[j];  end;
 for i:=2 to n do for j:=0 to n-i do
  begin dy1[i,j]:=dy1[i-1,j+1]-dy1[i-1,j];
  dy2[i,j]:=dy2[i-1,j+1]-dy2[i-1,j]; dy3[i,j]:=dy3[i-1,j+1]-dy3[i-1,j]; end;
end;
//построение полинома Ньютона в канонической форме по узловым значениям
procedure Newton(U1,U2,U3:Vect; n:byte; var Mcoef1,Mcoef2,Mcoef3:matrC);
var b1, b2, b3:vect; s1, s2, s3, p:extended; j, i:byte; dy1, dy2, dy3:matr;
begin Konech_Raznost(U1, U2, U3, n, dy1, dy2, dy3); p:=1;
 for j:=1 to n do begin
  p:=p*j; b1[j]:=dy1[j,0]/p; b2[j]:=dy2[j,0]/p; b3[j]:=dy3[j,0]/p;  end;
 Mcoef1[0]:=U1[0]; Mcoef2[0]:=U2[0]; Mcoef3[0]:=U3[0];
```

```
 for i:=1 to n do begin s1:=0;s2:=0;s3:=0;
  for j:=i to n do
   begin s1:=s1+d[i,j]*b1[j]; s2:=s2+d[i,j]*b2[j]; s3:=s3+d[i,j]*b3[j]; end;
  Mcoef1[i]:=s1; Mcoef2[i]:=s2; Mcoef3[i]:=s3;
endend;
//вычисление значения полинома по схеме Горнера
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
 for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
//построение приближения на текущем интервале
procedure Subinterval(k_,n,K_it:integer; a0,b0,Ynach1,Ynach2,Ynach3:extended;
var Ck1_,Ck2_,Ck3_:matrAll);
var hpd,a00,b00,y01,y02,y03,h:extended; m,pod:longint; x:vect; j:byte;
   i,iter,r:integer; fy1,fy2,fy3,y1,y2,y3:vect; C1,C2,C3:matrC;
   A1,A2,A3:matrC; t,pp1,pp2,pp3:extended;
Begin hpd:=(b0-a0)/exp(k_*ln(2)); a00:=a0;b00:=a00+hpd;
y01:=Ynach1;y02:=Ynach2;y03:=Ynach3;x[0]:=a0;m:=0; pod:=0;
while a00<=b0-hpd/2 do begin
  h:=(b00-a00)/n; for j:=1 to n do begin inc(m); x[j]:=a0+m*h end;
  for i:=0 to n do begin y1[i]:=y01; y2[i]:=y02; y3[i]:=y03; end;
  fy1[0]:=f1(x[0],y01,y02,y03); fy2[0]:=f2(x[0],y01,y02,y03);
  fy3[0]:=f3(x[0],y01,y02,y03);
  for iter:=1 to K_it do begin
    for i:=0 to n do
     begin fy1[i]:=f1(x[i],y1[i],y2[i],y3[i]);
      fy2[i]:=f2(x[i],y1[i],y2[i],y3[i]);fy3[i]:=f3(x[i],y1[i],y2[i],y3[i]);
     end;
    Newton(fy1,fy2,fy3,n,A1,A2,A3);
    for i:=1 to n do begin
      t:=(x[i]-x[0])/h; pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1); pp3:=a3[n]/(n+1);
      for r:=n-1 downto 0 do begin
       pp1:=pp1*t+A1[r]/(r+1); pp2:=pp2*t+A2[r]/(r+1); pp3:=pp3*t+A3[r]/(r+1);
      end;
      y1[i]:=pp1*h*t+y1[0]; y2[i]:=pp2*h*t+y2[0]; y3[i]:=pp3*h*t+y3[0];
   end; end;
  C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;
  C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h; C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h;
  C3[0]:=y3[0]; C3[-1]:=x[0]; C3[-2]:=h; C3[-3]:=n+1;C3[-4]:=k;C3[-5]:=n*h;
  for i:=1 to n+1 do
   begin C1[i]:=A1[i-1]*h/i; C2[i]:=A2[i-1]*h/i; C3[i]:=A3[i-1]*h/i; end;
  Ck1_[pod]:=C1; Ck2_[pod]:=C2; Ck3_[pod]:=C3;
  y01:=y1[n]; y02:=y2[n]; y03:=y3[n]; x[0]:=x[n];
  inc(pod); a00:=a00+hpd; b00:=a00+hpd
 end end;
begin
 Viet(nn,d); writeln('t':6,'AbsErr':25); writeln;
 assign(s_rez,'result_BZH.dat'); rewrite(s_rez);
 tnach:=Gettime;kk:=0; a0:=A_nach; b0:=a0+vel_int;
 y01:=y1_nach; y02:=y2_nach; y03:=y3_nach;
 while a0 <= B_konech-vel_int/2 do begin
  Subinterval(k,n,k_iter,a0,b0,y01,y02,y03,Ck1,Ck2,Ck3);
  Subinterval(k+1,n,k_iter,a0,b0,y01,y02,y03,Ck11,Ck21,Ck31);
  kk:=kk+1;if kk=tt_ then begin
//Вывод значений погрешности приближения в проверочных точках
x:= b0; pod1:=trunc(exp(k*ln(2)))-1; pod2:=trunc(exp((k+1)*ln(2)))-1;
   PogrFun1:=abs(Gorner(Ck11[pod2],x)-Gorner(Ck1[pod1],x));
   PogrFun2:=abs(Gorner(Ck21[pod2],x)-Gorner(Ck2[pod1],x));
   PogrFun3:=abs(Gorner(Ck31[pod2],x)-Gorner(Ck3[pod1],x));
   if PogrFun1>PogrFun2 then PogrFun:=PogrFun1 else PogrFun:=PogrFun2;
   if PogrFun3>PogrFun then PogrFun:=PogrFun3; writeln(x:6:1,'  ',PogrFun);
//Сохранение результатов приближения в файл для визуализации
{   for i:=0 to k_output-1 do begin
```

```
    x:= a0+i*Vel_int/k_output;pod1:=trunc((x-a0)/Ck1[0,-5]);
    writeln(s_rez, x,' ',log10(Gorner(Ck1[pod1],x)),' ',
    log10(Gorner(Ck2[pod1],x)),' ',log10(Gorner(Ck3[pod1],x))); end;}
  kk:=0; end;
  pod1:=trunc(exp(k*ln(2)))-1; y01:=Gorner(Ck1[pod1],b0);
  y02:=Gorner(Ck2[pod1],b0); y03:=Gorner(Ck3[pod1],b0);
  a0:=a0+vel_int; b0:=a0+vel_int;
 end;
tkonech:=Gettime; DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec);
writeln; writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec);
close(s_rez); end;
begin
 Anach:=0; Bkonech:=360; ynach1:=1;ynach2:=2;ynach3:=3;
 velint:=0.01; Npol:=3; k_:=9; kiter:=5; tt:=3000; koutput:=512;
 RD(ynach1,ynach2,ynach3,Anach,Bkonech,velint,Npol,k_,kiter,koutput,tt);
readln; end.
```

В данной программе помимо фиксирования параметров метода и исключения дополнительных уточняющих процедур выполенены изменения в подпрограмме вычисления значений коэффициентов интерполяционных полиномов *Newton* и подпрограмме для расчета значений конечных разностей *Konech_Raznost*, что позволило значительно сократить время решения задачи при сохранении высокой точности приближения (табл. 5.1). Именно, в программе, представленной в листинге 3.2, подпрограммы *Newton* и *Konech_Raznost*, вызывались для каждого уравнения системы отдельно; в листинге 5.1 подрограммы *Newton* и *Konech_Raznost* производят вычисления требуемых значений сразу для всех компонентов системы.

Каноническая норма абсолютной погрешности приближения решения задачи 5.2 и время ее достижения на основе кусочно-интерполяционного и разностных методов представлены в табл. 5.1.

Таблица 5.1

Абсолютная погрешность и время приближения решения задачи (5.2)

t	*ode15s* (*MATLAB*)	*Radau* (*SciPy*)	*Dormand-Prince_8*	*PI_S3*
	3:328	32:941	35:142	56:393
30.0	9.959**e-11**	2.419**e-10**	3.126**e-13**	4.441**e-15**
60.0	1.977**e-09**	5.912**e-12**	8.799**e-14**	3.331**e-16**
…	…	…	…	…
150.0	8.839**e-12**	3.268**e-13**	1.071**e-14**	8.327**e-17**
180.0	8.654**e-12**	1.279**e-13**	7.301**e-15**	2.776**e-17**
…	…	…	…	…
330.0	4.906**e-09**	2.801**e-10**	1.126**e-12**	1.421**e-14**
360.0	7.051**e-08**	2.501**e-12**	6.286**e-13**	1.665**e-15**

Высоким быстродействием, как и в случае приближения решения жесткой задачи (2.50) (см. глава 3, табл. 3.5), характеризуется функция *ode15s (MATLAB)*, реализующая метод конечных разностей переменного порядка с переменным шагом в сочетании с методом Гира для решения жестких задач (время решения $\approx 3s$). Вместе с тем, также как и в случае решения задачи (2.50), граница абсолютной погрешности данного метода не превышает порядка 10^{-8} (табл. 3.4, табл. 5.1). Применение функции *Radau* (библиотека *SciPy*), реализующей метод Радо IIA 5-го порядка [3], позволило повысить точность приближения на один десятичный порядок при этом время расчета составило $\approx 33s$.Наименьшей границей абсолютной погрешности 10^{-12}, среди представленных в табл. 5.1 разностных методов, характеризуется метод *Dormand-Prince_8.* Граница погрешности кусочно-интерполяционного приближения не превышает порядка 10^{-14} на всем отрезке приближения (табл. 5.1), при этом время кусочно-интерполяционного решения задачи (5.2) в сравнении в методом *Dormand-Prince_8* больше на $\approx 66\%$. Таким образом, ценой сравнительных затрат времени достигается превышение точности известных методов на два и более десятичных порядков.

Необходимы следующие пояснения к методике расчета значений абсолютной погрешности приближения в табл. 5.1. В случае, когда не известно аналитическое решение системы, погрешность приближения принято вычислять по способу Рунге [35]. Способ состоит в повторении вычислений с уменьшенной вдвое длиной шага и сравнении результатов: те десятичные знаки, которые не изменились, считаются верными. В случае кусочно-интерполяционного приближения аналогом уменьшения длины шага вдвое является увеличение вдвое числа отрезков разбиения ($P = 2^k$ из (3.3)). Для разностных методов с адаптивным выбором величины шага (*ode15s* (*MATLAB*) и *Radau* (*SciPy*) в табл. 5.1) результаты приближений со значением границы абсолютной погрешности $atol = 10^{-15}$ сравнивались с результатами, полученными при $atol = 10^{-16}$.

В случае кусочно-инерполяционного приближения решения задачи Коши, характеризующейся высокой степенью жесткости, отрезок $[\alpha_r, \beta_r]$ следует брать длиной меньше единицы, чтобы вариации n, k, ℓ отвечали частому изменению амплитуд производных решения. Параметры кусочно-интерполяционного метода: длина отрезка $\beta_r - \alpha_r = 0.01$, число итераций $\ell_{start} = 5$, степень полинома $n = 3$, логарифм числа отрезков разбиения $k = 9$. Автоматизированный выбор параметров и дополнительное итерационное уточнение исключены для ускорения процесса. Эксперимент проводился с ПК на базе процессора *Intel(R) Core(TM) i5-4460*.

Замечание 5.1. Следует отметить, что порядки значений погрешностей приближения в проверочных точках, представленные в табл. 5.1, не сохраняются при оценке погрешности вычисления требуемых характеристик автоколебательной реакции, таких как амплитуда, период колебаний значений концентраций, экстремальные значения концентраций реагентов.

Так, приближению задачи (5.2) с помощью функции *ode15s* (*MATLAB*) при заданном значении абсолютной погрешности $atol = 10^{-15}$ соответствуют следующие значения амплитуд колебаний переменных y_1, y_2, y_3:

$$A_{y_1} = 117844.776988242,\ A_{y_2} = 1768.694841439,\ A_{y_3} = 31262.838419668.$$

При этом приближению с помощью этой функции в тех же условиях, но при задании границы погрешности $atol = 10^{-14}$, соответствуют значения амплитуд:

$$A_{y_1} = 117844.777195366,\ A_{y_2} = 1768.694083568,\ A_{y_3} = 31262.838420083.$$

Согласно изложенному можно сделать вывод о том, что применение функции *ode15s* (*MATLAB*) позволило вычислить амплитуды колебаний значений переменных системы (5.2) с границей абсолютной погрешности не превышающей $\approx 7.6 \times 10^{-4}$. Аналогичные вычисления для функции *Radau* (*SciPy*):

	$atol = 10^{-15}$	$atol = 10^{-14}$	Δ
A_{y_1}	117844.77795497948	117844.77840133064	$\approx 4.5 \times 10^{-4}$
A_{y_2}	1768.6948424599038	1768.6948109623268	$\approx 3.1 \times 10^{-5}$
A_{y_3}	31262.83841135556	31262.838420303917	$\approx 8.9 \times 10^{-6}$

дают границу абсолютной погрешности приближения амплитуд колебаний значений переменных $\approx 4.5 \times 10^{-4}$. Метод *Dormand-Prince_8* характеризуется границей погрешности приближения амплитуд $\approx 9.6 \times 10^{-7}$ (сравнивались значения, полученные при $h = 10^{-5}$ и $h = 10^{-6}$):

	$h = 10^{-5}$	$h = 10^{-6}$	Δ
A_{y_1}	117844.777834722	117844.777835623	$\approx 9.0 \times 10^{-7}$
A_{y_2}	1768.69484279637	1768.69484279652	$\approx 1.5 \times 10^{-10}$
A_{y_3}	31262.8384193544	31262.8384203155	$\approx 9.6 \times 10^{-7}$

Кусочно-интерполяционный метод при всех вариациях (включая повышение числа отрезков разбиения $P = 2^k$ из (3.3)) даёт значения амплитуд колебания переменных с границей абсолютной погрешности не превышающей 2×10^{-13}:

	$k = 9$	$k = 10$	Δ
A_{y_1}	117844.77783562320220	117844.7778356232020	2×10^{-13}
A_{y_2}	1768.694842796521840	1768.694842796521870	3×10^{-14}
A_{y_3}	31262.8384203155070	31262.83842031550690	1×10^{-13}

Относительно низкая погрешность кусочно-интерполяционного приближения позволила повысить точность значений концентрации реагентов автоколебательной реакции и, как следствие, точность значений амплитуд и периода колебаний концентраций. Уточненные значения амплитуд колебаний значений переменных системы (5.2) y_1, y_2, y_3:

$$A_{y_1} = 117844.777835623,\ A_{y_2} = 1768.69484279652,\ A_{y_3} = 31262.8384203155.$$

что соответствует следующим физическим значениям амплитуд колебаний концентраций реагентов в модели «орегонатор» (см. глава 1, п. 1.1.4):

$$A_{[HBrO_2]} = 5.92170008624007 \times 10^{-6} M,\ A_{[Br^-]} = 5.30608452838957 \times 10^{-4} M,$$

$$A_{[Ce^{4+}]} = 7.54059662698010 \times 10^{-4} M,$$

где $A_{[HBrO_2]}$ – амплитуда изменения концентрации автокатализатора $HBrO_2$, $A_{[Br^-]}$ – амплитуда изменения концентрации бромид-иона Br^-, $A_{[Ce^{4+}]}$ – амплитуда изменения концентрации катализатора Ce^{4+} во времени.

Таким образом, уточняются физико-химические параметры моделируемого процесса. Помимо уточнения амплитуд колебаний концентрации реагентов химических реакций сравнительно точные кусочно-интерполяционные приближения экстремальных концентраций реагентов уточняют и период автоколебательной реакции. На основе кусочно-интерполяционного метода получено следующее значение периода изменения концентраций реагентов в системе (5.2): $T = 302.85804$. Данное значение согласуется и уточняет значение периода, рассчитанное в [63] с применением метода Гира, – $T = 302.9$.

Одной из важных характеристик реакции Белоусова-Жаботинского является минимальное значение концентрации бромид-иона Br^-, с момента возникновения которого в реакции начинает преобладать процесс II, представляющий собой реакцию взаимодействия бромат-иона с малоновой кислотой в присутствии катализатора иона Ce^{3+} с образованием броммалоновой кислоты и ионов Ce^{4+} (см. [63] и глава 1, п. 1.1.4). На основе кусочно-интерполяционного метода минимальное значение переменной $y_2 = 3.02455038486434 \times 10^{-3}$ возникает при $t = 323.273324$, что соответствует $[Br^-] = 9.07365115459302 \times 10^{-10} M$.

Таким образом, и в этой моделируемой реакции физико-химические параметры уточняются при помощи кусочно-интерполяционного решения дифференциальной модели.

По замечанию 3.3 кусочно-интерполяционное приближение непрерывно и непрерывно дифференцируемо на всем отрезке интегрирования. Отсюда достигается важная возможность моделирования реакции [3] во всех точках, промежуточных между разностными значениями. Свойство непрерывности кусочно-интерполяционного метода ниже используется при масштабировании графика зависимости переменной y_1 (соответствующей концентрации автокатализатора $HBrO_2$) с целью анализа поведения функции в окрестности точки локального максимума (рис. 5.3).

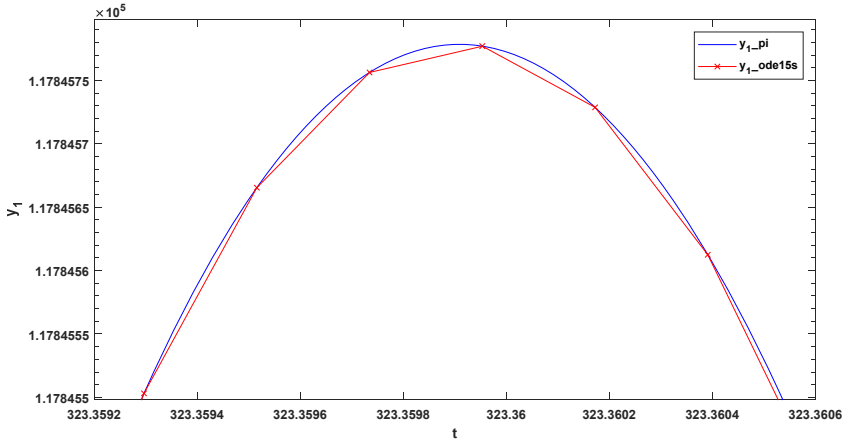

Рис. 5.3. Сравнение графиков изменения переменной y_1 по кусочно-интерполяционному методу (y_1_pi) и с помощью функции *ode15s* (y_1_ode15s)

Обозначение y_1_ode15s соответствует приближению с помощью функции *ode15s* (*MATLAB*), обозначение y_1_pi соответствует кусочно-интерполяционному методу. Из рисунка видно, что кусочно-интерполяционное приближение показывает гладкость (непрерывность решения и скорости его изменения) процесса изменения концентрации автокатализатора $HBrO_2$.

Таким образом, предложенный кусочно-интерполяционный метод позволяет уточнить параметры моделируемой реакции по сравнению с известными разностными методами, повышая точность численного моделирования автоколебательной химической реакции. В частности, на рис. 5.3 видно уточнение, достигаемое вследствие непрерывности кусочно-интерполяционного приближения решения (y_1_pi), в отличие от разностного

решения (y_1 _ *ode 15s*) на основе известного метода конечных разностей переменного порядка с переменным шагом в сочетании с методом Гира [54, 135].

5.1.2. Обработка числовых данных модели автоколебаний в системе гликолиза. Колебательные реакции в системе гликолиза были сначала предсказаны на математической модели Хиггинсом (1964), а после зарегистрированы экспериментально с помощью метода дифференциальной спектрофотометрии в лаборатории Б. Чанса (1966). В процессе гликолиза осуществляется распад глюкозы и других сахаров, при этом соединения, содержащие шесть молекул углерода, превращаются в трикарбоновые кислоты, включающие три молекулы углерода [60]. За счет избытка свободной энергии в процессе гликолиза на одну молекулу шестиуглеродного сахара образуются две молекулы АТФ. Основную роль в генерации наблюдаемых колебаний концентраций компонентов реакции: фруктозо-6-фосфата, фруктозо-1,6-фосфата и восстановленного НАД играет ключевой фермент гликолитического пути – фосфофруктокиназа (ФФК). Полная схема гликолитических реакций представлена в [60], упрощенная схема реакций представлена на рис. 5.4.

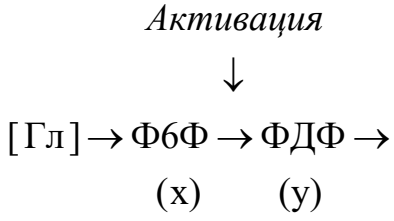

Рис. 5.4. Упрощенная схема реакций гликолиза

На рис. 5.4 приняты следующие обозначения: [Гл] – глюкоза, Ф6Ф – фруктозо-6-фосфат субстрат ключевой реакции, ФДФ – продукт этой реакции (фруктозодифосфат), который является субстратом в следующей стадии. Обе реакции катализируются ферментами. В безразмерных переменных простейшая математическая модель гликолиза выражается системой уравнений [182]:

$$\left.\begin{aligned} y_1' &= 1 - y_1 y_2, \\ y_2' &= \alpha y_2 \left(y_1 - \frac{1+\beta}{y_2+\beta} \right), \end{aligned}\right\} \tag{5.3}$$

предложенной Дж. Хиггинсом. Для получения решения системы в виде релаксационных автоколебаний (жесткий предельный цикл) используют следующие параметры: $\beta = 10$, $\alpha = 100$ [184]. Моделирование проводится при начальных условиях $y_1(0) = 1$, $y_2(0) = 0.001$, на интервале $[0, 20]$. На рис. 5.5 представлена визуализация результатов численного моделирования задачи (5.3) при данных значениях параметров с применением кусочно-интерполяционного метода.

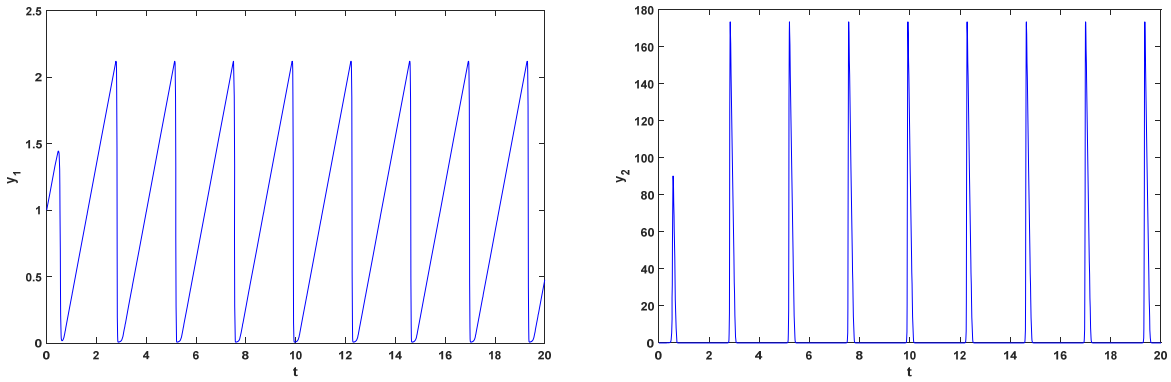

Рис. 5.5. Визуализация релаксационных автоколебаний значений компонентов системы (5.3) на основе кусочно-интерполяционного приближения на интервале $t \in [0, 20]$ при $\beta = 10$, $\alpha = 100$, $y_1(0) = 1$, $y_2(0) = 0.001$

Концентрации компонентов системы характеризуются периодическими высокоамплитудными быстрыми скачками с постоянным периодом и амплитудой (релаксационные автоколебания), что согласуется с результатами, представленными в работе [184].

Сравнительно высокая точность и непрерывность кусочно-интерполяционного решения позволяют провести детальный анализ кинетики изменения концентраций фруктозо-6-фосфата (y_1) и фруктозодифосфата (y_2) в окрестностях точек их локальных максимумов (рис. 5.6).

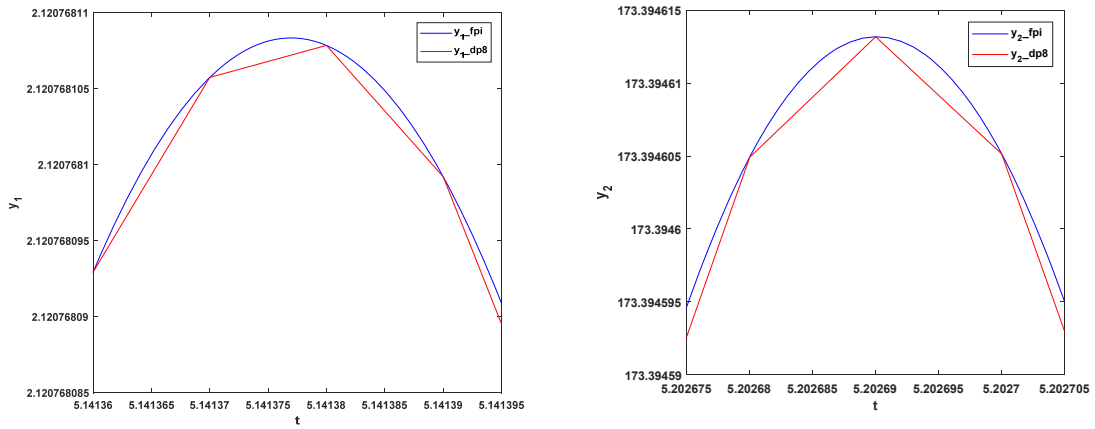

Рис. 5.6. Сравнение кинетики изменения концентраций реагентов реакций гликолиза на основе кусочно-интерполяционного решения системы (5.3) (y_1_fpi и y_2_fpi) и разностного решения по методу *Dormand-Prince_8* (y_1_dp8 и y_2_dp8)

Обозначения y_1_dp8, y_2_dp8, соответствуют приближению с помощью метода *Dormand-Prince_8,* обозначения y_1_fpi, y_2_fpi, соответствуют кусочно-интерполяционному приближению. Из рисунка видно, что кусочно-интерполяционное приближение показывает гладкость процесса изменения концентраций фруктозо-6-фосфата и фруктозодифосфата в отличие от дискретного приближения на основе известного метода *Dormand-Prince_8*.

Таким образом, в моделируемой автоколебательной химической реакции предложенный кусочно-интерполяционный метод позволяет не только уточнить параметры реакции по сравнению с известными разностными методами, повысить точность численного моделирования, но кроме того позволяет обнаружить качество гладкости зависимости изменения концентраций реагентов, которое не обнаруживается при разностном численном моделировании.

Следует отметить, что кусочно-интерпоряционное приближение значений концентрации реагентов характеризуется сравнительно высокой точностью и при сравнении значений в узлах разностных методов. В табл. 5.2 в 100 равномерно распределенных проверочных точках интервала $t \in [0, 20]$ представлены значения канонической нормы абсолютной погрешности приближения решения системы 5.3 при $\beta = 10$, $\alpha = 100$ на основе кусочно-

интерполяционного и разностных методов. Во второй сверху строке таблицы представлено время приближенного решения на персональном компьютере.

Таблица 5.2

Абсолютная погрешность и время приближения решения задачи (5.3)

t	*ode15s* (*MATLAB*)	*Dormand-Prince 8*	*FPI_S2*
	15:160	0:128	2:240
0.2	3.329**e-15**	8.782**e-18**	1.084**e-18**
0.4	1.842**e-13**	1.362**e-17**	1.084**e-19**
0.6	8.271**e-11**	5.683**e-15**	2.082**e-17**
…	…	…	…
9.8	2.539**e-11**	1.437**e-15**	2.602**e-18**
10.0	7.266**e-08**	1.298**e-12**	6.939**e-18**
10.2	8.009**e-11**	1.423**e-15**	8.809**e-20**
…	…	…	…
16.8	6.743**e-11**	2.507**e-15**	2.168**e-19**
17.0	6.945**e-08**	8.385**e-13**	9.714**e-17**
17.2	8.033**e-08**	1.099**e-12**	3.036**e-18**
…	…	…	…
19.6	6.780**e-09**	6.375**e-14**	1.636**e-19**
19.8	3.025**e-10**	2.855**e-15**	8.131**e-20**
20.0	3.025**e-10**	2.861**e-15**	9.758**e-19**

Погрешность приближения вычислялась по правилу Рунге, так же, как и при решении задачи (5.2) (табл. 5.1). Параметры кусочно-интерполяционного метода в соответствии со степенью жесткости задачи выбраны следующие: $\beta_r - \alpha_r = 0.01$, $\ell_{start} = 7$, $n = 4$, $k = 8$. В среде *MATLAB* решение данной задачи удалось вычислить с границей абсолютной погрешности порядка 10^{-8} (табл. 5.2) на основе функции *ode15s* при фиксировании максимального значения величины шага интегрирования ($MaxStep = 10^{-4}$), также в качестве параметра функции была задана матрица Якоби системы (что не требуется при использовании кусочно-интерполяционного метода). Метод *Dormand-Prince_8* характеризуется границей погрешности 10^{-12} при $h = 10^{-4}$, дальнейшее уменьшение величины шага, повышая трудоемкость не снижает погрешность приближения (время расчета $\approx 0.1s$). Граница погрешности кусочно-интерполяционного приближения не превышает порядка 10^{-17} на всем отрезке приближения (табл. 5.2) (время расчета $\approx 2.2\,s$).

Таким образом, применение кусочно-интерполяционного метода в силу его сравнительно высокой точности (на пять и более десятичных порядков) позволяет получить высокоточные значения для моделируемых концентраций реагентов автоколебательной реакции в произвольный момент времени. Как будет показано ниже, это позволит выявить новые уточненные экстремальные значения концентраций реагентов, и, как следствие, уточнить значения таких основных параметров автоколебательной реакции, как амплитуда и период колебаний. Кроме того, как отмечалось выше, с помощью предложенного метода обнаруживается качественная характеристика процесса изменения концентрации, которая не идентифицируется разностными методами численного моделирования.

Замечание 5.2. Специализированные решатели для жестких задач, представленные в системах компьютерной математики (*MATLAB*, *MathCAD*) и научной библиотеке (*SciPy*) при использовании их стандартных параметров не справляются с решением системы (5.3) при $\beta = 10$, $\alpha = 100$ на исследуемом интервале независимо от величины заданной границы погрешности.

В частности, при приближении решения данной задачи с помощью функции *ode15s* в среде *MATLAB* без фиксирования максимального значения величины шага интегрирования программа выдает ошибку «*Unable to meet integration tolerances without reducing the step size below the smallest value allowed (1.188180e-14) at time t*», то есть не справляется с задачей независимо от величины заданной границы погрешности. Другие решители жестких задач в среде *MATLAB*, и решатели, представленные в библиотеке *SciPy*, аналогично, не справляются с данной задачей и при фиксировании максимального значения величины шага интегрирования.

На основе применения функции *ode15s* (*MATLAB*) при фиксировании максимального значения величины шага интегрирования амплитуды колебаний значений переменных системы (5.3) вычисляются с границей абсолютной погрешности не превышающей $\approx 3.8 \times 10^{-6}$. Метод *Dormand-Prince_8* позволят

снизить границу погрешности вычисления амплитуд до $\approx 2.0\times10^{-7}$. Применение кусочно-интерполяционного метода даёт значения амплитуд колебаний реагентов химической реакции с границей абсолютной погрешности не превышающей 1.0×10^{-15}:

$$A_{y_1} = 2.11400123576868097,\ A_{y_2} = 173.394613181876717000.$$

На основе сравнительно точных кусочно-интерполяционных приближений экстремальных концентраций реагентов получено следующее значение периода изменения концентраций реагентов в системе (5.3): $T = 2.35895416.$

Таким образом, получают уточнения следующие важные параметры автоколебательной химической реакции: амплитуда и период колебаний значения концентраций реагентов, экстремальные значения концентраций.

Использование кусочно-интерполяционного метода позволило уточнить физико-химические параметры релаксационных автоколебаний в системе гликолиза на основе исследования математической модели, предложенной Дж. Хиггинсом, и как следствие сделать модель более адекватной моделируемому процессу.

В то же время кусочно-интерполяционный метод не является специально созданным для жестких задач и сохраняет инвариантность относительно вида задачи Коши для системы ОДУ (см. табл. 3.3, 3.4).

5.1.3. Обработка данных колебательной реакции окисления молекулярного водорода на модели Чумакова-Слинько. Одним из ярких примером гетерогенной колебательной химической реакций является реакция окисления молекулярного водорода на поверхности никелевого или платинового катализатора [183, 185]. Простейшая схема данной реакции имеет следующий вид [185]:

$$
\begin{aligned}
&(I) \quad H_2 + 2M \underset{k_{-1}}{\overset{k_1}{\rightleftarrows}} 2HM,\\
&(II) \quad O_2 + 2M \underset{k_{-2}}{\overset{k_2}{\rightleftarrows}} 2OM,\\
&(III) \quad 2HM + OM \xrightarrow{k_3} 3M + H_2O,\\
&(IV) \quad H_2 + OM \xrightarrow{k_4} M + H_2O,\\
&(V) \quad M_n + OM \underset{k_{-5}}{\overset{k_5}{\rightleftarrows}} (M_nO) + M.
\end{aligned}
\tag{5.4}
$$

Схема (5.4) может быть качественно описана следующим образом. «Первые две стадии обратимые и представляют собой адсорбцию молекул водорода и кислорода, стадия (III) основная необратимая стадия – окисление водорода по рекомбинационному механизму Лэнгмюра-Хиншельвуда, четвертая необратимая стадия – окисление водорода по ударному механизму Или-Ридила. Пятая обратимая стадия – растворение кислорода в приповерхностном слое катализатора. В приповерхностном слое происходит накопление излишка адсорбированного кислорода, иначе идет обмен поверхностными и приповерхностными центрами адсорбции через окисление. Данной реакции на основании закона действующих масс соответствует следующая математическая модель» [186]:

$$
\left.
\begin{aligned}
y_1' &= k_1(1 - y_1 - y_2)^2 - k_{-1}y_1^2 - 2k_3y_1^2y_2,\\
y_2' &= k_2(1 - y_1 - y_2)^2 - k_{-2}y_2^2 - k_3y_1^2y_2 - k_4y_2,\\
y_3' &= k_5\, y_2(1 - y_3) - k_{-5}\, y_3(1 - y_1 - y_2),
\end{aligned}
\right\}
\tag{5.5}
$$

где $y_1 = [HM]$, $y_2 = [OM]$, $y_3 = [M_nO]$, k_i, k_{-i} – константы скоростей реакций, $(1 - y_1 - y_2)$, $(1 - y_3)$ – степени покрытия поверхности M и подложки M_n катализатора свободными центрами адсорбции соответственно.

Для возникновения автоколебаний требуется наличие в системе обратной связи достаточной силы. Как отмечено в [186] «обратная связь осуществляется через воздействие реагирующих веществ на активность и свойства катализатора и выражается в зависимости энергий активации стадий адсорбции и реакции от

концентрации водорода и кислорода на поверхности катализатора». В простейшем случае константы скоростей стадий (III) и (IV) схемы (5.4) экспоненциально зависят от концентраций поверхностных компонентов [183]:

$$k_3 = k_{30}\, e^{-\mu_3 y_2},\ k_4 = k_{40}\, e^{-\mu_4 y_2 - \mu_5 y_3}.$$

Тогда система (5.5) примет вид

$$\left.\begin{aligned} y_1' &= k_1(1-y_1-y_2)^2 - k_{-1}y_1^2 - 2k_{30}\, e^{-\mu_3 y_2} y_1^2 y_2, \\ y_2' &= k_2(1-y_1-y_2)^2 - k_{-2}y_2^2 - k_{30}\, e^{-\mu_3 y_2} y_1^2 y_2 - k_{40}\, e^{-\mu_4 y_2 - \mu_5 y_3} y_2, \\ y_3' &= \varepsilon\,(y_2(1-y_3) - \alpha\, y_3(1-y_1-y_2)), \end{aligned}\right\} \qquad (5.6)$$

где $\varepsilon = k_5$ – малый параметр, $\alpha = k_{-5}/k_5$. Систему (5.6) принятно называть системой Чумакова-Слинько. В работе [187] для системы (5.6) на основе качественного и численного исследования выделена широкая область значений параметров, при которых реакция (5.4) является автоколебательной (все адсорбированные компоненты HM, OM и (M_nO) реакции испытывают незатухающие колебания с постоянным периодом). Сохранение автоколебательного режима реакции позволяет решать проблему аккомодации тепловой энергии, которая выделяется в рекомбинационных стадиях реакции. Значения параметров модели, при которых химическая реакция (5.4) имеет автоколебательный характер [187]:

$$\begin{aligned} &k_1 = 1.5,\ k_{-1} = 0.008,\ k_2 = 20,\ k_{-2} = 0.02,\ k_{30} = 100, \\ &k_{40} = 20,\ \varepsilon = 0.0024,\ \alpha = 7.88,\ \mu_3 = 2,\ \mu_4 = 7, \end{aligned} \qquad (5.7)$$

значение параметра μ_5 изменяется в диапазоне $-28 \le \mu_5 \le -12.320463$. Все параметры и условия считаются безразмерными. В [187] показано, что при уменьшении параметра μ_5, что соответствует изменению константы скорости основной стадии (IV) схемы (5.4), в системе (5.6) реализуется бифуркация Андронова-Хопфа рождения устойчивой периодической орбиты из состояния равновесия. Значение параметра μ_5, при котором возникает бифуркация,

определено: $\mu_5 = -12.320463$. Нижняя граница значения параметра μ_5 в работе не представлена, при этом показано, что и при уменьшии значения параметра до $\mu_5 = -28$ реакция еще носит автоколебательный характер. Численное решение системы (5.6) выполнялось в среде *MathCAD* на основе ЯМРК 4-го порядка.

Ниже с применением кусочно-интерполяционного метода выполнено уточнение границ изменения параметра μ_5, соответствующих автоколебательному режиму реакции, кроме того уточнены значения периода и амплитуды колебаний значений концентрации реагентов.

Кусочно-интерполяционное приближение решения системы (5.6) со значениями параметров (5.7) строилось на интервале $t \in [0, 1000]$ с начальными данными $y_1(0) = 0.1$, $y_2(0) = 0.5$, $y_3(0) = 0$ при постепенном увеличении значения параметра μ_5 от верхней границы $\mu_5 = -12.320463$, представленной в работе [187]. На основе полученных результатов численного моделирования установлен автоколебательный характер значений концентраций реагентов реакции при $-12.320463 \le \mu_5 \le -12.2732865\,366359$. При значении параметра $\mu_5 = -12.2732865\,366359$ в системе (5.6) реализуется бифуркация Андронова-Хопфа. При значениях $\mu_5 \ge -12.2732865\,36636$ система переходит в состояние равновесия. На рис. 5.7 представлена визуализация автоколебаний компонентов системы (5.6) с параметрами (5.7) при значении $\mu_5 = -12.2732865\,366359$, полученных на основе кусочно-интерполяционного приближения на интервале $t \in [0, 1000]$ при $y_1(0) = 0.1$, $y_2(0) = 0.5$, $y_3(0) = 0$.

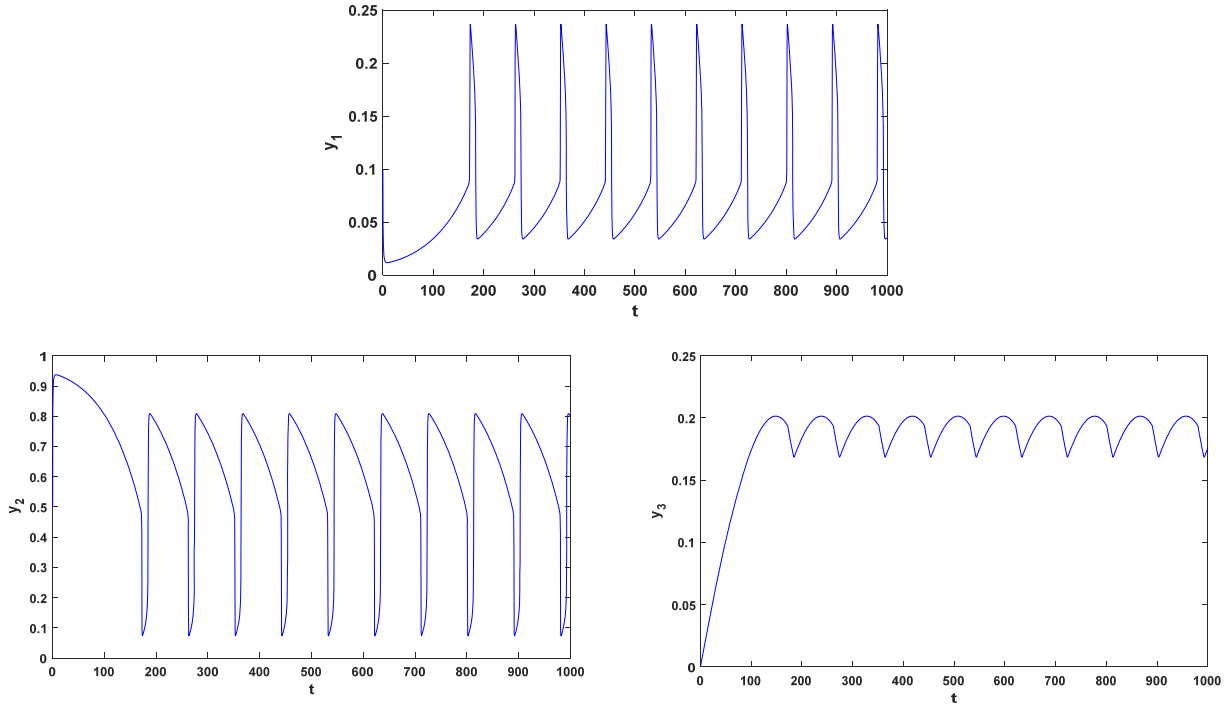

Рис. 5.7. Визуализация автоколебаний значений степени покрытия реагентами HM , OM и (M_nO) поверхности катализатора в модели (5.6), (5.7) при $\mu_5 = -12.2732865\,366359$, полученных на основе кусочно-интерполяционного приближения решения на интервале $t \in [0, 1000]$

На рисунке продемонстрированы релаксационные автоколебания адсорбированных компонентов HM, OM и квазисинусоидальный тип колебаний компонента (M_nO). Динамика адсорбированных компонентов аналогична при изменении $-12.320463 \le \mu_5 \le -12.2732865\,366359$, изменяется лишь период колебаний. Конфигурация и устойчивость предельного цикла показаны на рис. 5.8.

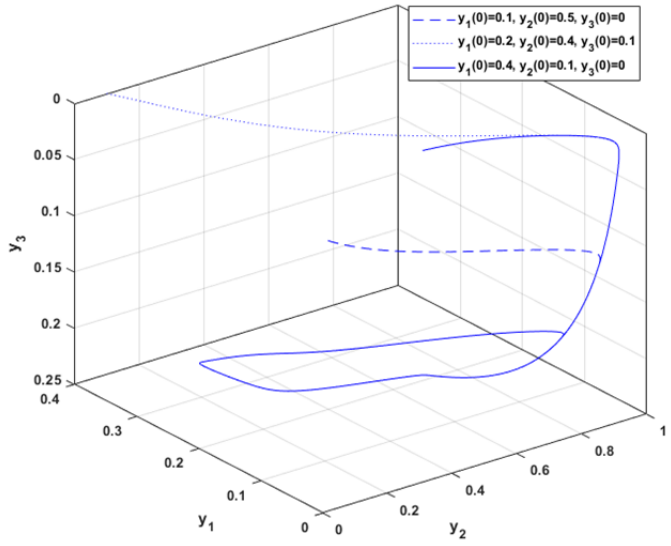

Рис. 5.8. Предельный цикл в системе Чумакова–Слинько (5.6) при значениях параметров (5.7)и $\mu_5 = -12.2732865366359$, полученный на основе кусочно-интерполяционного приближения

Конфигурация цикла аналогична предельному циклу системы, представленному в [187] для значения $\mu_5 = -12.320463$.

Таким образом, на основе кусочно-интерполяционного приближения решения дифференциальной модели уточнена верхняя граница параметра модели, при которой сохраняется автоколебательный процесс. Тем самым, уточнено соответствующее изменение константы скорости основной стадии (IV) схемы (5.4) реакции окисления молекулярного водорода на поверхности никелевого или платинового катализатора. Данные уточнения позволяют повысить качество решения проблемы аккомодации тепловой энергии, выделяемой в рекомбинационных стадиях.

Следует отметить, что решение задачи (5.6), (5.7) со значением параметра $\mu_5 = -12.2732865366359$ при использовании решателя *ode113*, реализующего в *MATLAB* многошаговый метод Адамса-Башворта-Мултона переменного порядка (от 1 до 13), приводит к неверному результату моделирования (вследствие высокой границы абсолютной погрешности приближения 2.2×10^{-1} для данной задачи). На рис. 5.9 представлена (ошибочная) зависимость от времени степени покрытия реагентом *HM* поверхности катализатора в модели

(5.6), (5.7) при $\mu_5 = -12.2732865\,366359$, полученная с использованием функции *ode113* (*MATLAB*).

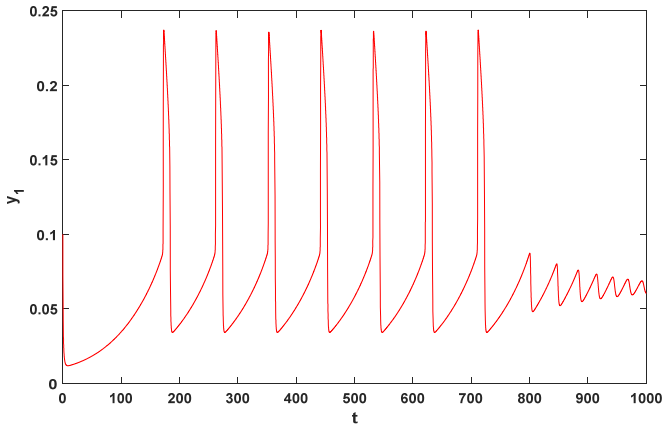

Рис. 5.9. Зависимость от времени степени покрытия реагентом *HM* поверхности катализатора в модели (5.6), (5.7) при $\mu_5 = -12.2732865\,366359$, полученная с использованием функции *ode113* (*MATLAB*)

Таким образом, использование кусочно-интерполяционного метода численного моделирования позволяет выявить качество химического процесса, которое нельзя выявить с помощью разностных методов.

В работе [186] с помощью «программного комплекса для численного исследования динамики колебательных химических реакций на основе полунеявных методов» вычислены период и амплитуды колебаний значений концентрации реагентов реакции (5.4): $T \approx 76\,s$, $A_{y_1} = 0.24$, $A_{y_2} = 0.74$, $A_{y_3} = 0.032$. Исследование выполнялось на основе численного интегрирования дифференциальной модели (5.6), (5.7) с начальными данными $y_1(0) = 0.1$, $y_2(0) = 0.5$, $y_3(0) = 0$ при $\mu_5 = -12.320463$ на интервале $t \in [0,\ 500]$.

Сравнительно высокая точность (граница абсолютной погрешности 2.3×10^{-18}) и непрерывность кусочно-интерполяционного приближения решения данной задачи позволили существенно уточнить экстремальные значения концентраций реагентов автоколебательной реакции и, как следствие, уточнить амплитуды колебаний концентраций реагентов автоколебательной реакции, представленные в [186]. А именно, получены следующие значения амплитуд:

$$A_{y_1} = 0.219292779881,\ A_{y_2} = 0.746920652997,\ A_{y_3} = 0.0328259406092.$$

Таким образом, амплитуда колебаний концентрации адсорбированного компонента HM уточнена на $\approx 8.6\,\%$, компонента OM – на $\approx 0.94\,\%$, компонента (M_nO) – на $\approx 2.6\,\%$. Далее, на основе кусочно-интерполяционного метода получено следующее уточненное значение периода изменения концентраций реагентов в системе (5.6): $T = 76.29589$. Данное значение согласуется и уточняет значение периода, рассчитанное в [186] на основе полунеявных численных методов на $\approx 0.39\,\%$.

5.2. Обработка данных в моделях колебательных динамических процессов. Как отмечалось в главе 1, сходные проблемы численного моделирования имеют колебательные системы и в других областях. Ниже на основе кусочно-интерполяционного метода выполняется уточнение численного моделирования колебательных динамических процессов, возникающих в задачах радиоэлектроники, баллистико-навигационного обеспечения полетов космических аппаратов, рассчета динамики возмущенного движения искусственных спутников Земли. Вместе с тем, на основе кусочно-интерполяционного приближения решений дифференциальных моделей указанных процессов попутно достигается и другая цель, которую не удается достигнуть при применении известных разностных методов: повышение качества моделирования вследствие непрерывности полученных уточненных приближений и их производных на всем промежутке интегрирования.

5.2.1 Обработка числовых данных модели электрического равновесия автогенератора с внутренней обратной связью. Одной из основных моделей для анализа периодических автоколебаний в задачах радиоэлектроники является модель электрического равновесия автогенератора с внутренней обратной связью, которая при использовании кубической

аппроксимации вольт-амперной характеристики туннельного диода может быть представлена в виде нелинейной системы [52, 188]:

$$u' = \frac{G(u - ku^3/3)}{C} - \frac{i}{C},\ i' = \frac{u}{L}, \tag{5.8}$$

где u – напряжение на контуре, i – ток индуктивного элемента, G – модуль дифференциальной проводимости туннельного диода в рабочей точке, C – емкость, L – индуктивность.

Численное моделирование на основе кусочно-интерполяционного решения системы (5.8) выполняется на интервале $t \in [0, 100]$ при значениях параметров и начальных условий:

$$L = 1,\ C = 1,\ G = 4,\ k = 1,\ u_0 = 1,\ i_0 = 4, \tag{5.9}$$

соответствующих релаксационному характеру автоколебаний [189]. Визуализация результатов моделирования, согласующаяся с физическим обоснованием характера колебаний, представлена на рис. 5.10.

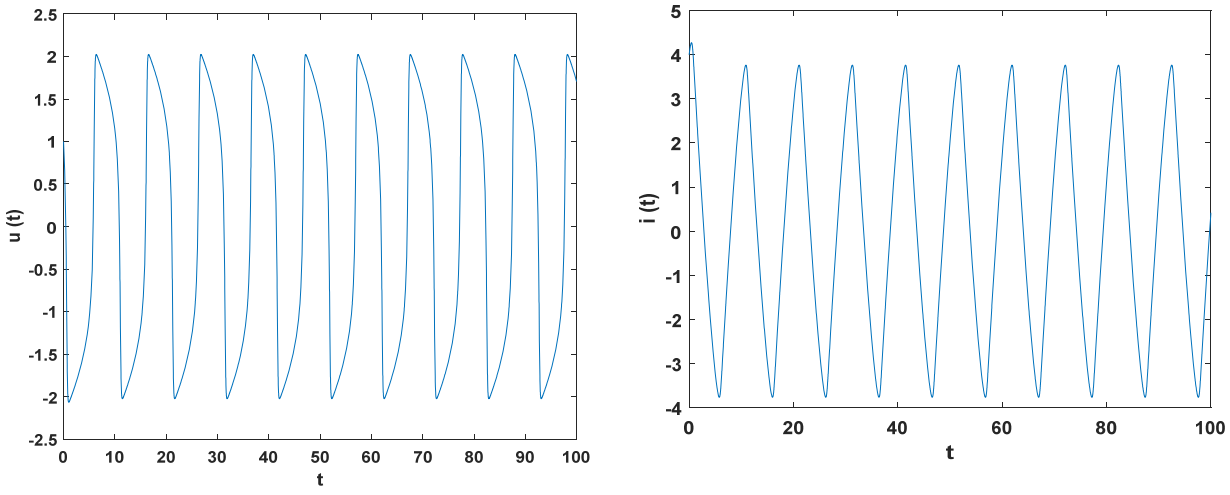

Рис. 5.10. Визуализация релаксационных автоколебаний значений компонентов системы (5.8) на основе кусочно-интерполяционного приближения на интервале $t \in [0, 100]$ при $L = 1,\ C = 1,\ G = 4,\ u_0 = 1,\ i_0 = 4$

Кусочно-интерполяционный метод дает приближение решения с точностью порядка 10^{-18} на всем интервале приближения. Вместе с тем, приближение решения данной задачи многошаговым методом Адамса-Башворта-Мултона переменного порядка (функция *ode113* в *MATLAB*)

характеризуется границей погрешности порядка 10^{-11}. Сравнительно высокая точность и непрерывность кусочно-интерполяционного решения позволяют уточнить динамику изменения значений напряжения на контуре автогенератора в окрестностях точек локальных максимумов (рис. 5.11).

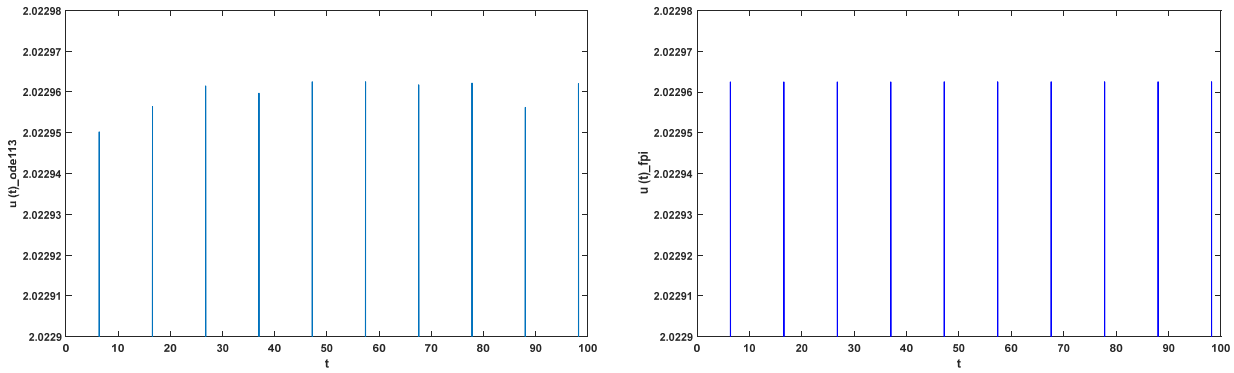

Рис. 5.11. Сравнение динамики изменения значений напряжения на контуре автогенератора на основе кусочно-интерполяционного решения задачи (5.8), (5.9) ($u(t)\_fpi$) и разностного решения по методу Адамса-Башворта-Мултона ($u(t)\_ode113$)

На рис. 5.11 справа представлено кусочно-интерполяционное приближение динамики изменения значений напряжения на контуре автогенератора, слева – приближение по методу Адамса-Башворта-Мултона. Из сравнения графиков приближения видно, что с помощью кусочно-интерполяционного метода удалось точнее показать автоколебательный характер изменения значений напряжения (гармоники идентичны) в отличие от известного метода Адамса-Башворта-Мултона.

Отмеченные качества кусочно-интерполяционного приближения позволяют также уточнить основные параметры колебаний на выходе автогенератора. В табл. 5.3 приведены значения амплитуды и периода релаксационных колебаний значений напряжения на контуре автогенератора, полученные на основе кусочно-интерполяционного и разностных приближений решения задачи (5.8), (5.9).

Таблица 5.3

Основные параметры релаксационных автоколебаний системы (5.8), (5.9)

	$\mathrm{A}_{u(t)}$	$\Delta\mathrm{A}_{u(t)}$	T	ΔT
(1)	4.0459	$\approx 1.7\times10^{-5}$	10.20	$\approx 4.9\times10^{-3}$
(2)	4.045924	$\approx 4.7\times10^{-7}$	10.203	$\approx 6.2\times10^{-4}$
(3)	4.0459250019	$\approx 3.2\times10^{-11}$	10.2035	$\approx 1\times10^{-5}$
(4)	4.04592500193762369	$\approx 1.0\times10^{-18}$	10.203523691	0

В строке (1) представлены значения, полученные с помощью метода Адамса-Башворта-Мултона (*ode113* (*MATLAB*)), в строке (2) – метода Дормана-Принса 5-го порядка (*ode45*(*MATLAB*)), в строке (3) – метода Дормана-Принса 8-го порядка аппроксимации, в строке (4) представлены значения, полученные на основе кусочно-интерполяционного метода. В столбцах $\Delta\mathrm{A}_{u(t)}$ и ΔT даны значения абсолютной погрешности приближения амплитуды и периода колебаний соответственно. Из таблицы видно, что среди разностных методов наименьшим значением погрешности приближения амплитуды (порядка 10^{-11}) характеризуется метод Дормана-Принса 8-го порядка. При этом погрешность вычисления амплитуды на основе кусочно-интерполяционного метода характеризуется значением порядка 10^{-18}. Значение периода колебаний на основе кусочно-интерполяционного метода вычислено с девятью десятичными знаками после запятой с «нулевой» погрешностью в формате вывода данных. Вместе с тем приближение значения периода колебаний на основе разностного метода характеризуется границей погрешности $\approx 1\times10^{-5}$.

Таким образом, на основе кусочно-интерполяционного приближения дифференциальной модели уточнены значения напряжения на контуре и тока индуктивного элемента автогенератора с внутренней обратной связью, уточнена динамика их изменения, значения амплитуды и периода релаксационных автоколебаний.

5.2.2. Обработка данных модели управляемого орбитального движения космического аппарата. С применением кусочно-интерполяционного метода выполняется уточнение численного моделирования движения КА с управлением при помощи «малой тяги», необходимое, в частности, для прогнозирования движения КА, повышения качества моделирования траекторных измерений при осуществлении маневров, повышения точности значений параметров орбиты устойчивого движения КА. При этом уточнение достигается без повышения времени численного моделирования в сравнении с разностными методами, включая методы с адаптивным изменением величины шага интегрирования, что актуально для баллистико-навигационного обеспечения полетов КА в режиме реального времени.

Математическая модель движения КА с управлением при помощи «малой тяги» в плоскости орбиты может быть представлена в виде [50, 51]:

$$\dot{r} = V_r, \quad \dot{V}_r = \frac{V_\theta^2}{r} - \frac{h^2}{r^2 p} + U_r, \; \dot{\theta} = \frac{V_\theta}{r}, \; \dot{V}_\theta = -\frac{V_r V_\theta}{r} + U_\theta, \tag{5.10}$$

где r, θ – полярные координаты, V_r, V_θ – радиальная и трансверсальная составляющие скорости; U_r, U_θ – составляющие вектора тяги, p – фокальный параметр. Законы управления U_r и U_θ, обеспечивающие устойчивое орбитальное движение КА, имеют вид [52]:

$$U_r = \frac{h^2}{p r^2} - \frac{(\omega_2 + h)^2}{p r^2} - \frac{\Phi}{p}\left(r\,\dot{\omega}_1(t) + e\,\omega_2 \sin\theta\right), \; U_\theta = -\frac{\omega_2}{r}\,\Phi, \tag{5.11}$$

где $\omega_1 = r(1 + e\cos\theta) - p = 0$, $\omega_2 = r^2\,\dot{\theta}(t) - h = 0$ – законы Кеплера, $\Phi = \frac{h}{r^2}$. Более подробно модель описывается в главе 1 (п. 1.1.3).

Для управления движением КА необходимо в режиме реального времени находить приближенное решение системы (5.10), (5.11) с высокой точностью. Численное моделирование решения системы (5.10), (5.11) выполняется с

помощью кусочно-интерполяционного метода на промежутке $[0, 2\times10^6]$ при начальных условиях

$$r(0)=\frac{p}{1+e}\ км,\ V_r(0)=\frac{eh}{p}\ км/с,\ \theta(0)=0,\ V_\theta(0)=0, \qquad (5.12)$$

где $p=36000\ км$, $e=0.5$, $T=24\,ч$, $h=\dfrac{2\pi p^2}{T(1-e^2)^{3/2}}$. На рис. 5.12 представлена динамика изменения значений декартовых координат положения космического аппарата с управлением (5.11) в плоскости орбиты, полученная на основе кусочно-интерполяционного решения задачи (5.10) – (5.12)

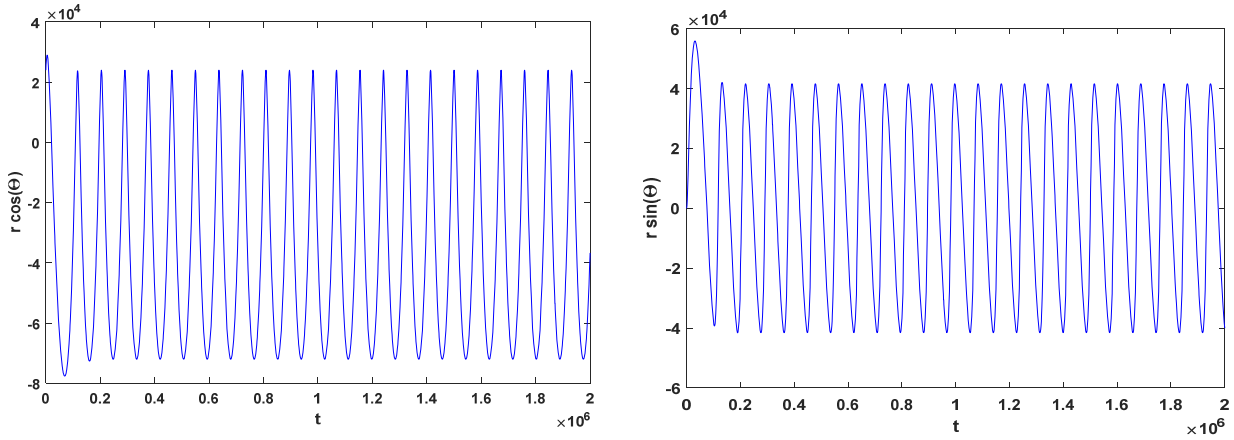

Рис. 5.12. Изменение декартовых координат положения космического аппарата с управлением (5.11) в плоскости орбиты, полученная на основе кусочно-интерполяционного решения задачи (5.10) – (5.12)

Следует отметить, что полученная траектория движения является непрерывной и вычислена с точностью не превышающей $\approx1\times10^{-14}$ (параметры метода: $\beta_r-\alpha_r=1000$, $\ell=10$, $n=11,\ k=2$). Время приближенного решения на персональном компьютере менее одной секунды, $531\,мс$. Вместе с тем, при применении разностных методов наименьшей границы абсолютной погрешности приближения решения данной задачи $\approx1\times10^{-9}$ удается достигнуть с использованием метода Дормана-Принса 8-го порядка аппроксимации ($h=0.1$, время решения $44\,с\ 810\,мс$). Использование других разностных методов, в том числе представленных в среде *MATLAB* и библиотеке *SciPy*, не снижает погрешность приближения. Кроме того, полученные разностные

значения являются дискретными, что отражается на качестве численного моделирования (рис. 5.13).

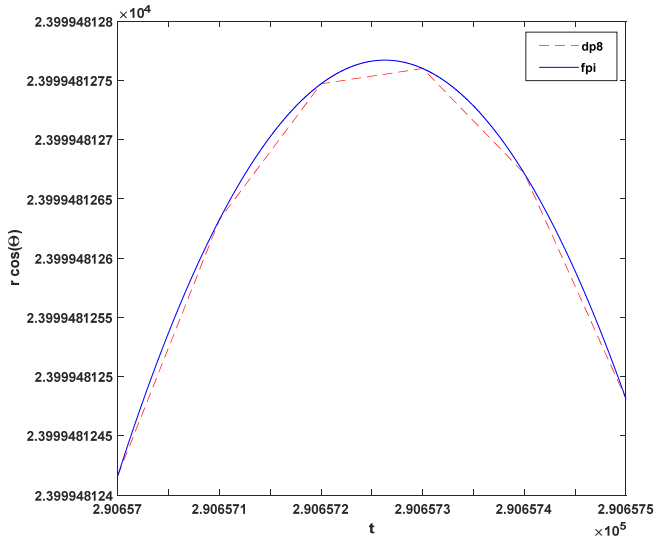

Рис. 5.13. График изменения $r(t)\cos(\theta(t))$, задача (5.10) – (5.12), $dp8$ – метод Дормана-Принса 8-го порядка ($h=10^{-1}$), fpi – кусочно-интерполяционный метод

На рис. 5.13. представлены графики изменения $r(t)\cos(\theta(t))$ в окрестности одной из точек локального максимума на основе метода Дормана-Принса 8-го порядка и кусочно-интерполяционного метода. Из рисунка видно, что кусочно-интерполяционное приближение показывает гладкость процесса изменения координаты положения КА, тем самым повышается качество численного моделирования в сравнении с дискретным приближением на основе известного метода Дормана-Принса 8-го порядка.

Таким образом, на основе кусочно-интерполяционного приближения дифференциальной модели получено уточненное непрерывное приближение траектории полета космического аппарата с управлением при помощи «малой тяги» (граница абсолютной погрешности $\approx 1\times10^{-14}$).

Замечение 5.2. Следует отметить, что уменьшение величины шага разностного метода с целью повышения качества визуализации результатов моделирования приводит к снижению точности приближения и значительному возрастанию времени расчета. При этом качество визуализации повышается незначительно, сохраняется дискретный характер приближения.

В частности, приближение решения задачи (5.10) – (5.12) методом Дормана-Принса 8-го порядка при $h=10^{-2}$ характеризуется границей абсолютной погрешности $\approx 9\times10^{-9}$ (при $h=10^{-1}$ граница погрешности $\approx 1\times10^{-9}$), время расчета – $7\,мин\;35\,с\;775\,мс$ (при $h=10^{-1}$ – $44\,с\;810\,мс$). Вместе с тем граница погрешности непрерывного кусочно-итерполяционного приближения решения той же задачи не превышает порядка $\approx 1\times10^{-14}$ на всем интервале интегрирования при времени расчета $531\,мс$.

На основе полученных уточненных значений положения КА установлено время, необходимое для вывода космического аппарата на устойчивую периодическую орбиту при управлении (5.11), соответствующему внешнему воздействию гравитационных сил. Устойчивое движение по эллиптической орбите с точным значением периода $T=86400\,c$ возникает при $t\geq 1.123\times10^{6}$ (рис. 5.14 (справа)).

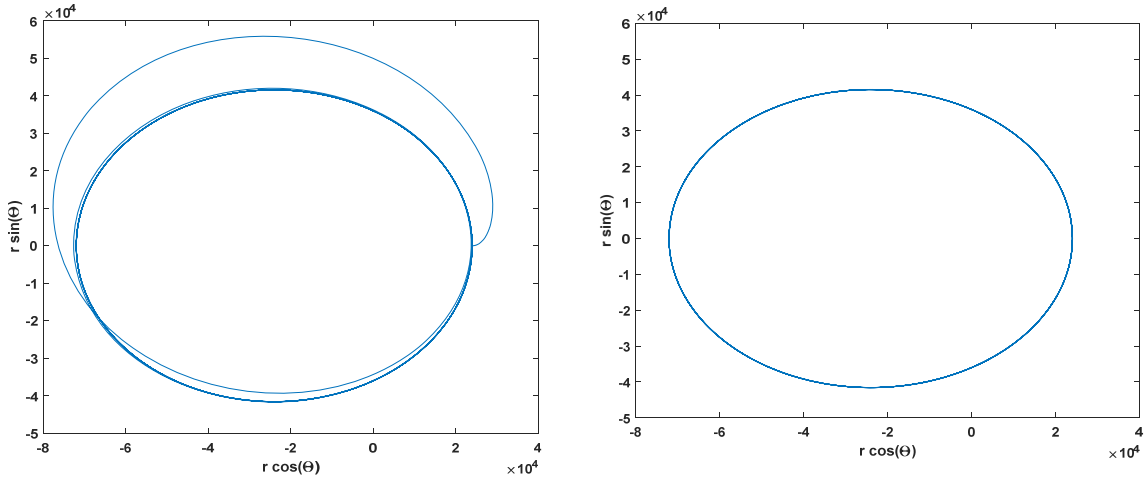

Рис. 5.14. Траектория движения космического аппарата с управлением (5.11) в плоскости орбиты, полученная на основе кусочно-интерполяционного решения задачи (5.10) – (5.12) при $t\in[0,\,2\times10^{6}]$ (слева), при $t\in[1.123\times10^{6},\,2\times10^{6}]$ (справа).

Далее, с применением кусочно-интерполяционного метода рассчитаны значения параметров орбиты устойчивого движения космического аппарата с границей абсолютной погрешности $\approx 1\times10^{-13}$:

$$a=96000\;км,\;b=83138{,}4387\;633061100\;км,$$

a, b – большая и малая полуоси эллипса. В соответствии с данными значениями фокальный параметр $p = 72000\ км$, эксцентриситет $e = 0.5$.

Таким образом, на основе кусочно-интерполяционного приближения дифференциальной модели (5.10) – (5.12) получено непрерывное приближение траектории полета космического аппарата с управлением при помощи «малой тяги» с точностью $\approx 1\times 10^{-14}$, рассчитаны параметры эллиптической орбиты устойчивого движения КА с точностью $\approx 1\times 10^{-13}$. В сравнении с разностными методами высокого порядка получено уточнение траектории движения и параметров орбиты на 5 десятичных порядков. При этом время кусочно-интерполяционного приближения решения дифференциальной модели на персональном компьютере в сравнении с временем разностного приближения меньше на 8 десятичных порядков, что особенно важно для управления движением КА в режиме реального времени. Ясно, что сокращение времени расчета устойчивой орбиты КА влечет сокращение расхода топлива на коррекцию орбиты, а получение точного значения параметров снижает риск ошибки вывода на орбиту.

Непрерывное влияние на эволюцию орбиты космического аппарата оказывают возмущающие гравитационные силы. Ниже выполняется уточнение численного моделирования возмущенного движения искусственного спутника Земли (ИСЗ), выведенного на низкую околоземную орбиту. Актуальность выполненного уточнения обусловлена, в частности, повышением точности наблюдательной и измерительной техники. Вместе с тем, на основе кусочно-интерполяционного приближения попутно достигается и повышение качества моделирования вследствие непрерывности полученных приближений решения уравнений движения, что необходимо для получения координат ИСЗ в произвольно заданные моменты времени, например, при измерениях с помощью лазерного дальномера от пункта наблюдения до ИСЗ [15].

5.2.3. Кусочно-интерполяционная обработка данных при моделировании возмущенного движения космического аппарата. С использованием кусочно-интерполяционного метода выполняется уточнение численного моделирования движения итальянского геодезического спутника LARES, выведенного на низкую околоземную орбиту 13.02.2012 г. Спутник используется для исследований в области геодинамики и решения задач космической геодезии; несет на своей поверхности 92 уголковых отражателя (кубических ретрорефлектора) для высокоточного измерения расстояний с лазерных локационных станций, расположенных на поверхности Земли.

Наибольшее влияние среди возмущающих факторов на движение ИСЗ, расположенного на низкой околоземной орбите, оказывают вторая, третья и четвертая зональные гармоники гравитационного поля Земли [190, 191]. Дифференциальные уравнения возмущенного движения ИСЗ с учётом данных зональных гармоник геопотенциала имеют вид [191, 192]:

$$\ddot{\mathbf{x}} = -\mu\frac{\mathbf{x}}{r^3} + \delta\ddot{\mathbf{x}}_{J_2} + \delta\ddot{\mathbf{x}}_{J_3} + \delta\ddot{\mathbf{x}}_{J_4}, \tag{5.13}$$

$$\delta\ddot{\mathbf{x}}_{J_2} = \frac{3}{2}J_2\mu\frac{a_e^2}{r^5}\begin{pmatrix}\dfrac{5x_1x_3^2}{r^2} - x_1\\ \dfrac{5x_2x_3^2}{r^2} - x_2\\ \dfrac{5x_3^3}{r^2} - 3x_3\end{pmatrix},\ \delta\ddot{\mathbf{x}}_{J_3} = \frac{5}{2}J_3\mu\frac{a_e^3}{r^7}\begin{pmatrix}\dfrac{7x_1x_3^3}{r^2} - 3x_1x_3\\ \dfrac{7x_2x_3^3}{r^2} - 3x_2x_3\\ \dfrac{7x_3^4}{r^2} - 6x_3^2 + \dfrac{3}{5}r^2\end{pmatrix},$$

$$\delta\ddot{\mathbf{x}}_{J_4} = \frac{1}{4}J_4\mu\frac{a_e^4}{r^7}\begin{pmatrix}\dfrac{315x_1x_3^4}{2r^4} - \dfrac{105x_1x_3^2}{r^2} + \dfrac{15x_1}{2}\\ \dfrac{315x_2x_3^4}{2r^4} - \dfrac{105x_2x_3^2}{r^2} + \dfrac{15x_2}{2}\\ \dfrac{315x_3^5}{2r^4} - \dfrac{175x_3^3}{r^2} + \dfrac{75x_3}{2}\end{pmatrix},$$

где $\mu = 3.986004418\times10^{14}\,м^3/с^2$ – геоцентрическая гравитационная постоянная (с учетом атмосферы), $\mathbf{x} = (x_1,\ x_2,\ x_3)$ – вектор положения ИСЗ в геоцентрической равноденственной прямоугольной системе координат, $a_e = 6378136\,м$ –

большая полуось общеземного эллипсоида; $\delta\ddot{\mathbf{x}}_{J_2}$, $\delta\ddot{\mathbf{x}}_{J_3}$ и $\delta\ddot{\mathbf{x}}_{J_4}$ – возмущающие ускорения от второй, третьей и четвертой зональных гармоник, $J_2 = 1082.62575\times10^{-6}$, $J_3 = -2.532435\times10^{-6}$, $J_4 = -2.37089\times10^{-6}$ – соответственные зональные гармонические коэффициенты.

Параметры орбиты спутника LARES на 10:44:02.839776 UTC 2021-05-01 приведены в табл. 5.4 [193].

Таблица 5.4

Начальные орбитальные данные ИСЗ LARES

Элемент орбиты	Числовое значение
a	7822074,833 м
e	0.0008463
i	69.4919°
Ω	251.0444°
ω	352.3935°
M_0	7.6961°
n	12.54930494422217витков/солнечные сутки
t_0	$10^h\,44^m\,2.839776^s$ 01.05.2021

В табл. 5.4 использованы традиционные обозначения элементов оскулирующей орбиты: a – большая полуось оскулирующего эллипса, e – эксцентриситет эллиптической орбиты, i – наклонение плоскости оскулирующей орбиты к плоскости экватора, Ω – долгота восходящего узла орбиты, ω – аргумент перицентра, M_0 – средняя аномалия, n – частота обращения, t_0 – время эпохи. С использованием модели, представленной в [194], в соответствии с данными табл. 5.4, вычислены начальные значения скорости и геоцентрических координат ИСЗ LARES:

$$\begin{aligned} &x_1 = \text{-}2609529.0\,721402253, \quad x_2 = \text{-}7369242.8\,79058394,\\ &x_3 = \ 106.348939\,15203872, \quad \dot{x}_1 = 2360.40028\,69474456,\\ &\dot{x}_2 = \text{-}829.84464\,03508121, \quad \dot{x}_3 = 6692.75677\,93414635. \end{aligned} \tag{5.14}$$

Кусочно-интерполяционное приближение решения задачи (5.13), (5.14) строится на интервале длиной $t \in [t_0, t_0 + \Delta t]$, $\Delta t = 86400\ c$, соответствующем

≈ 12.5 виткам спутника. Визуализация полученной на основе кусочно-интерполяционного приближения траектории движения спутника в геоцентрической равноденственной прямоугольной системе координат представлена на рис. 5.15.

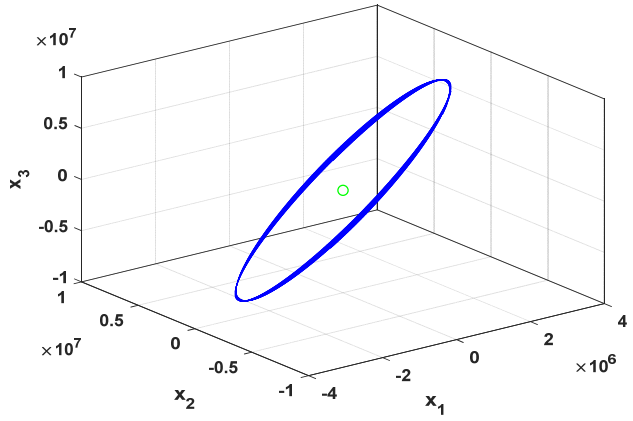

Рис. 5.15. Траектория движения ИСЗ LARES в геоцентрической равноденственной прямоугольной системе координат, полученная на основе кусочно-интерполяционного решения задачи (5.13) – (5.14).

Точность приближения, согласно методике [37, 195], рассчитывалась путем обратного интегрирования задачи (5.13), (5.14) на интервале $t \in [t_0 + \Delta t,\ t_0]$ с отрицательным значением шага интегрирования (для кусочно-интерполяционного метода с отрицательным значением длины интервала разбиения $\beta_r - \alpha_r$) и начальными данными, полученными при решении прямой задачи в конечной точке $t = t_0 + \Delta t$. По данной методике для кусочно-интерполяционного приближения получены следующие значения абсолютной погрешности приближения компонентов системы:

$$\Delta x_1 \approx 1.160 \times 10^{-11},\quad \Delta x_2 \approx 1.410 \times 10^{-11},\quad \Delta x_3 \approx 2.657 \times 10^{-11},$$
$$\Delta \dot{x}_1 \approx 2.176 \times 10^{-14},\quad \Delta \dot{x}_2 \approx 4.563 \times 10^{-14},\quad \Delta \dot{x}_3 \approx 9.770 \times 10^{-15}.$$

Точность кусочно-интерполяционного приближения задачи (5.13), (5.14) практически совпала с точностью представления начальных данных (5.14), при этом время решения составило 984 *мс*.

С другой стороны, разностное приближение решения этой же задачи с помощью метода Адамса-Башворта-Мултона (*ode113* (*MATLAB*)) характеризуется следующими значениями абсолютной погрешности

$$\Delta x_1 \approx 4.416\times10^{-6},\quad \Delta x_2 \approx 1.556\times10^{-6},\quad \Delta x_3 \approx 1.259\times10^{-5},$$
$$\Delta \dot{x}_1 \approx 4.108\times10^{-9},\quad \Delta \dot{x}_2 \approx 1.169\times10^{-8},\quad \Delta \dot{x}_3 \approx 2.637\times10^{-11}$$

при времени решения 111 *мс*.

Таким образом, применение кусочно-интерполяционного метода повысило точность приближения траектории движения искусственного спутника Земли на ≈ 5 десятичных порядков в сравнении с приближением, полученным на основе разностного метода Адамса-Башворта-Мултона, при незначительном повышении времени решения.

Вместе с тем, целесообразно отметить, что для уточненного численного моделирования в области небесной механики разработаны специализированные численные методы решения систем ОДУ. Как отмечалось в главе 1 (п. 1.2), одним из наиболее распространенных методов с повышенной точностью при решении астрономических задач является неявный одношаговый метод Эверхарта [5, 14]. В [16] представлена программная реализация метода, названная автором «интегратор Гаусса-Эверхарта», в которой выполнены некоторые видоизменения исходного алгоритма метода, позволяющие снизить временную сложность при сохранении высокой точности приближения. Применение интегратора Гаусса-Эверхарта 19-го порядка позволило получить приближение решения задачи (5.13) – (5.14) со следующими значениями абсолютной погрешности

$$\Delta x_1 \approx 2.853\times10^{-9},\quad \Delta x_2 \approx 1.141\times10^{-9},\quad \Delta x_3 \approx 8.204\times10^{-9},$$
$$\Delta \dot{x}_1 \approx 2.654\times10^{-12},\quad \Delta \dot{x}_2 \approx 7.536\times10^{-12},\quad \Delta \dot{x}_3 \approx 5.800\times10^{-14}$$

при времени решения 12 *с* 187 *мс*.

Таким образом, уточнение координат и скорости возмущенного движения спутника выполнено более чем на два десятичных порядка в сравнении с результатами моделирования на основе специализированного метода Гаусса-Эверхарта 19-го порядка. При этом время расчета кусочно-интерполяционного приближения на порядок меньше.

В итоге, можно заключить, что с помощью кусочно-интерполяционного метода выполнено уточнение численного моделирования возмущенного движения итальянского геодезического спутника LARES, выведенного на низкую околоземную орбиту. Сравнительно высокая точность и непрерывность достигнутого приближения позволяют получить достаточно точные координаты ИСЗ в произвольно заданные моменты времени, что, в частности, необходимо при измерениях с помощью лазерного дальномера от пункта наблюдения до ИСЗ.

Кроме того, можно отметить, что более ранний и более точный расчет возмущения орбиты позволяет своевременно выполнить коррекцию и избежать аварийного схода КА с орбиты.

В аспекте сравнения целесообразно заметить, что метод Гаусса-Эверхарта адаптирован специально для решения задач небесной механики, в частности, механизм выбора величины шага интегрирования осуществлен с учетом специфики плоской задачи двух тел [16]. Для других задач, в частности, описанных в п. 5.1, метод не дает такой эффективности. В то же время излагаемый в работе кусочно-интерполяционный метод не специализируется на решении задач небесной механики, оставаясь инвариантным для различных классов задач, включая жесткие.

При численном моделировании динамических колебательных процессов к проблеме повышения точности решения задачи Коши для систем дифференциальных уравнений естественно примыкают проблемы вычисления значений функций одной и двух переменных с заданной границей погрешности, проблемы повышения точности вычисления производных и ингегралов. С целью решения указанных проблем в главе 2 и главе 4 были описаны и математически обоснованы алогоритмы кусочно-интерполяционного приближения функций одной и двух переменных. Ниже представлено расширение кусочной интерполяции для приближения функций, производных и интегралов с приложением к численным дифференциальным моделям.

5.3. Повышение эффективности кусочно-интерполяционной обработки и организация хранения числовых данных. Рассматривается метод кусочно-интерполяционного приближения функций одной вещественной переменной с помощью интерполяционных полиномов Лагранжа и Ньютона, преобразованных к виду алгебраических полиномов с числовыми коэффициентами. Конструируемый метод является инвариантным относительно вида функции и диапазона независимой переменной, характеризуется высокой точностью приближения и одновременно малой временной сложности. Для класса повторяющихся функций после сохранения коэффициентов интерполяционных полиномов в памяти временная сложность конструируемого алгоритма измеряется количеством шагов схемы Горнера, то есть степенью полинома, – $n \times (t_y + t_c)$, t_y, t_c – время бинарного умножения и сложения. Повышение скорости вычисления повторяющихся функций при сохранении высокой точности приближения актуально при решении широкого класса задач прикладной математики, механики, физики и техники. В частности, в области небесной механики необходимо в режиме реального времени рассчитывать значения тригонометрических и гиперболических функций при вычислении элементов орбиты ИСЗ по координатам и составляющим скорости, при численном моделировании движения ИСЗ на основе дифференциальных уравнений, составленных относительно оскулирующих элементов орбиты, при вычислении положения и скорости спутника на околоземной орбите с использованием аналитических моделей (*SGP4*, *SDP4*, *SDP8*) [190, 192, 194]. Для решения ряда задач небесной механики также необходимо быстрое вычисление значений специальных функций, в частности, при разложении координат эллиптического движения в тригонометрические ряды используются функции Бесселя, при исследовании движения ИСЗ – функции наклона и функции эксцентриситета [196].

На основе представления интерполяционных полиномов в виде алгебраических полиномов с числовыми коэффициентами метод применяется

для приближения производных и первообразных. На основе кусочно-полиномиального приближения первообразной выполняется построение квадратурных формул [124]. Аналитическая и программная реализация квадратурных формул для произвольной степени полинома позволяет достигать сравнительно высокой точности вычисления интеграла. Малая временная сложность алгоритма позволяет применять метод для численного интегрирования сильно осциллирующих функций без выделения осциллирующих множителей в качестве весовых функций [73].

На основе кусочной интерполяции с использованием интерполяционного полинома Лагранжа, преобразованного к виду алгебраического полинома с числовыми коэффициентами, строится приближенное решение задачи Коши для ОДУ. Алгоритм построения метода аналогичен представленному в главе 3 для интерполяционного полинома Ньютона. Представлено математическое описание метода, его алгоритмизация и программная реализация. Анализируется вычислительная устойчивость, границы погрешности и временная сложность для модельных задач небесной механики, задачи Коши для дифференциального уравнения Бесселя. Описан численный эксперимент, исследуется возможность уменьшения погрешности без автоматического выбора параметров, изложенного в главе 3, с целью снижения трудоемкости метода.

5.3.1. Модификация кусочно-интерполяционной обработки и хранения данных на основе полинома Лагранжа. Для интерполяции функции $y = f(x)$, $x \in [a, b]$ полином Лагранжа может быть записан в виде

$$P_n(x) = \sum_{j=0}^{n} f(x_j) \prod_{\substack{r=0 \\ r \neq j}}^{n} (x - x_r) \; / \; \prod_{\substack{r=0 \\ r \neq j}}^{n} (x_j - x_r) \; , \qquad (5.15)$$

где x_r – узлы интерполяции. Можно вычислить значения $b_j = f(x_j) / \prod_{\substack{r=0 \\ r\neq j}}^{n}(x_j - x_r)$, $j=\overline{0, n}$, тогда $P_n(x) = \sum_{j=0}^{n} b_j \prod_{\substack{r=0 \\ r\neq j}}^{n}(x - x_r)$. Если при каждом $j=\overline{0, n}$ с использованием алгоритма [74]

$$\begin{aligned} &d_{kk} = d_{(k-1)(k-1)},\ d_{k(k-\ell)} = d_{(k-1)(k-\ell-1)} - d_{(k-1)(k-\ell)}\, x_{k-1},\ \ell=\overline{1, k-1}, \\ &d_{k0} = -d_{(k-1)0}\, x_{k-1},\ k=\overline{2, n},\ d_{11}=1,\ d_{10}=-x_0, \end{aligned} \tag{5.16}$$

найти коэффициенты полинома $P_{nj}(x) = \prod_{\substack{r=0 \\ r\neq j}}^{n}(x - x_r)$, $P_{nj}(x) = \sum_{\ell=0}^{n} d_{\ell j} x^{\ell}$, то $P_n(x) = \sum_{j=0}^{n} b_j \sum_{\ell=0}^{n} d_{\ell j} x^{\ell} = \sum_{\ell=0}^{n}\sum_{j=0}^{n} b_j d_{\ell j} x^{\ell}$. Отсюда $P_n(x) = \sum_{\ell=0}^{n} d_{\ell}\, x^{\ell}$, где $d_{\ell} = \sum_{j=0}^{n} b_j d_{\ell j}$.

Подробное описание алгоритма (5.16) и его матричная форма представлены в главе 2 соотношениями 2.6 – 2.9, ниже в программной реализации используется именно форма представления алгоритма (5.16). Такое преобразование программируется в общем случае. Вместе с тем, на основании численного эксперимента, компьютерная реализация дает меньшую погрешность в случае равноотстоящих узлов.

Пусть на отрезке $x \in [a, b]$ взяты равноотстоящие узлы для полинома (5.15): $x_i \in [a, b]$, $i \in \overline{0, n}$, $x_{i+1} - x_i = h$, $i \in \overline{0, n-1}$, $h = (b-a)\, n^{-1}$. В этом случае $x_j = x_0 + j\,h$, $x_r = x_0 + r\,h$, $x_0 = a$, $x_n = b$. Отсюда $\prod_{\substack{r=0 \\ r\neq j}}^{n}(x_j - x_r) = \prod_{\substack{r=0 \\ r\neq j}}^{n}(j-r)h$, $P_n(x) = \sum_{j=0}^{n} f(x_j) \prod_{\substack{r=0 \\ r\neq j}}^{n}(x - x_r)/(j-r)\,h^{-1}$. Вводится переменная $t = (x - x_0)\,h^{-1}$. Тогда $(x - x_r)\,h^{-1} = t - r$, $r \in \overline{0, n}$, и

$$P_n(x) = \sum_{j=0}^{n} f(x_j) \prod_{\substack{r=0 \\ r\neq j}}^{n}(t-r)/(j-r),\ t = (x - x_0)h^{-1}. \tag{5.17}$$

Аналогичное преобразование дано в [124]. Отличие составит перевод числителей из (5.17) в форму полиномов с целочисленными коэффициентами. В результате станет возможным аналитическое выражение первообразных, производных и организация итераций в правых частях ОДУ. Пусть числитель дроби $\prod_{\substack{r=0 \\ r\neq j}}^{n}(t-r)/(j-r)$ записан в виде

$$P_{nj}(t)=\prod_{r=0}^{n-1}(t-t_r), \tag{5.18}$$

где роль корней полинома играют последовательные натуральные числа, среди которых пропускается j: $t_r=\begin{cases} r, & r<j; \\ r+1, & r\geq j. \end{cases}$ По схеме (5.16) получится:

$$P_{nj}(t)=d_{0j}+d_{1j}t+d_{2j}t^2+...+d_{nj}t^n. \tag{5.19}$$

Коэффициенты полинома (5.19) будут целыми числами. Они могут быть априори вычислены и храниться в памяти компьютера как константы, не зависящие от интерполируемой функции, от диапазона и расположения её аргумента, причем это можно сделать для любого j и для всех степеней полинома n в априори заданных границах. Знаменатель в (5.17) отличается от числителя тем, что в нем $t=j$. В результате полином примет вид

$$P_n(x)=\sum_{j=0}^{n}f(x_j)\,\frac{d_{0j}+d_{1j}t+d_{2j}t^2+...+d_{nj}t^n}{d_{0j}+d_{1j}j+d_{2j}j^2+...+d_{nj}j^n}, \tag{5.20}$$

где $t=(x-x_0)h^{-1}$ Числитель и знаменатель в (5.20) вычисляется с помощью схемы Горнера. В принятых обозначениях (5.20) можно представить в виде

$$P_n(t)=\sum_{j=0}^{n}f(x_j)\,P_{nj}(t)/P_{nj}(j),\ t=(x-x_0)h^{-1}. \tag{5.21}$$

Если вычислить коэффициенты D_ℓ при равных степенях в (5.21), то

$$P_n(t)=\sum_{\ell=0}^{n} D_\ell\, t^\ell,\ t=(x-x_0)h^{-1}. \tag{5.22}$$

Приближение $f(x)\approx P_n(t)$ согласно (5.21), (5.22) влечет две разновидности приближения производных:

$$f'(x)\approx P_n'(x)=h^{-1}\sum_{j=0}^{n} f(x_j)\,\frac{d_{1j}+2d_{2j}t+\ldots+(n-1)d_{nj}t^{n-1}}{d_{0j}+d_{1j}j+d_{2j}j^2+\ldots+d_{nj}j^n}, \tag{5.23}$$

$$f'(x)\approx P_n'(x)=h^{-1}P_n'(t)=h^{-1}\sum_{\ell=1}^{n}\ell\, D_\ell\, t^{\ell-1}, \tag{5.24}$$

t из (5.22). На практике (5.23) несколько точнее (5.24).

Кусочная интерполяция. Если (5.21) – (5.24) применяются на малых подынтервалах с общими границами разбиения

$$[a,b]=\bigcup_{i=0}^{p-1}[a_i,b_i],\ b_i-a_i=(b-a)p^{-1}, \tag{5.25}$$

то точность приближения возрастет: функция и интеграл будут приближаться с точностью до $10^{-20}-10^{-19}$, производная – с точностью $10^{-16}-10^{-15}$ (*Delphi*, формат *extended*). Ниже предполагается, если не оговорено иное, что $p=2^k$, k – натуральное. Видоизменения (5.21), (5.22) формально совпадают между собой и с полиномом (5.15) при условии, что на одном и том же отрезке узлы одинаковы, одинаковы степени полиномов и интерполируется одна и та же функция. В этих же условиях все три полинома совпадают с интерполяционным полиномом Ньютона [124]. Для последнего в этом случае $\forall k\ge 0$ справедливо соотношение (см. глава 2, п. 2.1.1):

$$\begin{gathered}[a,b]=\bigcup_{i=0}^{2^k-1}[a_i,b_i],\ |f(x)-\Psi_{in}(t)|\le c\,2^{-k(n+1)}h^{n+1}\\ \forall i=\overline{0,2^k-1},\ \forall x\in[a_i,b_i],\end{gathered} \tag{5.26}$$

где предполагается, что $f(x)$ определена и непрерывно дифференцируема на $[a,b]$ $n+1$ раз. Здесь $\Psi_{in}(t)$ – интерполяционный полином Ньютона для интерполирования вперед на подынтервале $[a_i,b_i]$, $t=(x-a_i)h_i^{-1}$, h_i – шаг интерполяции на $[a_i,b_i]$, h – шаг интерполяции полинома $\Psi_{0n}(t)$ на $[a,b]$ (при $k=0$), поэтому $h=(b-a)n^{-1}$, $c=\max\limits_{[a,b]}\max\limits_{1\le n\le n_0}|f^{(n+1)}(x)|$, $n_0=\text{const}$, $c=\text{const}$.

При каждом i из (5.26) полином Ньютона имеет вид

$$\Psi_{in}(t)=f(x_{i0})+\sum_{j=1}^{n}\frac{\Delta^j f_{i0}}{j!}\prod_{r=0}^{j-1}(t-r),\ t=(x-a_i)h_i^{-1}, \tag{5.27}$$

узлы интерполяции $x_{ij}\in[a_i,b_i]$, $j\in\overline{0,n}$, $x_{i(j+1)}-x_{ij}=h_i$, $h_i=(b_i-a_i)n^{-1}$, $j\in\overline{0,n-1}$, $\Delta^j f_{i0}$ – конечная разность j-го порядка при $x_{i0}=a_i$, $i=\overline{0,p-1}$, p из (5.25), $p=2^k$. С учетом (5.26) последовательность полиномов $\Psi_{in}(t)$ равномерно сходится к $f(x)$ на $[a,b]$, если $k\to\infty$. Эти утверждения и (5.26) сохраняются для всех рассматриваемых видоизменений интерполяционных полиномов Лагранжа и Ньютона. В частности,

$$\begin{gathered}[a,b]=\bigcup_{i=0}^{2^k-1}[a_i,b_i],\ \ |f(x)-P_{in}(t)|\le c\,2^{-k(n+1)}h^{n+1}\\ \forall i=\overline{0,2^k-1},\ \forall x\in[a_i,b_i],\end{gathered} \tag{5.28}$$

где $P_{in}(t)$ – полином вида (5.21), построенный на $[a_i,b_i]$, $t=(x-a_i)h_i^{-1}$, $h_i=(b_i-a_i)n^{-1}$ – шаг интерполяции, $h=(b-a)n^{-1}$. В данном случае

$$P_{in}(t)=\sum_{j=0}^{n}f(x_{ij})\,P_{nj}(t)/P_{nj}(j)\ \ ,x\in[a_i,b_i],\ t=(x-a_i)h_i^{-1}, \tag{5.29}$$

аналогично, полином (5.22) примет вид

$$P_{in}(t)=\sum_{\ell=0}^{n}D_{i\ell}\,t^{\ell},\ x\in[a_i,b_i],\ t=(x-a_i)h_i^{-1}. \tag{5.30}$$

Как отмечалось, полином Ньютона (5.27) на каждом подынтервале также можно преобразовать к виду алгебраического полинома. Для этого преобразуется $\Psi_j(t) = \prod_{r=0}^{j-1}(t-r):\ t_r = r,\ r \le j-1$, согласно (5.16) получится

$$\Psi_j(t) = \widetilde{d}_{0j} + \widetilde{d}_{1j}t + \widetilde{d}_{2j}t^2 + ... + \widetilde{d}_{jj}t^j .$$

С умножением на $\frac{\Delta^j f_{i0}}{j!}$ (5.27) примет развернутую форму:

$$\Psi_{in}(t) = f(x_{i0}) + \sum_{j=1}^{n}(c_{0j} + c_{1j}t + c_{2j}t^2 + ... + c_{jj}t^j),\ x \in [a_i, b_i],\ t = (x-a_i)h_i^{-1}. \quad (5.31)$$

С приведением подобных в (5.31) полином примет вид

$$\Psi_{in}(t) = \sum_{j=0}^{n} C_{ij}\,t^j,\ x \in [a_i, b_i],\ t = (x-a_i)h_i^{-1}. \quad (5.32)$$

5.3.2. Модификация библиотеки стандартных программ для повышения эффективности обработки и организации хранения числовых данных. Исследуется кусочно-интерполяционное вычисление функций одной вещественной переменной в качестве основы расширяемой библиотеки стандартных программ. В [178, 179] представлены программные реализации описанных алгоритмов. Рассмотрены различные варианты хранения коэффициентов, дан алгоритм считывания [179]. Время вычисления функции определяется числом шагов схемы Горнера, с точностью до времени обращения к памяти. Исследуется эффективность программной реализации метода. Конструктивно достигается максимальная точность приближения функций при одновременной минимизации временной сложности [178].

Результаты подробного численного эксперимента и коды программ представлены в [178]. Стандартные функции и их композиции, например, $f(x) = sh(x)$, $f(x) = e^{-\cos x}$ на отрезке $[0, 1]$ приближаются полиномами Лагранжа 2-й степени с точностью 10^{-20} – 10^{-19} [178]. Аналогичные результаты

дает кусочная интерполяция по Ньютону (для степени полинома $n=3$). Следует подчеркнуть, что в случае приближения функций и их композиций с достаточно ограниченной областью значений (например, $\sin(x)$, $\cos(x)$, $e^{-\cos x}$) отрезок $[a, b]$ из (5.25) можно сдвигать на произвольную величину с такой же границей погрешности: кусочная интерполяция не требует редукции аргумента к основному промежутку [178].

Кусочно-интерполяционное приближение производных стандартных функций по аналогам соотношений (5.23), (5.24) на основе полинома Лагранжа вычисляется с точностью до 10^{-15} (степень полинома $n=6$). Для развернутой формы (5.31) полинома Ньютона производные стандартных функций вычисляются с точностью до 10^{-16} (при этом степень полинома $n=9$) [178].

Программная реализация метода на основе полинома Ньютона с автоматизированным выбором параметров, обеспечивающих вычисление функции с априори заданной границей абсолютной погрешности:

```
program VPI_function_Newton_min_k; {$APPTYPE CONSOLE} uses SysUtils;
const abs_error=1.0e-6; nn=20; type M=array[0..nn] of extended;
MM=array[0..nn,0..nn] of extended; var n_min,k_min:byte;a0,b0:extended; d: MM;
k_output:integer; {const n_fix=2; type Coeff = array[0..n_fix] of extended;
 var f: file of Coeff; dd:Coeff;}
function u(x:extended):extended; begin u:=sin(x) end;
procedure Konech_Raznoct(n:byte;a00,h: extended; var dy:MM); var i,k:byte;x:M;
begin for i:=0 to n do x[i]:=a00+i*h;
for i:=0 to n-1 do dy[1,i]:=u(x[i+1])-u(x[i]);
for k:=2 to n do for i:=0 to n-k do dy[k,i]:=dy[k-1,i+1]-dy[k-1,i] end;
procedure Viet(n:byte;var d:MM); var q:MM;k,i,j:byte;
begin q[1,1]:=1; q[1,0]:=0; for k:=2 to n do begin
q[k,0]:=-q[k-1,0]*(k-1); for i:=1 to k-1 do
q[k,k-i]:=q[k-1,k-i-1]-q[k-1,k-i]*(k-1); q[k,k]:=q[k-1,k-1] end;
 for j:=1 to n do  for i:=0 to j do d[i,j]:=q[j,i] end;
procedure pi_Newton(print:boolean;n,k:byte; var cond:boolean);
var xpr,t,hpr,s,p,h:extended; factorial:integer;flag:boolean;
dy:MM; a00,b00,vel_podint: extended; i,j,l: byte; b,y,a: M;kk, n_pod:int64;
begin vel_podint:=(b0-a0)/exp(k*ln(2));
a00:=a0; b00:=a00+vel_podint; kk:=0;
n_pod:=0; flag:=true; while a00<=b0-vel_podint do
begin h:=(b00-a00)/n; hpr:=h/33; Konech_Raznoct(n,a00,h,dy); xpr:=a00;
while xpr<=b00 do begin t:=(xpr-a00)/h; factorial:=1;
for j:=1 to n do begin factorial:=factorial*j; b[j]:=dy[j,0]/factorial; end;
a[0]:=u(a00); for l:=1 to n do begin s:=0; for j:=l to n do s:=s+d[l,j]*b[j];
a[l]:=s  end; p:=a[n]; for i:=n-1 downto 0 do p:=p*t+a[i];
if print then begin inc(kk); if flag then begin
writeln('n=',n,' n_pod=',n_pod); for i:= 0 to n do
writeln('a[',i,']=',a[i]);{for i:=0 to n_fix do dd[i]:=a[i];
write(f, dd);} flag:=false; end; if kk=k_output then begin
 writeln(xpr,' ',p,' ',abs(p-u(xpr)));kk:=0;end; end;
```

```
if abs(p-u(xpr))>abs_error then cond:=false; xpr:=xpr+hpr end;
a00:=a00+vel_podint; b00:=a00+vel_podint; n_pod:=n_pod+1;flag:=true; end; end;
procedure vpi_f(a0,b0:extended; var n_min,k_min:byte);
var n,k:byte; cond:boolean; begin k:=0; repeat n:=1; repeat cond:=true;
pi_Newton(false,n,k,cond); if cond=true then begin n_min:=n; k_min:=k;
pi_Newton(true,n,k,cond); writeln; writeln('n_min = ',n_min,'  k_min = ',k_min);
exit; end else n:=n+1; until n>15; k:=k+1 until k>17; end;
begin {AssignFile(f,'F:\sin.bin'); Rewrite(f);} k_output:=50;
 a0:=0; b0:=1; Viet(nn, d); vpi_f(a0, b0, n_min, k_min); {CloseFile(f);}
 readln; end.
```

Граница абсолютной погрешности задается в разделе констант (*abs_error*). Номер подынтервала определяется значением *n_pod*, переменная *a* определяет массив коэффициентов полинома на подынтервале. В результате выполнения программы в консоль выводится номер подынтервала и массив значений коэффициентов полинома, соответствующих данному номеру. Рассчитанные коэффициенты фиксируются в разделе констант в виде двумерного массива, определяемого переменной *dd* в следующей программе:

```
program NewtonConst; {$APPTYPE CONSOLE} uses SysUtils;
const aa=0 ;bb=1; hh=1; n=5; h=hh/n; var x,t:extended; i,rr:integer;
function f(var xx:extended): extended; begin f:=sin(xx) end;
function func(var x: extended): extended;
type vec0=array[0..0,0..5] of extended; const dd:vec0=
((0, 0.19999560375268066, 0.00000975661575774,
-0.00134093110346268, 0.00000258073717352, 0.00000232079291198));
function gorner11 (var rr: integer; const dd:vec0):extended; var pp:extended;
begin pp:=dd[rr,n]; for i:=1 to n do pp:=pp*t+dd[rr,n-i]; gorner11:=pp; end;
begin if x<1 then rr:=trunc((x-aa)/hh) else rr:=trunc((x-aa)/hh)-1;
t:=(x-(aa+rr*hh))/h; func:=gorner11(rr,dd); end; begin x:=1/21;
writeln('func(x)=', func(x),';  pogr=',abs(func(x)-f(x)));
readln; end.
```

Результат работы программы:

```
func(x)= 9.50936103572303E-0002;  pogr= 5.77227524499266E-0007
```

Таким образом, вычисление синуса от произвольного аргумента на отрезке [0,1] выполняется с точностью порядка 10^{-7} за время $5(t_y + t_c)$. Для реализации процесса записи коэффициентов аппроксимирующих полиномов в типизированный файл в представленной выше программе достаточно раскомментировать требуемые части программы и удалить фрагмент вывода значений коэффициентов в консоль. В этом случае программа вычисления действительной функции одной действительной переменной примет

следующий вид (в данном случае $f(x)=\sin x$ на отрезке $[0,1]$ приближается полиномом Ньютона степени $n=2$ с погрешностью порядка 10^{-19}):

```
program NewtonFile; {$APPTYPE CONSOLE} uses SysUtils;
const aa=0;bb=1; hh=3.81469726562500E-0006; n=2; h = hh/n;
var i:integer; rr:int64; x,t:extended;
function f(var xx:extended):extended; begin f:=sin(xx) end;
function func(var x:extended):extended; type Coeff = array[0..n] of extended;
var f: file of Coeff; dd: Coeff;
function gorner11 (const dd:Coeff): extended; var pp:extended;
begin pp:=dd[n]; for i:=1 to n do  pp:=pp*t+dd[n-i]; gorner11:=pp; end;
begin AssignFile(f,'F:\sin.bin'); Reset(f);
if x<bb then rr:=trunc((x-aa)/hh) else rr:=trunc((x-aa)/hh)-1;
seek(f,rr); read(f,dd); t:=(x-(aa+rr*hh))/h;
func:=gorner11(dd); CloseFile(f); end;
begin x:=0.23; writeln('x=',x,'; func(x)=',func(x),'pogr=',abs(func(x)-f(x)));
readln; end.
```

Результат работы программы:

```
x=2.30000E-0001; func(x)=2.27977523535188E-0001; pogr=3.38813178901720E-0019
```

В данной программе коэффициенты интерполяционного полинома считываются из файла *sin.bin* в соответствии с номером подынтервала, который рассчитывается для требуемого значения аргумента функции. Число хранимых в файле коэффициентов – 3×2^{18} (число подынтервалов $p=2^{18}$, степень полинома $n=2$). В результате значение синуса вычисляется с точностью порядка 10^{-19} для произвольно фиксированного значения аргумента из [0,1] за время $2(t_y+t_c)$. Вариант программы с хранением коэффициентов в файле позволяет достигать максимальной точности вычисления стандартных и специальных функций с минимальной временной сложностью [179, 180].

5.3.3. Стандартизация программ интегральной обработки данных. Непосредственно из (5.20), (5.21), (5.25) вытекают формулы приближенного вычисления интегралов. С разбиением (5.25)

$$\int_a^b f(x)dx=\sum_{i=0}^{p-1}\int_{a_i}^{b_i} f(x)\,dx,\ \int_{a_i}^{b_i} f(x)dx\approx\int_{a_i}^{b_i} P_{in}(x)\,dx, \qquad (5.33)$$

где $P_{in}(x)$ – полином (5.20), построенный на подынтервале $[a_i, b_i]$:

$$P_{in}(t)=\sum_{j=0}^{n} f(x_{ij})\,(D_{0j}+D_{1j}t+D_{2j}t^2+\ldots+D_{nj}t^n),$$
$$x\in[a_i,\,b_i],\ t=(x-a_i)h_{pn}^{-1}, \tag{5.34}$$

где $D_{\ell j}=d_{\ell j}/(d_{0j}+d_{1j}j+d_{2j}j^2+\ldots+d_{nj}j^n)$, $\ell\in\overline{0,\,n}$; $h_{pn}=(b-a)/p/n$ – одинаковый на всех подынтервалах шаг интерполяции. Отсюда выводится приближение [181]:

$$\int_a^b f(x)\,dx\approx(b-a)p^{-1}n^{-1}\sum_{i=0}^{p-1}\sum_{j=0}^{n}c_{nj}f(x_{ij}), \tag{5.35}$$

где $c_{nj}=D_{0j}n+2^{-1}D_{1j}n^2+3^{-1}D_{2j}n^3+\ldots+(n+1)^{-1}D_{nj}n^{n+1}$ зависит от степени интерполяционного полинома и от номера узла интерполяции, но не зависит от номера подынтервала и от интегрируемой функции. Коэффициенты c_{nj} могут быть вычислены априори, быть хранимыми константами для вычисления интеграла от произвольной функции. В частности c_{nj} можно сохранять в разделе описаний констант программы и непосредственно использовать в разделе инструкций программы вычисления интегралов.

Имеет место

Теорема 5.1. На основе кусочной интерполяции интегрируемой функции с помощью полиномов Лагранжа интеграл вычисляется из (5.35), где $f(x_{ij})$ – значение функции в j-м узле интерполяции на i-м подынтервале из (5.25), коэффициенты и параметры определяются из (5.16), (5.34).

Если на каждом отрезке из (5.35) $f(x)$ вычисляется с погрешностью (5.28), где $h=(b-a)n^{-1}$, $b_i-a_i=(b-a)2^{-k}$, то

$$\left|\int_a^b f(x)\,dx-\sum_{i=0}^{2^k-1}\int_{a_i}^{b_i}P_{in}(x)\,dx\right|\le c\,2^{-k(n+1)}(b-a)^{n+2}/n^{n+1}. \tag{5.36}$$

Если $k\to\infty$, то $c\,2^{-k(n+1)}(b-a)^{n+2}/n^{n+1}\to 0$. Формулу (5.35) можно интерпретировать как разновидность формул Ньютона-Котеса. В табл. 5.5 даны

значения c_{nj}/n для различных степеней интерполяционных полиномов Лагранжа.

Таблица 5.5

Значения коэффициентов c_{nj}/n приближения интеграла (5.35)

n	$c_{nj}/n,\ j\in\overline{0,n}$
1	(0.50000000000000000000, 0.50000000000000000000)
2	(0.16666666666666666667, 0.66666666666666666667, 0.16666666666666666667)
3	(0.12500000000000000000, 0.37500000000000000000, 0.37500000000000000000, 0.12500000000000000000)
4	(0.07777777777777777778, 0.35555555555555555556, 0.13333333333333333333, 0.35555555555555555556, 0.07777777777777777778)
5	(0.06597222222222222222, 0.26041666666666666667, 0.17361111111111111111, 0.17361111111111111111, 0.26041666666666666667, 0.06597222222222222222)
6	(0.04880952380952380952, 0.25714285714285714285, 0.03214285714285714285, 0.32380952380952380952, 0.03214285714285714285, 0.25714285714285714285, 0.04880952380952380952)
7	(0.04346064814814814815, 0.20700231481481481481, 0.07656250000000000000, 0.17297453703703703703, 0.17297453703703703703, 0.07656250000000000000, 0.20700231481481481481, 0.04346064814814814815)
8	(0.03488536155202821869, 0.20768959435626102293, -0.03273368606701940035, 0.37022927689594356261, -0.16014109347442680776, 0.37022927689594356261, -0.03273368606701940035, 0.20768959435626102293, 0.03488536155202821869)
9	(0.03188616071428571429, 0.17568080357142857143, 0.01205357142857142857, 0.21589285714285714286, 0.06446607142857142857, 0.06446607142857142857, 0.21589285714285714286, 0.01205357142857142857, 0.17568080357142857143, 0.03188616071428571429)
10	(0.02683141483619261397, 0.17753594142483031372, -0.08104357062690396024, 0.45494628827962161296, -0.43515512265512265512, 0.71376463043129709796, -0.43515512265512265512, 0.45494628827962161296, -0.08104357062690396024, 0.17753594142483031372, 0.02683141483619261397)

Замечание 5.3. Для повышения точности вычислений при составлении табл. 5.5 использовался язык *Python* с модулем *decimal*.

Вычисление интеграла из (5.35) обладает естественным параллелизмом. Временная сложность $T(R)$ параллельного вычисления интеграла составит [181]

$$T(p(n+1)) = 2t_y + \lfloor \log_2 p + \log_2 (n+1) \rfloor t_c \sim (\log_2 p + \log_2 (n+1))\, t_c = \\ = O\,((\log_2 p + \log_2 (n+1))). \tag{5.37}$$

При ограничении степени полинома, $n \le n_0 = \text{const}$, – $T(p(n+1)) = O\,(\log_2 p\,)$.

Аналогичное вычисление интеграла на основе полинома Ньютона выполнено для (5.31). С этой целью $P_{in}(x)$ в (5.33) заменяется на $\Psi_{in}(t)$ из (5.21) и выводится формула [181]

$$\int_a^b f(x)\,dx \approx (b-a)\,p^{-1}n^{-1}\sum_{i=0}^{p-1}\left(n\,f(x_{i0})+\sum_{j=1}^{n}\Delta^j f_{i0}\,c_{Nnj}\right), \tag{5.38}$$

где $c_{Nnj}=c_{0j}\,n+2^{-1}c_{1j}n^2+3^{-1}c_{2j}n^3+\ldots+(j+1)^{-1}c_{jj}\,n^{j+1}$, $c_{\ell j}=d_{\ell j}/j!$, $\ell\in\overline{0,\,j}$, $j\in\overline{1,\,n}$. При этом c_{Nnj} зависят от степени интерполяционного полинома, от номера узла интерполяции, но не зависят от номера подынтервала и вида функции. Эти коэффициенты вычисляются априори, записываются в память компьютера (в раздел описания констант программы), применяются для вычисления интеграла от произвольной функции. Аналогично (5.36), погрешность (5.38) оценивается из неравенства

$$\left|\int_a^b f(x)\,dx-\sum_{i=0}^{2^k-1}\int_{a_i}^{b_i}\Psi_{in}(t)\,dx\right|\le c\,2^{-k\,(n+1)}(b-a)^{n+2}/n^{n+1}.$$

Вычисление интеграла на основе первообразной от непрерывного кусочно-интерполяцинного приближения интегрируемой функции выполняется следующим образом. Пусть на отрезке (5.25) выполнено приближение (5.34),

$$f(x)\approx P_{in}(t),\ x\in[a,\,b]\cap[a_i,\,b_i].$$

Номер подынтервала, при котором $x\in[a_i,\,b_i]$, $t=(x-a_i)h_{pn}^{-1}$, h_{pn} из (5.34), определяется по формуле [178]

$$i=[(x-a)/((b-a)\,p^{-1})],$$

$[\alpha]$ – целая часть числа α. Пусть функция $f(x)$ непрерывна $\forall\,x\in[a,b]$ из (2.25), тогда она интегрируема, ее первообразная

$$\Phi(x)=\int_a^x f(x)\,dx \tag{5.39}$$

непрерывна на $[a, b]$. Приближение подынтегральной функции, обозначаемое $F_n(x)$, рассматривается как самостоятельная функция, определенная на всем $[a, b]$,

$$F_n(x) \approx f(x) \ \forall x \in [a, b],$$

при этом определяющими являются соотношения:

$$\begin{aligned} &\forall x \in [a, b], \forall\ i = 0, 1, \ldots, p-1: \\ &x \in [a_i, b_i] \ \Rightarrow F_n(x) = P_{in}(t), \ t = (x - a_i) h_{pn}^{-1}, \end{aligned} \tag{5.40}$$

$P_{in}(t)$ из (5.34).

Начальный и конечный узел интерполяции полинома (5.34) всегда находятся соответственно на левой и правой границе подынтервала, поэтому

$$\begin{aligned} &F_n(a_{i+1}) = P_{(i+1)n}(t)\Big|_{x=a_{i+1}} = P_{(i+1)n}(0) = f(a_{i+1}) = \\ &f(b_i) = P_{in}(n) = P_{in}(t)\Big|_{x=b_i} = F_n(b_i), \ \ i = 0, 1, \ldots, p-2. \end{aligned} \tag{5.41}$$

Согласно (5.34), (5.40), (5.41) функция $F_n(x)$ непрерывна $\forall x \in [a, b]$, следовательно, интегрируема на $[a, b]$. Ее первообразная

$$\Phi_n(x) = \int_a^x F_n(x)\, dx + C \tag{5.42}$$

при $\forall\ C = \text{const}$ определена и непрерывна $\forall x \in [a, b]$. Первообразную (5.42) можно построить из полиномов (5.34). В случае $x \in [a_i, b_i]$ интеграл

$$\int_{a_i}^x P_{in}(t)\, dx = \sum_{j=0}^{n} f(x_{ij})\, h_{pn} \int_{a_i}^x (D_{0j} + D_{1j} t + D_{2j} t^2 + \ldots + D_{nj} t^n)\, dt \tag{5.43}$$

представляет собой первообразную полинома (5.34). Равенство (5.43) эквивалентно равенству

$$\int_{a_i}^{x} P_{in}(t)\,dx = \\ = \sum_{j=0}^{n} f(x_{ij})\, h_{pn}\left(D_{0j} t\Big|_{a_i}^{x} + 2^{-1} D_{1j} t^2\Big|_{a_i}^{x} + 3^{-1} D_{2j} t^3\Big|_{a_i}^{x} + \ldots + (n+1)^{-1} D_{nj} t^{n+1}\Big|_{a_i}^{x} \right). \tag{5.44}$$

Для полинома (5.44) вводится обозначение

$$P_{INT\,in}(x) = \sum_{j=0}^{n} f(x_{ij})\, h_{pn}\left(D_{0j} t\Big|_{a_i}^{x} + 2^{-1} D_{1j} t^2\Big|_{a_i}^{x} + 3^{-1} D_{2j} t^3\Big|_{a_i}^{x} + \ldots \right. \\ \left. + (n+1)^{-1} D_{nj} t^{n+1}\Big|_{a_i}^{x} \right), \quad x \in [a_i, b_i], \tag{5.45}$$

так что

$$\int_{a_i}^{x} P_{in}(t)\,dx = P_{INT\,in}(x). \tag{5.46}$$

В (5.45) каждая нижняя подстановка равна нулю, поэтому

$$\int_{a_r}^{b_r} P_{rn}(t)\,dx = \int_{a_r}^{a_{r+1}} P_{rn}(t)\,dx = P_{INT\,rn}(a_{r+1}) \quad \forall\, r = 0, 1, \ldots, p-1. \tag{5.47}$$

Из (5.46), (5.47) следует

$$\int_{a_i}^{x} P_{in}(t)\,dx + \int_{a_{i-1}}^{a_i} P_{(i-1)n}(t)\,dx = P_{INT\,in}(x) + P_{INT\,(i-1)n}(a_i). \tag{5.48}$$

С учетом (5.40)

$$\int_{a_i}^{x} F_n(x)\,dx + \int_{a_{i-1}}^{a_i} F_n(x)\,dx = P_{INT\,in}(x) + P_{INT\,(i-1)n}(a_i),$$

или, с учетом (5.25),

$$\int_{a_{i-1}}^{x} F_n(x)\,dx = P_{INT\,in}(x) + P_{INT\,(i-1)n}(b_{i-1}). \tag{5.49}$$

Аналогично, добавление к обеим частям (5.48) интеграла $\int_{a_{i-2}}^{a_{i-1}} P_{(i-2)n}(t)\,dx$ влечет

$$\int_{a_i}^{x} P_{in}(t)\,dx + \int_{a_{i-1}}^{a_i} P_{(i-1)n}(t)\,dx + \int_{a_{i-2}}^{a_{i-1}} P_{(i-2)n}(t)\,dx = P_{INT\,in}(x) + P_{INT\,(i-1)n}(a_i) + P_{INT\,(i-2)n}(a_{i-1}).$$

С учетом определения (5.40) функции $F_n(x)$,

$$\int_{a_i}^{x} F_n(x)\,dx + \int_{a_{i-1}}^{a_i} F_n(x)\,dx + \int_{a_{i-2}}^{a_{i-1}} F_n(x)\,dx = P_{INT\,in}(x) + P_{INT\,(i-1)n}(a_i) + P_{INT\,(i-2)n}(a_{i-1}).$$

Так как $a_0 = a$, в продолжение процесса получится

$$\begin{aligned}\int_{a}^{x} F_n(x)\,dx &= P_{INT\,in}(x) + P_{INT\,(i-1)n}(b_{i-1}) + P_{INT\,(i-2)n}(b_{i-2}) + \\ &+ P_{INT\,(i-3)n}(b_{i-3}) + \cdots + P_{INT\,0n}(b_0), \quad x \in [a_i\,,\, b_i].\end{aligned} \tag{5.50}$$

Из изложенного вытекает

Лемма 5.1. Первообразная (5.39) может вычисляться с помощью непрерывного кусочно-интерполяционного приближения (5.40) интегрируемой функции из соотношения

$$\Phi(x) \approx \int_{a}^{x} F_n(x)\,dx = P_{INT\,in}(x) + \sum_{\ell=1}^{i} P_{INT\,(i-\ell)n}(b_{i-\ell}), \quad x \in [a_i\,,\, b_i], \tag{5.51}$$

где $P_{INT\,in}(x)$ из (5.45), $[a_i\,,\, b_i]$ – из (5.25), $P_{INT\,rn}(b_r)$ вычисляется из (5.45) при $i = r$, $x = b_r$.

Первообразная (5.42), (5.50) представляет собой непрерывное аналитическое приближение первообразной подынтегральной функции (5.39), если первообразная от последней аналитического выражения не имеет. Из (5.50), (5.51) при $x = b$ получается формула приближенного вычисления интеграла

$$\begin{aligned}\int_{a}^{b} F_n(x)\,dx &= P_{INT\,(p-1)n}(b_{p-1}) + P_{INT\,(p-2)n}(b_{p-2}) + \ldots \\ &\ldots + P_{INT\,(i-1)n}(b_{i-1}) + P_{INT\,(i-2)n}(b_{i-2}) + \ldots + P_{INT\,1\,n}(b_1) + P_{INT\,0n}(b_0).\end{aligned} \tag{5.52}$$

Более точно, имеет место

Теорема 5.2. В условиях леммы 5.1

$$\int_a^b f(x)\,dx \approx \int_a^b F_n(x)\,dx = \sum_{i=0}^{p-1} P_{INT\,i\,n}(b_i)\,, \tag{5.53}$$

где $F_n(x)$ определяется из (5.40), $P_{INT\,i\,n}(b_i)$ вычисляется из (5.45) при $x = b_i$.

В обозначении (5.42) первообразная (5.50) примет вид

$$\Phi_n(x) = P_{INT\,i\,n}(x) + \sum_{\ell=1}^{i} P_{INT\,(i-\ell)\,n}(b_{i-\ell}),\ \ \forall x \in [a_i\,,\, b_i]\,,\ \forall i \in \overline{0,\ p-1}\,. \tag{5.54}$$

Для (5.54) по основной теореме интегрального исчисления должно выполняться равенство

$$\int_a^b F_n(x)\,dx = \Phi_n(b) - \Phi_n(a)\,. \tag{5.55}$$

Более подробно,

$$\begin{aligned}
&\int_a^b F_n(x)\,dx = \\
&= P_{INT\,(p-1)\,n}(b_{p-1}) + P_{INT\,(p-2)\,n}(b_{p-2}) + \ldots + \\
&\quad + P_{INT\,(i-1)\,n}(b_{i-1}) + P_{INT\,(i-2)\,n}(b_{i-2}) + \ldots + P_{INT\,1\,n}(b_1) + P_{INT\,0\,n}(b_0) - \\
&\quad - P_{INT\,(p-1)\,n}(a_{p-1}) - P_{INT\,(p-2)\,n}(a_{p-2}) - \ldots - \\
&\quad - P_{INT\,(i-1)\,n}(a_{i-1}) - P_{INT\,(i-2)\,n}(a_{i-2}) - \ldots - P_{INT\,1\,n}(a_1) - P_{INT\,0\,n}(a_0)\,.
\end{aligned} \tag{5.56}$$

Из (5.45) $P_{INT\,r\,n}(a_r) = 0,\ r = 0, 1, \ldots, p-1,$ и из (5.55), (5.56) $\Phi_n(b) - \Phi_n(a) = \Phi_n(b) = \int_a^b F_n(x)\,dx = \sum_{i=0}^{p-1} P_{INT\,i\,n}(b_i)$. Окончательно,

$$\int_a^b f(x)\,dx \approx \Phi_n(b) = \int_a^b F_n(x)\,dx = \sum_{i=0}^{p-1} P_{INT\,i\,n}(b_i)\,,$$

что является разновидностью записи (5.53). Для коэффициентов полинома $P_{INT\,i\,n}(x)$ из (5.45) вводится обозначение

$$D_{INT\,\ell\,j} = (\ell+1)^{-1} D_{\ell\,j},\ \ell = 0,1,\ldots,n. \qquad (5.57)$$

Полином (5.45) примет вид

$$P_{INT\,i\,n}(x) = h_{pn} \sum_{j=0}^{n} f(x_{i\,j}) \sum_{\ell=0}^{n} D_{INT\,\ell\,j}\, t^{\ell+1}\Big|_{a_i}^{x},\ \ x \in [a_i, b_i],\ i \in \overline{0,\ p-1}. \qquad (5.58)$$

В табл. 5.6 представлены коэффициенты (5.57) полинома (5.58), поделенные на n, для степени $n = 4$.

Таблица 5.6

Значения коэффициентов $D_{INT\,\ell\,j} / n$ для приближения первообразной в случае полинома Лаганжа степени $n = 4$

j	$D_{INT\,\ell\,j} / n,\ \ell \in \overline{0,n}$
0	(1, -1.04166666666666666667E+0, 4.86111111111111111111E-1, -1.04166666666666666667E-1, 8.33333333333333333333E-3)
1	(0, 2.00000000000000000000E+0, -1.44444444444444444444E+0, 3.75000000000000000000E-1, -3.33333333333333333333E-2)
2	(0, -1.50000000000000000000E+0, 1.58333333333333333333E+0, -5.00000000000000000000E-1, 5.00000000000000000000E-2)
3	(0, 6.66666666666666666667E-1, -7.77777777777777777778E-1, 2.91666666666666666667E-1, -3.33333333333333333333E-2)
4	(0, -1.25000000000000000000E-1, 1.52777777777777777778E-1, -6.25000000000000000000E-2, 8.33333333333333333333E-3)

Сумму (5.52), аналогично, (5.51) можно вычислять параллельно с использованием схемы сдваивания. С оговоркой относительно распараллеливания вычисления двойной суммы (5.52), с точностью до числового множителя сохранится оценка (5.37).

5.3.4. Численный эксперимент и сравнение эффективности модифицированного метода с известными. Программная реализация кусочно-интерполяционного вычисления интеграла на основе полиномов Лагранжа, соотношений (5.33) – (5.35) и коэффициентов из табл. 5.5 имеет вид:

```
program PIL_integral_n1_10; {$APPTYPE CONSOLE} uses SysUtils;
var a,b,S:extended; n:byte; p:int64;
function f(x:extended):extended; begin f:=cos(x)*exp(sin(x)){cos(x)}
{sqrt(1-0.5*sqr(sin(x)))} {exp(x/2)+cos(4*x)} {x*exp(-x)*cos(2*x)} end;
function exact_int(a,b:extended):extended;
begin exact_int:=exp(sin(b))-exp(sin(a)){sin(b)-sin(a)} {1.35064388104767550252}
{2*(exp(Pi)-1)} {1/25*(exp(-2*Pi)*(3-10*Pi)-3)} end;
function PIL_int(a,b:extended; n:byte; p:int64):extended;
const c:array [1..10, 0..5] of extended =((0.5, 0, 0, 0, 0, 0),
(0.16666666666666666667,0.66666666666666666667, 0, 0, 0, 0),
```

```
(0.125, 0.375, 0, 0, 0, 0), (0.07777777777777777778, 0.35555555555555555556,
 0.13333333333333333333, 0, 0, 0), (0.065972222222222222222,
0.260416666666666666667, 0.173611111111111111111, 0, 0, 0),
(0.048809523809523809524, 0.257142857142857142857, 0.032142857142857142857,
0.323809523809523809524, 0, 0), (0.043460648148148148148,
0.207002314814814814815, 0.0765625,0.172974537037037037037, 0, 0),
(0.034885361552028218695, 0.207689594356261022928,-0.032733686067019400353,
0.370229276895943562610,-0.160141093474426807760, 0), (0.031886160714285714286,
0.175680803571428571429,0.012053571428571428571, 0.215892857142857142857,
0.064486607142857142857, 0),(0.026834148361926139704, 0.177535941424830313719,
-0.081043570626903960237,0.454946288279621612955, -0.435155122655122655123,
0.713764630431297097963)); type coeffs = array[0..5] of extended;
x_arr = array[0..10] of extended; var ai,hp,hpn,s1,s2:extended; j:byte;i:int64;
s2_arr:coeffs; x:x_arr;begin hp:=(b-a)/p; hpn:=hp/n; s1:=0; ai:=a; i:=0;
for j:=1 to n div 2 do s2_arr[j]:=0; repeat for j:=1 to n do x[j]:=ai+j*hpn;
for j:=1 to n div 2 - 1 do s2_arr[j]:=s2_arr[j]+f(x[j])+f(x[n-j]); if n mod 2
<>0 then s2_arr[n div 2]:=s2_arr[n div 2]+f(x[n div 2])+f(x[n div 2 +1]) else
s2_arr[n div 2]:=s2_arr[n div 2]+f(x[n div 2]);
if i < p-1 then s1:=s1+f(x[n]); inc(i); ai:=a+i*hp; until i = p;
s2:=0; for j:=1 to n div 2 do s2:=s2+c[n,j]*s2_arr[j];
s1:=c[n,0]*(f(a)+2*s1+f(b)); PIL_int:=(s1+s2)*hp; end;
begin a:=0; b:=Pi/2; n:=10; p:=16; S:=PIL_int(a, b, n, p);
writeln('S=',S,' Pogr=',abs(S-exact_int(a,b))); readln; end.
```

На основе работы данной программы выполнен численный эксперимент, результаты которого даны в табл. 5.7.

Таблица 5.7

Значения параметров и погрешности вычисления интегралов по результатам численного эксперимента

Интеграл	Приближение интеграла	Абсолютная погрешность приближения	Параметры программы
$\int_0^{\pi/2} \cos(x)\, e^{\sin(x)} dx$	1.71828182845905e+00	0.000**e+00**	$n = 5,\ p = 512$
$\int_0^{500} \cos(x)\, e^{\sin(x)} dx$	-3.73603552314934e-01	5.421**e-20**	$n = 9,\ p = 4096$
$\int_0^{\pi/2} \cos(x)\ dx$	1.00000000000000e+00	0.000**e+00**	$n = 6,\ p = 32$
$\int_0^{\pi/2} \sqrt{1 - 2^{-1}\sin^2(x)}\, dx$	1.35064388104768e+00	0.000**e+00**	$n = 2,\ p = 64$
$\int_0^{2\pi} \left(e^{x/2} + \cos(4x)\right) dx$	4.42813852655585e+01	0.000**e+00**	$n = 5,\ p = 1024$
$\int_0^{2\pi} \left(x\, e^{-x} \cos(2x)\right) dx$	-1.22122604618968e-01	0.000**e+00**	$n = 7,\ p = 4096$

Во втором столбце табл. 5.7 – рассчитанное приближенное значение интеграла, представленного в первом столбце той же строки, в третьем столбце – абсолютная погрешность приближения, в четвертом – параметры программы.

Программная реализация алгоритма вычисления интеграла на основе первообразной от кусочно-интерполяцинного приближения интегрируемой функции с помощью полинома Лагранжа и коэффициентов табл. 5.6 имеет вид:

```
program PIL_integral_n4_perv; {$APPTYPE CONSOLE} uses SysUtils;
 var a,b,S:extended; p:int64;
function f(x:extended):extended; begin f:=cos(x)*exp(sin(x)){cos(x)}
{sqrt(1-0.5*sqr(sin(x)))}{exp(x/2)+cos(4*x)}{x*exp(-x)*cos(2*x)} end;
function exact_int(a,b:extended):extended;
begin exact_int:=exp(sin(b))-exp(sin(a))
{sin(b)-sin(a)}{1.35064388104767550252}{2*(exp(Pi)-1)}
{1/25*(exp(-2*Pi)*(3-10*Pi)-3)} end;
function PIL_int(a,b:extended; p:int64):extended;
const n=4; Dint:array[0..n,0..n] of extended =
((1, -1.04166666666666666667, 4.8611111111111111111E-1,
 -1.04166666666666666667E-1, 8.33333333333333333333E-3),
(0, 2, -1.44444444444444444444E+0, 3.75E-1, -3.33333333333333333333E-2),
(0, -1.5, 1.58333333333333333333, -5.0E-1, 5.0E-2),
(0, 6.66666666666666666667E-1, -7.77777777777777777778E-1,
2.91666666666666666667E-1, -3.33333333333333333333E-2),
 (0, -1.25E-1, 1.52777777777777777778E-1,
-6.25E-2, 8.33333333333333333333E-3));
var ai,hp,hpn,s,pp2:extended; j,r:byte;i:int64; c:array[0..n] of extended;
begin  for j:=0 to n do begin
pp2:= Dint[j][n]; for r:=1 to n do pp2 := pp2*n + Dint[j][n-r];
c[j]:=pp2; end; hp:=(b-a)/p; hpn:=hp/n; i:=0; ai:=a; s:=0;
repeat for j:=0 to n do s:=s+c[j]*(f(ai+j*hpn));  inc(i); ai:=a+i*hp;
until i = p; PIL_int:=s*hp; end;
begin a:=0; b:=Pi/2; p:=1024; S:=PIL_int(a, b, p);
writeln('S=', S, ' Pogr=', abs(S-exact_int(a,b))); readln; end.
```

Реализация дана для степени $n=4$. Результаты численных экспериментов с программой *PIL_integral_n4_perv* приведены в табл. 5.8.

Таблица 5.8

Результаты приближенного вычисления интегралов с помощью программы PIL_integral_n4_perv

Интеграл	Приближение интеграла	Абсолютная погрешность приближения	Параметры программы
$\int_0^{\pi/2} \cos(x)\, e^{\sin(x)}\, dx$	1.71828182845905e+00	4.337e-19	$n=4,\ p=1024$
$\int_0^{500} \cos(x)\, e^{\sin(x)}\, dx$	-3.73603552314934e-01	1.897e-19	$n=4,\ p=1024000$

$\int_0^{\pi/2} \cos(x)\ dx$	1.00000000000000e+00	1.084e-19	$n=4,\ p=2048$
$\int_0^{\pi/2} \sqrt{1-2^{-1}\sin^2(x)}\,dx$	1.35064388104768e+00	0.000e+00	$n=4,\ p=64$
$\int_0^{2\pi} \left(e^{x/2}+\cos(4x)\right)dx$	4.42813852655585e+01	1.388e-17	$n=4,\ p=4096$
$\int_0^{2\pi} \left(x\,e^{-x}\cos(2x)\right)dx$	-1.22122604618968e-01	2.507e-19	$n=4,\ p=2048$

Замечание 5.4. Возможность представления (5.50) как непрерывного аналитического приближения первообразной подынтегральной функции является отличительной особенностью метода, обусловленной алгебраической формой полиномов с числовыми коэффициентами, приближающих подынтегральную функцию и интеграл. Известные методы аналогичную возможность непосредственно не предоставляют.

Как видно из табл. 5.7 приближение интеграла на основе полинома Лагранжа из соотношения (5.35) характеризуется высокой точностью, аналогичные значения погрешности получаются при приближении интегралов на основе полинома Ньютона из соотношения (5.38) [181]. При этом оба приближения согласно эксперименту отличаются стабильностью. Приближение на основе полинома Лагранжа из (5.35) положительно отличается по трудоемкости, в частности, от методов вычисления интеграла, представленных в [124] и от современных модификаций формул Ньютона-Котеса. Все рассмотренные варианты приближений интеграла являются сравнительно быстродействующими и обладают максимальным параллелизмом.

Программы, реализующие приближения, достаточно компактны, коэффициенты приближений в них являются хранимыми в разделах описания констант, отсюда сравнительное быстродействие их выполнения. Для запуска соответствующей программы, пользователю достаточно задать на ее входе вид подынтегральной функции, промежуток интегрирования, степень полинома и число подынтервалов в качестве параметров. Можно упростить

пользовательский интерфейс, если дополнить программу выбором наилучшего приближения подынтегральной функции. Программная реализация такого приближения выполняется в [178].

Ниже исследуется применимость кусочно-интерполяционного вычисления интеграла от быстро осциллирующих функций на основе полинома Ньютона.

Пример 5.1. На основе полинома Ньютона вычисляется кусочно-интерполяционное приближение коэффициентов тригонометрического ряда Фурье функции $f(x)=e^x$, определяемых формулами [197]:

$$\frac{1}{\pi}\int_{-\pi}^{\pi} e^x \cos(m\,x)\,dx = \frac{(-1)^m 2\,sh(\pi)}{\pi(1+m^2)},\ \frac{1}{\pi}\int_{-\pi}^{\pi} e^x \sin(m\,x)\,dx = \frac{(-1)^{m+1} 2\,m\,sh(\pi)}{\pi(1+m^2)}.$$

Точные решения, представленные в правых частях равенств использованы для оценки абсолютной погрешности приближения. Степень полинома $n=13$ на каждом подынтервале, длина подынтервала при $m=1$ – $2\pi/11$, $m=10$ – $2\pi/100$, $m=100$ – $2\pi/7000$.

Результат работы программы:

при $m=1$:

```
-3.67607791037498E+0000  0.00000000000000E+0000
-3.67607791037498E+0000  4.33680868994202E-0019
```

при $m=10$:

```
7.27936219876233E-0002  2.16840434497101E-0019
7.27936219876233E-0002  2.43945488809238E-0019
```

при $m=100$:

```
7.35142067868209E-0004  2.79679691271529E-0019
7.35142067868209E-0004  2.55539252274782E-0019
```

Таким образом, без внесения дополнительных изменений в алгоритм метода значения интегралов от сильно осциллирующих функций приближены с помощью кусочно-интерполяционного метода на основе полинома Ньютона с точностью 10^{-19}.

5.3.5. Сравнение модифицированной обработки данных с известными в случае моделей периодических процессов. Для наибольшей связности текста ниже кратко повторяются основные предположения и базовые шаги алгоритма построения кусочно-интерполяционного метода решения задачи Коши для ОДУ, описанного в главе 3. Рассматривается задача Коши

$$y' = f(x,y),\ y(x_0) = y_0, \tag{5.59}$$

в области $R: \{ a \le x \le b; \left| y - y_0 \right| \le B;\ B = \text{const} \}$, где функция $f(x,y)$ определена, непрерывно дифференцируема (в точках a – справа, b – слева) и удовлетворяет условию Липшица: $\left| f(x,y) - f(x,\tilde{y}) \right| \le L \left| y - \tilde{y} \right|$, $L = \text{const}$, $\forall\ (x,y), (x,\tilde{y}) \in R$. Предполагается, что в R решение задачи (5.59) существует и единственно. Для простоты a, b те же, что в (5.25), и выполнено такое же разбиение на подынтервалы $[a_i, b_i]$. Для интерполяции правой части (5.59) в $f(x,y)$ подставляется приближенное значение y, вначале $y \approx y_0$. Функция $f(x,y_0)$ приближается полиномами вида (5.29), (5.30) по изложенной выше схеме. При фиксированных n и k на отрезке $[a_i, b_i]$, $i = 0$, затем, аналогично, при $i = 1, 2, \ldots$, выполняется итерационное уточнение, которое состоит в следующем.

Пусть $P_{in}(t) = \sum_{\ell=0}^{n} D_{i\ell}\, t^{\ell}$, тогда $f(x,y_0) \approx P_{in}(t)$, $t = (x - a_i) h_i^{-1}$, h_i – шаг интерполяции на $[a_i, b_i]$. Первообразная $P_{(\text{int})i(n+1)}(x) = y_{0(i-1)} + h_i \int_0^{(x-a_i)h_i^{-1}} P_{in}(t)\, dt$ равная $y_{0(i-1)} + h_i \sum_{\ell=0}^{n} D_{i\ell} / (\ell+1) t^{\ell+1}$ принимается за приближение решения: $y(x) \approx P_{(\text{int})i(n+1)}(x)$, $x \in [a_i, b_i]$. Далее, полагается $f(x,y) \approx f(x, P_{(\text{int})i\,(n+1)}(x))$, и при том же n, на том же отрезке строится интерполяционный полином вида (5.30) для приближения полученной функции: $P_{in}^{(1)}(t) \approx f(x, P_{(\text{int})i(n+1)}(x))$, $t = (x - a_i) h_i^{-1}$. От этого полинома снова берется первообразная с тем же значением константы

$P^{(1)}_{(\mathrm{int})i\,(n+1)}(x) = y_{0(i-1)} + h_i \int\limits_0^{(x-a_i)h_i^{-1}} P^{(1)}_{in}(t)\,dt$, подставляется в правую часть, $f(x,y) \approx f(x, P^{(1)}_{(\mathrm{int})i\,(n+1)}(x))$, которая затем интерполируется аналогично: $P^{(2)}_{in}(t) \approx f(x, P^{(1)}_{(\mathrm{int})i\,(n+1)}(x))$. Фактически итерации $P^{(\ell)}_{in}(t) \approx f(x, P^{(\ell-1)}_{(\mathrm{int})\,i\,(n+1)}(x))$, $t = (x-a_i)h_i^{-1}$, $P^{(\ell)}_{(\mathrm{int})i\,(n+1)}(x) = y_{0(i-1)} + h_i \int\limits_0^{(x-a_i)h_i^{-1}} P^{(\ell)}_{in}(t)\,dt$, $\ell = 1,2,\ldots,$ продолжаются до заданной границы $\ell \le q = \mathrm{const}$, абстрактно их количество не ограничивается. Выше за значение $y_{0(i-1)}$ было принято $P^{(q)}_{(\mathrm{int})i\,(n+1)}(b_{i-1})$. По окончании итераций на $[a_i, b_i]$ выполняется переход к $[a_{i+1}, b_{i+1}]$, где за значение y_{0i} принимается $P^{(q)}_{(\mathrm{int})\,i\,(n+1)}(b_i)$. При оценках погрешности предполагается, что интерполяция выполняется полиномами (5.15 приведенными к виду

$$P_{in}(x) = \sum_{\ell=0}^{n} d_{i\ell} x^{\ell},\ x \in [a_i, b_i]. \tag{5.60}$$

Погрешность оценивается из (5.28), правая часть этой оценки обозначается $c_{ik} = c\,2^{-k\,(n+1)}\,h^{n+1}$, $h = (b-a)n^{-1}$. Вначале погрешность приближения решения с итерационным уточнением рассматривается только на отрезке $[a_i, b_i]$. Предполагается, что решение на этом отрезке соответствует начальным значениям $y_{0(i-1)}$, тогда решение $y(x)$ и приближение $P^{(\ell)}_{(\mathrm{int})i\,(n+1)}(x)$ имеют одинаковые начальные значения на $[a_i, b_i]$. Отсюда

$$y = y_{0(i-1)} + \int\limits_{a_i}^{x} f(x,y)\,dx,\ y(a_i) = y_{0(i-1)},\ P^{(\ell)}_{(\mathrm{int})\,i\,(n+1)}(x) = y_{0(i-1)} + \int\limits_{a_i}^{x} P_{in}(x)\,dx.$$

Сначала оценки выполняются в предположении, что для некоторой последовательности номеров ℓ

$$0 < c_{ik} \le \max_{[a_i, b_i]} \left| y(x) - P^{(\ell)}_{(\mathrm{int})\,i\,(n+1)}(x) \right|,\ \ell = 0,1,\ldots, \tag{5.61}$$

где $P^{(0)}_{(\text{int})\,i\,(n+1)}(x)=P_{(\text{int})\,i\,(n+1)}(x)$. С учетом одинаковых начальных значений абсолютная погрешность ℓ-й итерации примет вид $\left| y(x)-P^{(\ell)}_{(\text{int})\,i\,(n+1)}(x)\right| = \left|\int_{a_i}^{x}(f(x,y)-P^{(\ell)}_{in}(x))dx\right|$, отсюда

$$\left| y(x)-P^{(\ell)}_{(\text{int})\,i\,(n+1)}(x)\right| \le \int_{a_i}^{x}\left| f(x,y)-P^{(\ell)}_{in}(x)\right|dx\,,\ \ell=0,1,2,\ldots. \qquad (5.62)$$

По построению $P^{(\ell)}_{in}(x)\approx f(x,P^{(\ell-1)}_{(\text{int})\,i\,(n+1)}(x))$, погрешность интерполяции обозначается $\widetilde{c}_{ik}$, в этом обозначении $f(x,P^{(\ell-1)}_{(\text{int})\,i\,(n+1)}(x))=P^{(\ell)}_{in}(x)+\widetilde{c}_{ik}$. Подстановка в (5.62) влечет

$$\left| y(x)-P^{(\ell)}_{(\text{int})\,i\,(n+1)}(x)\right| \le \int_{a_i}^{x}\left| f(x,y)-(f(x,P^{(\ell-1)}_{(\text{int})\,i\,(n+1)}(x))-\widetilde{c}_{ik})\right|dx\ \forall x\in[a_i,b_i].$$

Согласно (5.28) $\left|\widetilde{c}_{ik}\right|\le c_{ik}$, поэтому

$$\left| y(x)-P^{(\ell)}_{(\text{int})\,i\,(n+1)}(x)\right| \le \int_{a_i}^{x}\left(\left| f(x,y)-f(x,P^{(\ell-1)}_{(\text{int})\,i\,(n+1)}(x))\right|+c_{ik}\right)dx\,. \qquad (5.63)$$

Отсюда, при условии, что для индекса $\ell-1$ верно (5.61), следует неравенство

$$\left| y(x)-P^{(\ell)}_{(\text{int})\,i\,(n+1)}(x)\right| \le$$
$$\le \int_{a_i}^{x}\left(\max_{[a_i,b_i]}\left| f(x,y)-f(x,P^{(\ell-1)}_{(\text{int})\,i\,(n+1)}(x))\right|+\max_{[a_i,b_i]}\left| y(x)-P^{(\ell-1)}_{(\text{int})\,i\,(n+1)}(x)\right|\right)dx\ \forall x\in[a_i,b_i].$$

С применением условия Липшица,

$$\left| y(x)-P^{(\ell)}_{(\text{int})\,i\,(n+1)}(x)\right| \le$$
$$\le \int_{a_i}^{x}\left(L\max_{[a_i,b_i]}\left| y(x)-P^{(\ell-1)}_{(\text{int})\,i\,(n+1)}(x)\right|+\max_{[a_i,b_i]}\left| y(x)-P^{(\ell-1)}_{(\text{int})\,i\,(n+1)}(x)\right|\right)dx\ \forall x\in[a_i,b_i], \qquad (5.64)$$

или,

$$\left| y(x) - P^{(\ell)}_{(\text{int})\, i\, (n+1)}(x) \right| \le N \int_{a_i}^{x} \max_{[a_i,\, b_i]} \left| y(x) - P^{(\ell-1)}_{(\text{int})\, i\, (n+1)}(x) \right| dx \;\forall\, x \in [a_i, b_i],\; N = L+1.$$

Очевидно,

$$\left| y(x) - P^{(\ell)}_{(\text{int})\, i\, (n+1)}(x) \right| \le N \max_{[a_i,\, b_i]} \left| y(x) - P^{(\ell-1)}_{(\text{int})\, i\, (n+1)}(x) \right| \int_{a_i}^{x} dx,\; \ell = 0,1,2,\ldots,$$

или,

$$\left| y(x) - P^{(\ell)}_{(\text{int})\, i\, (n+1)}(x) \right| \le$$
$$\le N \max_{[a_i,\, b_i]} \left| y(x) - P^{(\ell-1)}_{(\text{int})\, i\, (n+1)}(x) \right| (x - a_i) \;\forall\, x \in [a_i, b_i],\; \ell = 0,1,2,\ldots.$$

Ввиду произвольности $x \in [a_i, b_i]$ в обеих частях неравенства можно перейти к максимуму:

$$\begin{aligned} &\max_{[a_i,\, b_i]} \left| y(x) - P^{(\ell)}_{(\text{int})\, i\, (n+1)}(x) \right| \le \\ &\le N \max_{[a_i,\, b_i]} \left| y(x) - P^{(\ell-1)}_{(\text{int})\, i\, (n+1)}(x) \right| (b_i - a_i),\; \ell = 0,1,2,\ldots \end{aligned} \tag{5.65}$$

Пусть левая часть неравенства (5.65) обозначена $\varepsilon_{i\ell}$, в этом обозначении

$$\varepsilon_{i\ell} \le N\, \varepsilon_{i\,(\ell-1)} (b_i - a_i),\; \ell = 0,1,2,\ldots. \tag{5.66}$$

В (5.66) $\varepsilon_{i\,0} = \max\limits_{[a_i,\, b_i]} \left| y(x) - P_{(\text{int})\, i\, (n+1)}(x) \right|$, и $\varepsilon_{i1} \le N \max\limits_{[a_i,\, b_i]} \left| y(x) - P_{(\text{int})\, i\, (n+1)}(x) \right| (b_i - a_i)$.

Согласно построению $f(x, y_{0(i-1)}) = P_{in}(x) + \widetilde{c}_{ik}$, поэтому, с учетом (5.62) при $\ell = 0$ и проделанных выше преобразований,

$$\left| y(x) - P_{(\text{int}) i (n+1)}(x) \right| \le \int_{a_i}^{b_i} \left(\left| f(x, y_{0(i-1)}) - P_{in}(x) \right| + c_{ik} \right) dx \;\forall\, x \in [a_i, b_i].$$

Отсюда $\max\limits_{[a_i,\, b_i]} \left| y(x) - P_{(\text{int}) i (n+1)}(x) \right| \le 2 c_{ik} \int\limits_{a_i}^{b_i} dx$, где $c_{ik} = c\; 2^{-k\,(n+1)}\, h^{n+1}$, $h = (b-a) n^{-1}$,

или,

$$\varepsilon_{i\,0} \le (b_i - a_i)\times\alpha_{kn}\ ,\ \alpha_{kn} = 2c_{ik}. \tag{5.67}$$

Из (5.66), (5.67) $\varepsilon_{i1} \le N\times(b_i - a_i)^2\times\alpha_{kn}$, поэтому $\varepsilon_{i2} \le N^2(b_i - a_i)^3\alpha_{kn}$. По индукции

$$\varepsilon_{i\ell} \le \alpha_{kn} N^{-1}((b_i - a_i)N)^{\ell+1}. \tag{5.68}$$

Неравенство (5.68) является следствием (5.65) и верно для тех последовательных $\ell = 0,1,2,\ldots$, для которых не нарушено (5.61). Не умаляя общности, можно предположить, что $(b_i - a_i)N = 2^{-k}(b-a)N < 1$, тогда в (5.68) $\varepsilon_{i\ell} \to 0$, $\ell \to \infty$. Поэтому неравенство (5.61) при некотором ℓ необходимо окажется нарушенным, нарушение означает, что

$$0 < \max_{[a_i,\,b_i]} \left| y(x) - P^{(\ell)}_{(\text{int})\,i\,(n+1)}(x) \right| < c_{ik}\ ,\ \ell = \ell_0 + 1. \tag{5.69}$$

Пусть $\ell = \ell_0 + 1$ будет первым элементом последовательности $\ell = 0,1,2,\ldots$, при котором неравенство (5.61) будет нарушено и выполнится (5.69). В этом случае для номера $\ell = \ell_0 + 1$ еще сохраняются (5.65) и (5.68). Из (5.63) с применением условия Липшица следует

$$\begin{aligned} &\left| y(x) - P^{(\ell_0+2)}_{(\text{int})\,i\,(n+1)}(x) \right| \le \\ &\le \int_{a_i}^{x} (L \max_{[a_i,\,b_i]} \left| y(x) - P^{(\ell_0+1)}_{(\text{int})\,i\,(n+1)}(x) \right| + c_{ik})\,dx\ \forall\, x \in [a_i, b_i]. \end{aligned} \tag{5.70}$$

Отсюда, с учетом выполнения (5.68) для $\ell = \ell_0 + 1$ и того, что

$$\max_{[a_i,\,b_i]} \left| y(x) - P^{(\ell_0+1)}_{(\text{int})\,i\,(n+1)}(x) \right| < c_{ik} < \max_{[a_i,\,b_i]} \left| y(x) - P^{(\ell_0)}_{(\text{int})\,i\,(n+1)}(x) \right|,$$

подынтегральное выражение в (5.70) можно заменить на соответственное выражение из (5.64):

$$\left| y(x) - P^{(\ell_0+2)}_{(\text{int})\, i\, (n+1)}(x) \right| \le$$

$$\le \int_{a_i}^{x} (L \max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0)}_{(\text{int})\, i\, (n+1)}(x) \right| + \max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0)}_{(\text{int})\, i\, (n+1)}(x) \right|) dx \ \forall x \in [a_i, b_i]. \tag{5.71}$$

Но для правой части (5.71), после перехода к максимуму в обеих частях неравенства, сохраняется оценка (5.68), при которой левая часть неравенства (5.68) уже нарушает (5.70) и не превосходит c_{ik}. Поэтому и левая часть (5.71) не превзойдет c_{ik}:

$$\max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0+2)}_{(\text{int})\, i\, (n+1)}(x) \right| < c_{ik} < \max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0)}_{(\text{int})\, i\, (n+1)}(x) \right|. \tag{5.72}$$

Аналогично,

$$\left| y(x) - P^{(\ell_0+3)}_{(\text{int})\, i\, (n+1)}(x) \right| \le \int_{a_i}^{x} (L \max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0+2)}_{(\text{int})\, i\, (n+1)}(x) \right| + c_{ik}) dx \ \forall x \in [a_i, b_i], \tag{5.73}$$

и

$$\max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0+2)}_{(\text{int})\, i\, (n+1)}(x) \right| < c_{ik} < \max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0)}_{(\text{int})\, i\, (n+1)}(x) \right|,$$

поэтому подынтегральное выражение в правой части (5.73) можно заменить на выражение из (5.64), что повлечет совпадение с правой частью (5.71):

$$\left| y(x) - P^{(\ell_0+3)}_{(\text{int})\, i\, (n+1)}(x) \right| \le$$

$$\le \int_{a_i}^{x} (L \max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0)}_{(\text{int})\, i\, (n+1)}(x) \right| + \max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0)}_{(\text{int})\, i\, (n+1)}(x) \right|) dx \ \forall x \in [a_i, b_i].$$

Повторение предыдущих рассуждений влечет

$$\max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0+3)}_{(\text{int})\, i\, (n+1)}(x) \right| < c_{ik}. \tag{5.74}$$

В силу очевидной индукции неравенства (5.72) и (5.74) перейдут в неравенство

$$\max_{[a_i, b_i]} \left| y(x) - P^{(\ell_0+r)}_{(\text{int})\, i\, (n+1)}(x) \right| < c_{ik}, \ 1 \le r.$$

Имеет место

Лемма 5.2. Пусть в рассматриваемых условиях, в числе которых условия выполнения (5.26), (5.28) применительно к правой части (5.59), а также (5.61) и предположение $2^{-k}(b-a)N<1,$ выполняется кусочная интерполяция с итерационным уточнением решения задачи (5.59). Тогда на произвольном отрезке из (5.25) найдется номер r_0, такой, что итерационное уточнение будет удовлетворять неравенству

$$\max_{[a_i,\,b_i]}\left|\ y(x)-P^{(r)}_{(\mathrm{int})\ i\ (n+1)}(x)\right|<c_{ik}\ \forall\ r\ge r_0. \tag{5.75}$$

Для номеров из (5.75) сохраняется (5.63), откуда с применением условия Липшица

$$\left|\ y(x)-P^{(r+1)}_{(\mathrm{int})\ i\ (n+1)}(x)\right|\le\int_{a_i}^{x}\left(L\max_{[a_i,\,b_i]}\left|\ y(x)-P^{(r)}_{(\mathrm{int})\ i\ (n+1)}(x)\right|+c_{ik}\right)dx\ \forall\, x\in[a_i,\,b_i].$$

Из (5.75)

$$\left|\ y(x)-P^{(r+1)}_{(\mathrm{int})\ i\ (n+1)}(x)\right|\le N\int_{a_i}^{x}c_{ik}\,dx\ \forall\, x\in[a_i,\,b_i],$$

или, $\left|\ y(x)-P^{(r+1)}_{(\mathrm{int})\ i\ (n+1)}(x)\right|\le N\,c_{ik}(x-a_i)\ \forall\, x\in[a_i,\,b_i]$. Следовательно,

$$\max_{[a_i,\,b_i]}\left|\ y(x)-P^{(r+1)}_{(\mathrm{int})\ i\ (n+1)}(x)\right|\le N\,c_{ik}(b_i-a_i)\ \ \forall\ r\ge r_0,\ \ N=L+1,$$

или,

$$\max_{[a_i,\,b_i]}\left|\ y(x)-P^{(r+1)}_{(\mathrm{int})\ i\ (n+1)}(x)\right|\le N\,c_{ik}(b-a)2^{-k}\ \ \forall\ r\ge r_0,\ \ N=L+1. \tag{5.76}$$

Теорема 5.3. В условиях леммы 5.2 абсолютная погрешность решения задачи (5.59) оценивается из (5.76). Согласно (5.76) погрешность на подынтервале за счет итерационного уточнения уменьшается пропорционально $(b-a)2^{-k}$.

Следствие 5.1. В тех же условиях скорость сходимости к (5.76) определяется геометрической прогрессией из (5.68).

Следствие 5.2. В условиях теоремы 5.3 абсолютная погрешность решения задачи (5.59) на всем отрезке (5.25) оценивается из соотношения $\sum_{i=0}^{2^k-1} \max_{[a_i,\,b_i]} \left| y(x) - P^{(r+1)}_{(\mathrm{int})\,i\,(n+1)}(x) \right| \le \sum_{i=0}^{2^k-1} N c_{ik} (b-a) 2^{-k}$, или

$$\sum_{i=0}^{2^k-1} \max_{[a_i,\,b_i]} \left| y(x) - P^{(r+1)}_{(\mathrm{int})\,i\,(n+1)}(x) \right| \le N(b-a) c_{ik} . \qquad (5.77)$$

Согласно (5.77) итерационное уточнение позволяет, с точностью до постоянного множителя $N(b-a)$, приближать решение на всем отрезке (5.25) с абсолютной погрешностью кусочной интерполяции на одном подынтервале из (5.25). Подстановка в (5.77) c_{ik} из (5.28) влечет

$$\sum_{i=0}^{2^k-1} \max_{[a_i,\,b_i]} \left| y(x) - P^{(r+1)}_{(\mathrm{int})\,i\,(n+1)}(x) \right| \le N(b-a) c\, 2^{-k\,(n+1)} h^{n+1} ,$$

или,

$$\sum_{i=0}^{2^k-1} \max_{[a_i,\,b_i]} \left| y(x) - P^{(r+1)}_{(\mathrm{int})\,i\,(n+1)}(x) \right| \le N(b-a)((b-a)/n)^{n+1} c\, 2^{-k\,(n+1)} . \qquad (5.78)$$

Пусть задано $\forall \varepsilon > 0$. В (5.78) можно указать такое $k_0 = k_0(\varepsilon)$, что правая часть не превзойдет ε при $\forall k > k_0$. Именно, $k_0 = (n+1)^{-1} \log_2 (N(b-a)((b-a)/n)^{n+1} c\, \varepsilon^{-1})$.

Теорема 5.4. В условиях леммы 5.2 абсолютная погрешность решения задачи (5.59) на всем отрезке решения из (5.25) оценивается из (5.78). При этом $\forall \varepsilon > 0$ верно соотношение

$$\sum_{i=0}^{2^k-1} \max_{[a_i,\,b_i]} \left| y(x) - P^{(r+1)}_{(\mathrm{int})\,i\,(n+1)}(x) \right| \le \varepsilon$$
$$\forall k > (n+1)^{-1} \log_2 (N(b-a)((b-a)/n)^{n+1} c\, \varepsilon^{-1}),\ \forall x \in [a, b],$$

означающее равномерную сходимость метода, если $k \to \infty$.

Теоремы 5.3 и 5.4 переносятся на случай интерполяционного полинома Ньютона в форме (5.31) или (5.32) с точностью до замены обозначений $P_{in}^{(\ell)}(t)$ на $\Psi_{in}^{(\ell)}(x)$, и $P_{(\text{int})i(n+1)}^{(\ell)}(x)$ на $\Psi_{(\text{int})i(n+1)}^{(\ell)}(x)$.

Теоремы 5.3, 5.4 и следствия 5.1, 5.2 дают формальную оценку погрешности в абстрактных условиях, включающих существование $n+1$ производной у функции правой части (5.59). Однако сама по себе интерполяция возможна в самых широких условиях, поэтому метод всегда можно применять в условиях существования и единственности. На практике многое определяется не только малостью подынтервала в (5.78), но видом правой части (5.59), устойчивостью решения в смысле Ляпунова, жесткостью или нежесткостью класса задач. Тем не менее, во всех таких экспериментально рассмотренных случаях предложенный метод обладает меньшей погрешностью, чем известные методы, отличаясь фиксированной границей накопления погрешности на большом промежутке решения.

Исследуются границы погрешности и трудоемкость кусочно-интерполяционного приближения задачи Коши для систем ОДУ при фиксированных значениях параметров метода. Ниже представлены результаты на основе интерполяции по Ньютону, так как он дает большую чем полином Лагранжа свободу в вариации степени полинома в реализации кусочно-интерполяционного метода.

Пример 5.2. Задача $y' = \cos(x+y)$, $y(0)=0$, имеет решение $y=-x+2arctg(x)$. Абсолютная погрешность приближения на $[0, 512]$ разностными и предложенным кусочно-интерполяционным (*FPI*) методами дана в табл. 5.9 с числом обращений (*fc*) к правой части.

Таблица 5.9

Погрешность и число обращений к правой части при решении задачи примера 5.2

x	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*FPI*
	$h=1.024\times10^{-3}$	$h=1.024\times10^{-2}$	$h=1.024\times10^{-2}$	$h=2.3\times10^{-2}$
	$fc=2000000$	$fc=350000$	$fc=650000$	$fc=183344$
5.12	1.315**e-15**	2.665**e-16**	1.518**e-18**	1.084**e-18**
10.24	3.422**e-16**	6.765**e-17**	7.373**e-18**	0.000**e+00**
…	…	…	…	…
256.00	3.608**e-16**	2.498**e-16**	7.078**e-16**	4.163**e-17**
261.12	8.327**e-16**	4.441**e-16**	8.882**e-16**	5.551**e-17**
…	…	…	…	…
506.88	3.053**e-16**	1.749**e-15**	4.524**e-15**	2.776**e-17**
512.00	8.049**e-16**	2.137**e-15**	3.747**e-15**	5.551**e-17**

Граница погрешности 10^{-15} соответствует методам 4-го (*Runge-Kutta_4*), 6-го (*Butcher_6*) и 8-го (*Dormand-Prince_8*) порядков. Величины шагов для каждого разностного метода выбраны с целью наименьшей погрешности. Методу Бутчера соответствует количество обращений к функции правой части $fc=350000$. Кусочно-интерполяционное решение с параметрами $b_i-a_i=0.345$, $n=15$, $\ell=13$, характеризуется порядком погрешности 10^{-17} при $fc=183344$. Таким образом, сравнительно высокая точность кусочно-интерполяционного приближения достигнута при меньшем количестве обращений к функции правой части. В сравнении с варьируемым кусочно-интерполяционным методом, для данной задачи сохранен порядок абсолютной погрешности приближения (см. глава 3, табл. 3.2) при значительном снижении трудоемкости. Программа предложенного решения данной задачи имеет вид:

```
program RD_FPI_example_1; {$APPTYPE CONSOLE} uses SysUtils, Math;
var koutput,kiter:integer; Npol:byte; fc:longint;
    Anach,Bkonech,velint,ynach,hpd_:extended;
function f(x,y: extended):extended; begin f:=cos(x+y);fc:=fc+1; end;
function fun(x:extended):extended; begin fun:=-x+2*arctan(x) end;
procedure RD(y_nach,A_nach,B_konech,vel_int:extended;n:byte;
hpd:extended;k_iter,k_output:integer);
const nn=20; type matr=array[0..nn,0..nn] of extended;
 vect=array[0..nn] of extended; matrC=array[-4..nn+1] of extended;
 matrAll=array[0..33000] of matrC; matrC_=^matrAll;
var d:matr; a0,b0,x,y0:extended; i:integer; Ck1:matrC_; pod:longint ;
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr;
  begin e[1,1]:=1; e[1,0]:=0; for k:=2 to n do
begin e[k,0]:=-e[k-1,0]*(k-1);  for i:=1 to k-1 do
 e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1] end;
for k:=1 to n do for i:=0 to k do d[i,k]:=e[k,i] end;
procedure Konech_Raznoct(fy:vect; n:byte; var dy:matr); var i,j:byte;
  begin for j:=0 to n-1 do dy[1,j]:=fy[j+1]-fy[j];
```

```
   for i:=2 to n do for j:=0 to n-i do
   dy[i,j]:=dy[i-1,j+1]-dy[i-1,j] end;
procedure Newton(U:Vect; n:byte; var Mcoef:matrC);
 var dy:matr; b:vect; p,s:extended; j,i:byte;
 begin Konech_Raznoct(U,n,dy); p:=1; for j:=1 to n do begin p:=p*j;
 b[j]:=dy[j,0]/p; end;  Mcoef[0]:=U[0]; for i:=1 to n do begin s:=0;
for j:=i to n do s:=s+d[i,j]*b[j];  Mcoef[i]:=s; end end;
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
 begin  t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
  for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
procedure Polynomial(x:vect; h,y0:extended; n,K_it:integer; var C:matrC);
var i,iter:integer; fy,y,ytemp:vect; A:matrC; sum:extended;
begin  for i:=0 to n do y[i]:=y0; for iter:=1 to K_it do
 begin for i:=1 to n do ytemp[i]:=y[i];
 for i:=0 to n do fy[i]:=f(x[i],y[i]);  Newton(fy,n,A); C[0]:=y[0];
C[-1]:=x[0]; C[-2]:=h; C[-3]:=n+1;C[-4]:=n*h;
for i:=1 to n+1 do C[i]:=A[i-1]*h/i;
for i:=1 to n do y[i]:=Gorner(C,x[i]); sum:=0;
for i:=1 to n do sum:=sum+abs(y[i]-ytemp[i]);
  if (sum<1e-40) then break; if (sum>=1e50) then break; end end;
procedure Subinterval(hpd:extended;n,K_it:integer; a0,b0,Ynach:extended;
var Ck:matrC_); var a00,b00,y0,h:extended; m,pod:longint; x:vect; j:byte;
begin a00:=a0;b00:=a00+hpd;  y0:=Ynach;x[0]:=a0;m:=0; pod:=0;
 while a00<=b0-hpd/10 do begin h:=(b00-a00)/n;
for j:=1 to n do begin inc(m); x[j]:=a0+m*h end;
Polynomial(x,h,y0,n,K_it,Ck^[pod]); y0:=gorner(Ck^[pod],x[n]);
 x[0]:=x[n];inc(pod); a00:=a00+hpd; b00:=a00+hpd end end;
begin fc:=0; New(Ck1);  Viet(nn,d); writeln('x':4,'y':15,'Pogr':25); writeln;
 a0:=A_nach; b0:=a0+vel_int;  y0:=y_nach;
 while a0 <= B_konech-vel_int/2 do
  begin Subinterval(hpd,n,k_iter,a0,b0,y0,Ck1); writeln;
  for i:=1 to k_output-1 do begin x:= a0+i*Vel_int/k_output;
  pod:=trunc((x-a0)/Ck1^[0,-4]); if x>Bkonech then break;
writeln(x:7:3,'  ',Gorner(Ck1^[pod],x),'  ',abs(Gorner(Ck1^[pod],x)-fun(x)));end;
if abs(frac((b0-a0)/Ck1^[0,-4]))<1e-18 then pod:=trunc((b0-a0)/Ck1^[0,-4])-1
else pod:=trunc((b0-a0)/Ck1^[0,-4])-1; y0:=Gorner(Ck1^[pod],b0);
 writeln(b0:7:3,' ', y0, ' ',abs(y0-fun(b0)));
 a0:=a0+vel_int; b0:=a0+vel_int; end; Dispose(Ck1); end;
begin Anach:=0; Bkonech:=512; ynach:=0; velint:=512;
Npol:=15; hpd_:=0.345; kiter:=13; koutput:=100;
RD(ynach,Anach,Bkonech,velint,Npol,hpd_,kiter,koutput);writeln(fc); readln;end.
```

Пример 5.3. Система $y_1' = x + 2y_1/x - \sqrt{y_2}$, $y_2' = 2\sqrt{y_2}$, при $y_1(1) = 2$, $y_2(1) = 4$ имеет неустойчивое в смысле Ляпунова решение $y_1 = x(1+x)$, $y_2 = (1+x)^2$. Канонические нормы вектора абсолютных погрешностей компонентов системы на отрезке $[1, 513]$ наряду с *fc* представлены в табл. 5.10.

Таблица 5.10

Погрешность и число обращений к правой части при решении задачи примера 5.3

x	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*FPI*
	$h = 1.024\times10^{-5}$	$h = 1.0\times10^{-4}$	$h = 1.0\times10^{-3}$	$h \approx 8.3\times10^{-2}$
	$fc = 2.0\times10^{8}$	$fc = 35840000$	$fc = 6656000$	$fc = 56028$
6.12	2.082**e-17**	3.539**e-16**	8.674**e-17**	0.000e+**00**
11.24	1.110**e-16**	1.388**e-16**	3.608**e-16**	0.000e+**00**
…	…	…	…	…
257.00	5.684**e-14**	2.842**e-14**	6.821**e-13**	0.000e+**00**
262.12	5.684**e-14**	4.974**e-14**	3.837**e-13**	0.000e+**00**
…	…	…	…	…
507.88	1.563**e-13**	1.705**e-13**	2.416**e-13**	0.000e+**00**
513.00	1.421**e-13**	4.832**e-13**	4.263**e-13**	0.000e+**00**

При одинаковом порядке погрешности среди разностных методов наименьшим значением $fc = 6656000$ характеризуется метод Дормана-Принса 8-го порядка. Кусочно-интерполяционное приближение с параметрами: $b_i - a_i = 0.25$, $n = 3$, $\ell = 20$, в большинстве проверочных точек дает значения «нулевых» погрешностей в формате вывода данных (*extended*), при увеличении числа проверочных точек встречаются значения 10^{-18} – 10^{-16}. При этом $fc = 56028$. Таким образом, относительно высокая точность кусочно-интерпояционного приближения решения, неустойчивого к возмущениям начальных условий, сохраняется и при исключении автоматического выбора параметров (см. табл. 5.10, табл. 3.3), что значительно снижает трудоемкость метода.

Пример 5.4. В [35] на серии тестовых задач представлено сравнение вычислительных качеств наиболее эффективных методов высокоточного решения нежестких задач Коши. Типичные результаты дает тестовая задача двух тел:

$$\begin{aligned} &y_1' = y_3,\ y_2' = y_4,\ y_3' = -y_1(y_1^2 + y_2^2)^{-3/2},\ y_4' = -y_2(y_1^2 + y_2^2)^{-3/2}, \\ &y_1(0) = 0.5,\ y_2(0) = 0,\ y_3(0) = 0,\ y_4(0) = \sqrt{3}. \end{aligned} \tag{5.79}$$

Канонические нормы вектора абсолютных погрешностей компонентов системы на отрезке $[0, 6\pi]$ наряду с fc представлены в табл. 5.11.

Таблица 5.11

Погрешность и число обращений к правой части при решении задачи (5.79)

x	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*FPI*
	$h = 2\pi \times 10^{-5}$	$h = 2\pi \times 10^{-4}$	$h = 2\pi \times 10^{-4}$	$h \approx 2\pi \times 5 \times 10^{-4}$
	$fc = 1.2 \times 10^{6}$	$fc = 2.1 \times 10^{5}$	$fc = 3.9 \times 10^{5}$	$fc = 68409$
2π	1.018**e-15**	2.979**e-16**	7.045**e-17**	3.946**e-17**
4π	1.866**e-15**	1.738**e-15**	2.160**e-16**	5.482**e-18**
6π	4.435**e-15**	3.295**e-15**	2.632**e-16**	5.094**e-17**

Наименьшей границы погрешности решения задачи (5.79), порядка 10^{-16} на отрезке $[0, 6\pi]$, среди разностных методов достигает метод Дормана-Принса 8-го порядка при $fc = 3.9 \times 10^{5}$. При этом кусочно-интерполяционное приближение характеризуется границей погрешности порядка 10^{-17} при $fc = 68409$ (параметры метода: $b_i - a_i = 2\pi / 256$, $n = 8$, $\ell = 20$). Как отмечалось выше, п. 5.2.3, для решения астрономических задач с повышенной точностью разработан специализированный неявный метод переменного порядка, получивший название «интегратор Гаусса-Эверхарта» [16]. Решение задачи (5.79) на том же отрезке на основе интегратора Гаусса-Эверхарта 19-го порядка, как и на основе метода Дормана-Принса, характеризуетсяграницей погрешности порядка 10^{-16}, при этом специализированный механизм выбора величины шага интегрирования, реализованный в программе Гаусса-Эверхарта с учетом специфики плоской задачи двух тел, позволяет снизить количество обращений к правой части до значения $fc = 14723$.

Таким образом, на примере решения модельной задачи небесной механики показана возможность повышения точности численного моделирования на один десятичный порядок с трудоемкостью сравнимой с трудоемкостью специализированных методов.

Относительная универсальность кусочно-интерполяционного метода, высокая точность и непрерывный характер приближения позволяют применить метод для построения высокоточных приближений специальных функций. Ниже показана возможность построения с помощью кусочно-

интерполяционного метода высокоточного непрерывного приближения функции Бесселя.

Пример 5.5. Дифференциальное уравнение Бесселя может быть представлено в виде системы

$$y_1' = y_2,\ \ y_2' = -(x\,y_2 + (x^2 - \alpha^2)y_1)/x^2. \tag{5.80}$$

При $\alpha = 0.5$ согласно определению функции Бесселя [198] система (5.80) имеет аналитически представимое решение: $y_1(x) = J_{1/2}(x) = \sqrt{\dfrac{2}{\pi x}}\sin(x)$. Абсолютная погрешность приближения $y_1(x)$ на отрезке $[1, 500]$ при $y_1(1) = \sqrt{2/\pi}\sin(1)$, $y_2(1) = \sqrt{2/\pi}\cos(1) - \sqrt{1/2\pi}\sin(1)$, представлена в табл. 5.12 с числом обращений к правой части.

Таблица 5.12

Абсолютная погрешность приближения функции Бесселя $J_{1/2}(x)$, полученного на основе решения системы (5.80) при $y_1(1) = \sqrt{2/\pi}\sin(1)$, $y_2(1) = \sqrt{2/\pi}\cos(1) - \sqrt{1/2\pi}\sin(1)$

x	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*FPI*
	$h = 1.0\times10^{-5}$	$h = 1.0\times10^{-3}$	$h = 1.0\times10^{-2}$	$h \approx 4.9\times10^{-3}$
	$fc = 1.996\times10^{8}$	$fc \approx 3.493\times10^{6}$	$fc = 6.487\times10^{5}$	$fc = 1.152\times10^{6}$
1.2	3.415**e-18**	1.572**e-18**	1.626**e-19**	5.421**e-20**
5.4	8.863**e-18**	6.505**e-19**	8.403**e-19**	5.421**e-20**
…	…	…	…	…
250.2	3.900**e-18**	1.931**e-18**	3.375**e-18**	3.388**e-21**
255.4	1.694**e-18**	1.403**e-18**	5.330**e-18**	2.711**e-20**
…	…	…	…	…
490.2	4.815**e-18**	5.044**e-19**	1.212**e-17**	3.642**e-20**
495.4	1.120**e-18**	1.875**e-18**	6.688**e-18**	1.694**e-20**
500.0	4.662**e-18**	2.574**e-19**	1.033**e-17**	5.421**e-20**

Таким образом, функция Бесселя вычислена с точностью порядка 10^{-20}, что превосходит точность вычисления известных методов (см. табл. 5.12) на два десятичных порядка. При границе погрешности порядка 10^{-18} среди разностных методов наименьшим значением $fc \approx 3.493\times10^{6}$ характеризуется метод Бутчера. При этом кусочно-интерполяционное приближение

характеризуется границей погрешности порядка 10^{-20} при $fc = 1.152 \times 10^6$ (параметры метода: $b_i - a_i = 10/256$, $n = 8$, $\ell = 10$).

Замечание 5.5. Так как функция Бесселя является специальной функцией и ее можно считать стандартной повторяющейся функцией на одном и том же интервале, то все коэффициенты интерполяционных полиномов могут быть рассчитаны априори. В этом случае временная сложность предложенного алгоритма составит $n \times (t_y + t_c)$, в данном в примере 5.5 случае $8 \times (t_y + t_c)$, что в принципе не достижимо другими разностными методами, где обращение к правой части (5.79) приходится выполнять в точности на каждом шаге.

Кроме того, следует отметить, что снижение шага разностного метода с целью плотной выдачи результатов приближения приводит к снижению точности приближения вследствие накопления вычислительной погрешности. Вместе с тем, полученное кусочно-интерполяционное приближение функции Бесселя является непрерывным и непрерывно дифференцируемым на всем отрезке интегрирования (см. глава 3, замечание 3.3).

На основе кусочно-интерполяционного метода возможно аналогичное построение приближения и других специальных функций, представимых дифференциальными уравнениями, с сохранением высокой точности приближения и малой временной сложности алгоритма его вычисления.

Многие специальные и элементарные функции при определённых значениях параметров и преобразовании независимого аргумента могут быть получены из гипергеометрической функции $y = F(a,b,c,z)$, удовлетворяющей дифференциальному уравнению Гаусса [196]

$$z(1-z)\frac{d^2y}{dz^2} + (c - (a+b+1)z)\frac{dy}{dz} - aby = 0. \qquad (5.81)$$

Пример 5.6. Рассматривается гипергеометрическая функция $F(1, 1, 2, -x)$, удовлетворяющая согласно (5.81) уравнению

$$\frac{d^2y}{dx^2} = \frac{(2+3x)y_2 + y_1}{-x(1+x)}. \qquad (5.82)$$

Через функцию $F(1, 1, 2, -x)$ выражается суперпозиция элементарных функций $x^{-1}\ln(1+x)$. Абсолютная погрешность приближения $F(1, 1, 2, -x)$, полученного на основе решения уравнения (5.82) при $y(1)=\ln(2)$, $y'(1)=0.5-\ln(2)$ на отрезке $[1, 500]$, наряду с fc представлены в табл. 5.13.

Таблица 5.13

Абсолютная погрешность приближения гипергеометрической функции $F(1, 1, 2, -x)$, полученного на основе решения (5.82) при $y(1)=\ln(2)$, $y'(1)=0.5-\ln(2)$

x	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*FPI*
	$h=1.0\times10^{-4}$	$h=1.0\times10^{-3}$	$h=1.0\times10^{-2}$	$h\approx4.3\times10^{-3}$
	$fc=1.996\times10^{7}$	$fc\approx3.493\times10^{6}$	$fc=6.487\times10^{5}$	$fc=1.28\times10^{6}$
1.2	2.168**e-19**	1.084**e-19**	0.000**e+00**	0.000**e+00**
5.4	1.328**e-18**	3.632**e-18**	2.982**e-19**	5.421**e-20**
…	…	…	…	…
250.2	3.727**e-20**	1.403**e-18**	8.470**e-20**	1.016**e-20**
255.4	1.135**e-19**	1.213**e-18**	1.287**e-19**	0.000**e+00**
…	…	…	…	…
495.4	2.329**e-19**	9.715**e-19**	8.894**e-20**	3.134**e-20**
500.0	1.635**e-19**	1.028**e-18**	8.301**e-20**	2.965**e-20**

Таким образом, гипергеометрическая функция вычислена с точностью порядка 10^{-20}, что превосходит точность вычисления известных методов (см. табл. 5.13) на один десятичный порядок. При границе погрешности порядка 10^{-19} среди разностных методов наименьшим значением $fc=6.487\times10^{5}$ характеризуется метод Дормана-Принса. Граница погрешности кусочно-интерполяционного приближения порядка 10^{-20} достигается при $fc=1.28\times10^{6}$ (параметры метода: $b_i-a_i=10/256$, $n=9$, $\ell=10$). Вместе с тем, согласно замечанию 5.5 после предварительного рассчета коэффициентов интерполяционных полиномов, временная сложность предложенного алгоритма для повторного вычисления функции составит $9\times(t_y+t_c)$.

Таким образом, на основе численного эксперимента показано, что в случае кусочно-интерполяционной обработки данных для дифференциальной модели эвристический подбор параметров n и k позволяет получить приближение данных с существенно меньшей трудоемкостью, чем при автоматическом выборе этих параметров (см. глава 3), причем без потери

точности приближения. При этом в случае вычисления специальной или стандартной функции, в частности, функции Бесселя и гипергеометрической функции, предложенный метод позволяет достигать временной сложности, измеряемой числом шагов схемы Горнера, то есть $T = n \times (t_y + t_c)$, что в принципе не достижимо с помощью известных разностных методов. Данные разностные методы в рассматриваемом применении продолжат сохранять количество обращений к правой части, указанное в табл. 5.12, табл. 5.13.

Таким образом, в главе представлен комплекс программ обработки данных в ИВС и выполнено моделирование периодических процессов из различных предметных областей, включая химические и биохимические реакции, электрическое равновесие автогенератора с внутренней обратной связью, движение КА с управлением при помощи «малой тяги», возмущенное периодическое движение КА. Для повышения качества обработки данных и результатов моделирования построена библиотека стандартных программ быстродействующего высокоточного вычисления функций и интегралов. Вычисление специальных и стандартных функций выполняется с точностью порядка 10^{-20} – 10^{-19} при временной сложности $O(1)$. На основе хранимых коэффициентов любая функция библиотеки может параллельно воспроизводиться на произвольном множестве точек фиксированной области. На аналогичной основе построено приближение производных с точностью до 10^{-16}. Представлены программы вычисления интеграла и результаты эксперимента, согласно которым на временном отрезке длины 500 достигается граница погрешности порядка 10^{-20}. На промежутках стандартной длины достигается нулевая граница погрешности вычисления интеграла. Обоснована сходимость метода, даны оценки скорости сходимости, представлены таблицы коэффициентов с целью стандартизации программ. В целом, результаты обработки данных отличаются гладкостью приближений в допустимых границах трудоемкости, что позволяет уточнить физико-химические параметры автоколебаний и повысить качество моделирования, применять метод для управления движением КА в режиме реального времени.

5.4. Выводы

1. Предложен метод создания библиотеки стандартных программ в ИВС на основе кусочно-интерполяционной обработки данных. Разновидность кусочной интерполяции позволяет приближать функции с точностью до 10^{-20} на произвольном временном отрезке, при этом стандартные и специальные функции приближаются за время единичного порядка, что положительно отличает метод от известных. Реализованы варианты хранения коэффициентов в разделе констант программы и в типизированном файле, дан алгоритм считывания. Вычисления функции взаимно независимы по значениям аргумента, что влечет параллелизм метода. На основе хранимых коэффициентов любая функция библиотеки может параллельно воспроизводиться на произвольном множестве точек фиксированной области. На аналогичной основе разработано приближение производных в ИВС с точностью до 10^{-16}.

2. Обработка данных использует многообразие методов, в числе которых вычисление интеграла. Разработанный кусочно-интерполяционный метод используется для приближения подынтегральной функции. Интерполяция выполняется с помощью полиномов Лагранжа и Ньютона, преобразуемых в форму алгебраических полиномов с числовыми коэффициентами. Полученный полином интегрируется, приводя к инвариантным относительно степени полинома формулам Ньютона-Котеса. Коэффициенты формул не зависят от подынтегральной функции, промежутка интегрирования, хранятся в разделе констант программы. Пользовательский интерфейс программ стандартизируется до задания подынтегральной функции, промежутка интегрирования, как вариант может включать степень полинома и число подынтервалов. Показана сходимость метода, даны оценки скорости сходимости, представлены таблицы коэффициентов для стандартизации программ. Приведены коды программ и результаты эксперимента, согласно которым на промежутке длины 500 достигается граница погрешности

приближения интеграла порядка 10^{-20}. На промежутках стандартной для обработки данных длины достигается нулевая граница погрешности. Метод распространяется на приближение первообразной функции, в этом случае погрешность имеет порядок 10^{-19}. Одновременно с минимизацией погрешности минимизируется время вычисления интегралов и первообразных, что отличает метод от известных.

3. Показана возможность существенного снижения трудоемкости без потери точности в предложенном методе обработки данных дифференциальной модели при замене автоматического выбора параметров их пользовательским подбором и фиксированием. Согласно численному и программному эксперименту предложенная модификация превосходит известные методы по быстродействию, при этом достигает более высокой точности обработки данных. На этой основе разработаны быстродействующие алгоритмы воспроизведения с помощью хранимых коэффициентов специальных и стандартных функций с гладкостью приближения. В частности, функция Бесселя и гипергеометрическая функция, применяемые в моделях периодических процессов, реализуются гладким приближением на больших временных промежутках с быстродействием и точностью, превосходящими характеристики известных аналогов.

Наиболее практический характер результатов исследования данной главы содержат следующие выводы.

4. Разработан комплекс программ обработки данных в ИВС на основе предложенного метода для моделей жестких и нежестких задач, включающий автоматический и пользовательский выбор параметров для адаптации к классам моделей. С помощью комплекса выполнено моделирование различных периодических реакций и процессов. Результаты отличаются сравнительно высокой точностью и гладкостью обработки данных моделей в допустимых границах трудоемкости, что позволяет уточнить физико-химические параметры автоколебаний и повысить качество моделирования. Так, обработка данных модели Филда-Нойеса для колебательной реакции Белоусова-Жаботинского и

модели Дж. Хиггинса для релаксационных автоколебаний в системе гликолиза показала уточнение на два и более десятичных порядка в сравнении с известными методами значений концентраций реагентов реакций в произвольный момент времени. Уточнены экстремальные значения концентраций реагентов, и как следствие амплитуды и период колебаний. При этом существенное уточнение в процессе визуализации данных, в отличие от разностных методов, достигается за счет непрерывности приближенного решения жестких систем. На этой основе для релаксационных автоколебаний в системе гликолиза обнаружена качественная характеристика процесса изменения концентрации, которая не идентифицируется известными методами.

5. На основе кусочно-интерполяционной обработки данных модели Чумакова-Слинько выполнено моделирование гетерогенной колебательной реакции окисления молекулярного водорода. Сравнительно высокая точность (граница абсолютной погрешности порядка 10^{-18}) и непрерывность полученного приближения данных позволили уточнить верхнюю границу изменения константы скорости основной стадии реакции окисления молекулярного водорода на поверхности никелевого или платинового катализатора, при которой сохраняется автоколебательный процесс. При этом использование известных методов вследствие высокой границы абсолютной погрешности приближения приводит к неверному результату моделирования (автоколебательный характер процесса не идентифицируется). Данные уточнения повышают качество решения проблемы аккомодации тепловой энергии, выделяемой в рекомбинационных стадиях. Уточнены экстремальные значения концентраций реагентов реакции и как следствие амплитуды и период их колебаний. Уточнение амплитуды и периода колебаний адсорбированных компонентов реакции в сравнении с результатами, полученными с использованием специализированного программного комплекса для численного исследования динамики колебательных химических реакций на основе полунеявных методов, составило от 1% до 9% в зависимости от компонентов реакции.

6. На основе кусочно-интерполяционной обработки данных дифференциальной модели уточнены значения напряжения на контуре и тока индуктивного элемента автогенератора с внутренней обратной связью, уточнена динамика их изменения, значения амплитуды и периода релаксационных автоколебаний. Непрерывность приближения данных позволила в сравнении с известными методами точнее отобразить визуализацию автоколебательного характера изменения значений напряжения. В частности, исправляется принципиальная неточность визуализации с помощью разностных методов, состоящая в не идентичности отображения гармоник, – кусочно-интерполяционное отображение дает графически идентичные гармоники.

7. С применением разработанного комплекса программ обработки данных в ИВС получено уточненное (на 5 десятичных порядков) непрерывное приближение траектории полета КА с управлением при помощи «малой тяги». На основе полученных значений положения КА рассчитано уточненное время, необходимое для вывода КА на устойчивую периодическую орбиту, при управлении, соответствующем внешнему воздействию гравитационных сил. Получены уточненные параметры орбиты устойчивого движения КА. Достаточно малая временная сложность предложенной обработки данных позволяет применять метод для управления движением КА в режиме реального времени.

8. С применением комплекса программ выполнено моделирование возмущенного движения итальянского геодезического спутника LARES, выведенного на низкую околоземную орбиту. Сравнительно высокая точность предложенного метода обработки данных в сочетании с гладкостью и малой временной сложностью дают координаты ИСЗ с требуемой точностью в произвольный момент времени, что необходимо для измерений с помощью лазерного дальномера от пункта наблюдения до ИСЗ и позволяет предупреждать аварийный сход с орбиты.

ГЛАВА 6. ВЫСОКОТОЧНАЯ БЫСТРОДЕЙСТВУЮЩАЯ ИДЕНТИФИКАЦИЯ И ХРАНЕНИЕ ДАННЫХ МОДЕЛИ ДВИЖЕНИЯ СПУТНИКОВ ГЛОНАСС С ПРИМЕНЕНИЕМ КУСОЧНОЙ ИНТЕРПОЛЯЦИИ

В главе разработан комплекс программ обработки данных эфемерид на модели периодического возмущенного движения НКА ГЛОНАСС на околоземной орбите. С помощью комплекса выполнен расчет параметров движения НКА по оперативной эфемеридной информации из навигационного сообщения. Точность и гладкость предложенного метода обработки данных позволяют получить координаты и составляющие вектора скорости центра масс НКА с превышением границ требуемой точности в произвольно заданные моменты времени из 30-минутного интервала прогнозирования, при этом время расчета примерно вдвое меньше времени расчета на основе известных методов. Представлены алгоритмы и программы, которые позволяют сохранять гладкое аналитическое приближение компонентов траектории и скорости движения НКА ГЛОНАСС в памяти компьютера и восстанавливать его без повторного вычисления траектории в произвольной точке интервала прогнозирования за время единичного порядка без снижения точности приближений.

Материал главы соответствует содержанию публикации автора [199].

6.1. Подсистема космических аппаратов ГЛОНАСС. Глобальная навигационная спутниковая система предназначена для определения местоположения, скорости движения, точного времени морских, воздушных, сухопутных и космических потребителей, а также для выполнения дополнительных информационных функций.

Система ГЛОНАСС состоит из трех подсистем [200]:

– подсистемы космических аппаратов (ПКА);

– подсистемы контроля и управления (ПКУ);

– подсистемы потребителей (ПП).

ПКА системы ГЛОНАСС при полном развертывании включает в себя от 24-х до 30-ти штатных среднеорбитальных НКА, движущихся по круговым орбитам с номинальной высотой $19100\,км$, наклонением 64.8° и периодом обращения $11\,ч\,15\,мин\,44\,с$. Плоскости орбит НКА и их согласованное движение по орбитам выбраны таким образом, чтобы обеспечить непрерывное и глобальное покрытие навигационным полем земной поверхности и околоземного пространства, до высоты $2000\,км$. В частности, указанное значение периода, вследствие повышения устойчивости решений уравнений траекторного движения спутников в процессе моделирования и снижения корреляции между параметрами отдельных уравнений, как отмечено в [200], «позволило создать устойчивую орбитальную систему, не требующую в отличие от острорезонансных орбит GPS для своего поддержания корректирующих импульсов практически в течение всего срока активного существования». Для иллюстрации на рис. 6.1 сравниваются наземные трассы одного спутника ГЛОНАСС и одного спутника GPS.

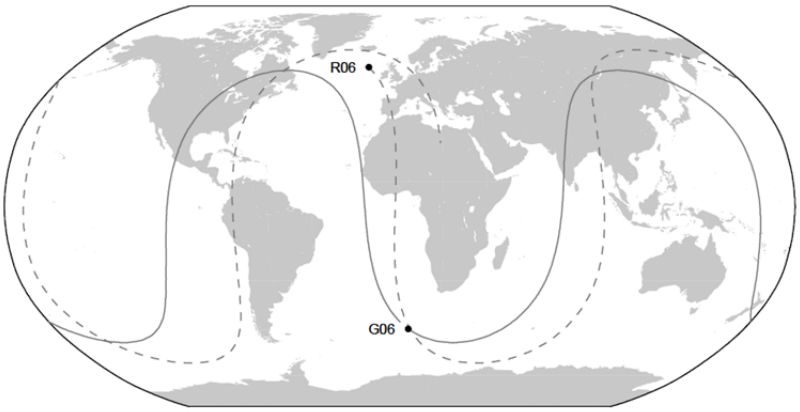

Рис. 6.1. Сравнение трассы орбиты спутника ГЛОНАСС (R06) и трассы спутника GPS (G06) для временного интервала в одни звездные сутки

На рис. 6.1 также наглядно представлено лучшее покрытие высокоширотных регионов спутником ГЛОНАСС вследствие большего наклона плоскостей орбиты ($i = 64.8^\circ$) в сравнении с GPS ($i = 55^\circ$), что обеспечивает 100%-ную доступность навигации на территории России даже

при условии выхода из орбитальной группировки нескольких космических аппаратов.

По состоянию на 05.08.2021 г. всего в составе орбитальной группировки ГЛОНАСС находилось 25 НКА, три из которых относятся к классу «ГЛОНАСС-К», остальные НКА класса «ГЛОНАСС-М». Информация о текущем составе и состоянии орбитальной группировки ГЛОНАСС доступна по адресу: https://www.glonass-iac.ru/glonass/sostavOG/. На этапе проектирования для системы ГЛОНАСС был принят частотный метод разделения сигналов различных КА: каждый из них использует свою пару несущих частот, одна из которых принадлежит диапазону L1, другая – диапазону L2. Выведенный на орбиту в 2011 году для летных испытаний КА модификации «Глонасс-К» 1-го этапа наряду с радиосигналами L1 и L2 с частотным разделением, полностью аналогичным сигналам «Глонасс-М», дополнительно излучает в диапазоне L3 радиосигналы открытого доступа с кодовым разделением. Модернизированные аппараты «Глонасс-М» начиная с № 55 также излучают навигационный радиосигнал с кодовым разделением в диапазоне L3. В настоящее время завершена разработка НКА «Глонасс-К2» с повышенными тактико-техническими характеристиками, планируется запуск и проведение летно-конструкторских испытаний. В составе навигационного сообщения перспективных сигналов с кодовым разделением в частотных диапазонах L1 и L3 ГЛОНАСС с борта НКА «Глонасс-К2» предусмотрена передача согласованной (в части скоростей и ускорений) оперативной эфемеридной иноформации (ЭИ) с повышенной разрядностью представления.

Навигационное сообщение содержится в псевдокадрах, каждый из которых содержит оперативную и неоперативную цифровую информацию. В отличие от GPS, где эфемериды вещания определяются модифицированными кеплеровскими элементами, эфемериды вещания спутников ГЛОНАСС определяются координатами и составляющими вектора скорости центра масс НКА в системе координат «Параметры Земли 1990 года» (ПЗ-90) последней

редакции [192]. Данная информация передается каждым КА в составе оперативной информации. Кроме того, в той же системе передаются ускорения спутников, вызванные Солнцем и Луной. Эфемериды спутников ГЛОНАСС обновляются каждые 30 минут. «В качестве шкалы системного времени системы ГЛОНАСС принята условная непрерывная шкала времени, формируемая на основе шкалы времени Центрального синхронизатора системы. Центральный синхронизатор оснащен водородными стандартами частоты. Опорной шкалой времени для системы ГЛОНАСС является национальная координированная шкала времени России UTC(SU). Расхождение между шкалой системного времени ГЛОНАСС и UTC(SU) не должно превышать 1 мс» [200].

«ПКУ состоит из центра управления системой ГЛОНАСС и сети станций измерения, управления и контроля. В задачи ПКУ входит контроль правильности функционирования ПКА, непрерывное уточнение параметров орбит и часов ПКА и выдача на НКА программных и разовых команд управления, информации навигационного сообщения или исходных данных для ее формирования. ПП состоит из множества навигационной аппаратуры потребителей (НАП), осуществляющей прием навигационных радиосигналов, измерение радионавигационных параметров, решение задач определения местоположения, скорости и времени на основе обработки полученных измерений» [200].

Аппаратные и программно-алгоритмические средства глобальных навигационный спутниковых систем постоянно совершенствуются с целью повышения точности навигационных определений.

Для определения местоположения и скорости НАП ГЛОНАСС принимает навигационные радиосигналы не менее чем от 4-х НКА и выполняет беззапросные измерения псевдодальности и радиальной псевдоскорости относительно каждого НКА, а также прием и обработку навигационного сообщения, содержащегося в составе навигационных радиосигналов. В

навигационном сообщении описывается положение НКА в пространстве и времени. В результате обработки полученных измерений и данных навигационного сообщения определяются координаты местоположения потребителя и составляющие вектора скорости его движения, а также осуществляется синхронизация шкалы времени НАП.

При определении местоположения в навигационном приемнике текущее положение НКА ГЛОНАСС рассчитывается по оперативной эфемеридной информации из навигационного сообщения на основе обработки данных модели движения НКА.

6.2. Эталонный алгоритм расчета координат и скорости центра масс навигационного космического аппарата ГЛОНАСС по данным эфемерид. В интерфейсном контрольном документе (ИКД) ГЛОНАСС конструктивно описан алгоритм расчета координат и составляющих вектора скорости центра масс НКА на заданный момент времени t_i шкалы МДВ (московского декретного времени) на 30-минутном интервале прогнозирования по данным эфемерид [200]. Пересчет эфемерид потребителем с момента t_b шкалы МДВ на заданный момент времени t_i той же шкалы, $|t_i - t_b| \le 15\,мин$, проводится методом численного интегрирования дифференциальных уравнений движения центра масс НКА. В правых частях этих уравнений задаются функции ускорения, определяемые геоцентрической константой гравитационного поля Земли с учетом влияния атмосферы GM, зональным гармоническим коэффициентом второй степени J_2^0, характеризующим полярное сжатие Земли, а также ускорениями от лунно-солнечных гравитационных возмущений. Эти уравнения движения определены в виде системы [200]:

$$\left.\begin{aligned}
&\frac{dx_0}{dt} = v_{x_0},\ \frac{dy_0}{dt} = v_{y_0},\ \frac{dz_0}{dt} = v_{z_0},\\
&\frac{dv_{x_0}}{dt} = -G\hat{M}\cdot\hat{x}_0 - \frac{3}{2}J_2^0\, G\hat{M}\cdot\hat{x}_0\,\rho^2(1-5\hat{z}_0^2) + j_{x0c} + j_{x0л},\\
&\frac{dv_{y_0}}{dt} = -G\hat{M}\cdot\hat{y}_0 - \frac{3}{2}J_2^0\, G\hat{M}\cdot\hat{y}_0\,\rho^2(1-5\hat{z}_0^2) + j_{y0c} + j_{y0л},\\
&\frac{dv_{z_0}}{dt} = -G\hat{M}\cdot\hat{z}_0 - \frac{3}{2}J_2^0\, G\hat{M}\cdot\hat{z}_0\,\rho^2(3-5\hat{z}_0^2) + j_{z0c} + j_{z0л},
\end{aligned}\right\} \qquad (6.1)$$

где $G\hat{M} = \dfrac{GM}{r_0^2}$; $\hat{x}_0 = \dfrac{x_0}{r_0}$; $\hat{y}_0 = \dfrac{y_0}{r_0}$; $\hat{z}_0 = \dfrac{z_0}{r_0}$; $\rho = \dfrac{a_e}{r_0}$; $r_0 = \sqrt{x_0^2 + y_0^2 + z_0^2}$;

j_{x0c}, j_{y0c}, j_{z0c}, $j_{x0л}$, $j_{y0л}$, $j_{z0л}$ – ускорения от солнечных и лунных гравитационных возмущений соответственно; a_e – большая полуось общеземного эллипсоида, $a_e = 6378136\ м$; GM – геоцентрическая константа гравитационного поля Земли с учетом атмосферы, $GM = (398600441.8 \pm 0.8)\cdot 10^6\ м^3/с^2$; J_2^0 – зональный гармонический коэффициент второй степени, характеризующий полярное сжатие Земли, $J_2^0 = 1082625.75 \times 10^{-9}$.

Следующие исходные данные необходимы для пересчета эфемерид: N_4 – номер эфемеридного четырехлетнего периода; N_T – номер эфемеридных суток в эфемеридном четырехлетнем периоде; момент времени t_b; координаты и составляющие вектора скорости центра масс НКА на момент времени t_b из оперативной информации ГЛОНАСС; заданный момент времени t_i шкалы МДВ, на который необходимо пересчитать координаты и составляющие вектора скорости НКА.

Уравнения (6.1) интегрируются в прямоугольной инерциальной геоцентрической системе координат $OX_0Y_0Z_0$, «начало которой O совпадает с началом координат системы ПЗ-90, ось OX_0 направлена в точку весеннего

равноденствия, ось OZ_0 направлена на Северный полюс мира и совпадает с осью OZ системы ПЗ-90, ось OY_0 дополняет систему до правой» [200].

Ускорения от лунных и солнечных гравитационных возмущений в инерциальной геоцентрической прямоугольной системе координат $OX_0Y_0Z_0$ вычисляются по формулам [200]:

$$j_{x0k} = \hat{G}_k\left[\frac{\xi_k - \hat{x}_k}{\Delta_k} - \xi_k\right],\ j_{y0k} = \hat{G}_k\left[\frac{\eta_k - \hat{y}_k}{\Delta_k} - \eta_k\right],\ j_{z0k} = \hat{G}_k\left[\frac{\Im_k - \hat{z}_k}{\Delta_k} - \Im_k\right], \quad (6.2)$$

где $\hat{x}_k = \frac{x_0}{r_k};\ \hat{y}_k = \frac{y_0}{r_k};\ \hat{z}_k = \frac{z_0}{r_k};\ \hat{G}_k = \frac{G_k}{r_k^2};$

$\Delta_k = \left[(\xi_k - \hat{x}_k)^2 + (\eta_k - \hat{y}_k)^2 + (\Im_k - \hat{z}_k)^2\right]^{3/2};$

k – индекс возмущающего тела, $k =$«$л$» для Луны и $k =$«c» для Солнца;

ξ_k, η_k, $\Im_k$, r_k – направляющие косинусы и удаление возмущающего тела в системе координат $OX_0Y_0Z_0$ на момент времени t;

$G_л$, G_c – константы гравитационного поля Луны и Солнца соответственно, $G_л = 4902.799 \times 10^9\ м^3/с^2;\ G_c = 13271244\ .0 \times 10^{13}\ м^3/с^2.$

Входящие в равенства (6.2) направляющие косинусы и удаления Луны и Солнца ξ_k, η_k, $\Im_k$, r_k рассчитываются из соотношений [201, 202]:

$$\left.\begin{aligned}
\xi_л &= (\sin\vartheta_л\cos\Gamma' + \cos\vartheta_л\sin\Gamma')\xi_{11} + (\cos\vartheta_л\cos\Gamma' - \sin\vartheta_л\sin\Gamma')\xi_{12},\\
\eta_л &= (\sin\vartheta_л\cos\Gamma' + \cos\vartheta_л\sin\Gamma')\eta_{11} + (\cos\vartheta_л\cos\Gamma' - \sin\vartheta_л\sin\Gamma')\eta_{12},\\
\Im_л &= (\sin\vartheta_л\cos\Gamma' + \cos\vartheta_л\sin\Gamma')\Im_{11} + (\cos\vartheta_л\cos\Gamma' - \sin\vartheta_л\sin\Gamma')\Im_{12},\\
\xi_c &= \cos\vartheta_c\cos\omega_c - \sin\vartheta_c\sin\omega_c,\\
\eta_c &= (\sin\vartheta_c\cos\omega_c + \cos\vartheta_c\sin\omega_c)\cos\varepsilon,\\
\Im_c &= (\sin\vartheta_c\cos\omega_c + \cos\vartheta_c\sin\omega_c)\sin\varepsilon,\\
r_k &= a_k(1 - e_k\cos\mathrm{E}_k),
\end{aligned}\right\} \quad (6.3)$$

где $\sin\vartheta_k = \frac{\sqrt{1-e_k^2}\sin\mathrm{E}_k}{1 - e_k\cos\mathrm{E}_k};\ \cos\vartheta_k = \frac{\cos\mathrm{E}_k - e_k}{1 - e_k\cos\mathrm{E}_k};\ \mathrm{E}_k = q_k + e_k\sin\mathrm{E}_k$ – уравнение Кеплера для эксцентрической аномалии, которое решается методом итераций,

начальное приближение $\mathrm{E}_k = q_k$, пока $\left|\mathrm{E}_{ki} - \mathrm{E}_{k(i-1)}\right|$ не будет меньше 10^{-8} (i – номер итерации);

$\xi_{11} = \sin\Omega_л \cos\ \Omega_л(1-\cos i_л),\ \xi_{12} = 1-\sin^2\Omega_л(1-\cos i_л),$

$\eta_{11} = \xi^* \cos\ \varepsilon - \Im^* \sin\varepsilon,\ \eta_{12} = \xi_{11}\cos\ \varepsilon + \eta^* \sin\varepsilon,$

$\Im_{11} = \xi^* \sin\ \varepsilon + \Im^* \cos\varepsilon,\ \Im_{12} = \xi_{11}\sin\ \varepsilon - \eta^* \cos\varepsilon,$

$\xi^* = 1-\cos^2\Omega_л(1-\cos i_л),\ \eta^* = \sin\Omega_л \sin i_л,\ \Im^* = \cos\Omega_л \sin i_л;$

$a_л$ – большая полуось орбиты Луны, $a_л = 3.84385243\times10^5\ км$;

a_c – большая полуось «орбиты» Солнца, $a_c = 1.49598\times10^8\ км$;

$e_л$ – эксцентриситет лунной орбиты, $e_л = 0.054900489$;

e_c – эксцентриситет «орбиты» Солнца, $e_c = 0.016719$;

$i_л$ – среднее наклонение орбиты Луны к плоскости эклиптики, $i_л = 0.0898041080\ рад$.

Параметры нутации Луны и Солнца на момент времени t по шкале МДВ рассчитываются по формулам:

– средняя аномалия Луны и Солнца, рад:

$$q_л = 2.3555557435 + 8328.6914257190\ \times T + 0.0001545547\ \times T^2,$$

$$q_c = 6.2400601269 + 628.3019551714\ \times T - 2.6820\times10^{-6}\times T^2;$$

– средняя долгота восходящего узла Луны, рад:

$$\Omega_л = 2.1824391966\ - 33.7570459536\ \times T + 0.0000362262\ \times T^2;$$

– средняя долгота перигея орбиты Луны, рад:

$$\Gamma' = 1.4547885346 + 71.0176852437\times T - 0.0001801481\times T^2;$$

– средняя тропическая долгота перигея «орбиты» Солнца, рад:

$$\omega_c = -7.6281824375 + 0.0300101976\ \times T + 7.9741\times10^{-6}\times T^2$$

– средний наклон эклиптики к экватору, рад:

$$\varepsilon = 0.4090926006 - 0.0002270711\ \times T;$$

– время от эпохи 2000 года, 1 января, 12 ч (UTC(SU)) до момента времени шкалы МДВ t в юлианских столетиях по 36525 эфемеридных суток:

$$T = (JD0 + (t - 10800)/86400 - 2451545.0)/36525,$$

где t – момент по шкале МДВ, к которому привязаны эфемериды ГЛОНАСС, в секундах; 10800 – разница между МДВ и UTC(SU), в секундах; 24515450 – юлианская дата на 12 ч 1 января 2000 года (UTC(SU)), $JD0$ – текущая юлианская дата на 0 часов шкалы МДВ, которая до окончания 2119 года по МДВ рассчитывается по соотношению:

$$JD0 = 1461\ (N_4^{тек} - 1) + N_T^{тек} + 2450082\ .5,$$

где $N_4^{тек}$ – номер текущего четырехлетия, $N_T^{тек}$ – номер текущих суток, значения которых определяются из данных, переданных в навигационном сообщении ГЛОНАСС.

Начальными условиями для интегрирования системы (6.1) являются координаты центра масс НКА $x_0(t_b)$, $y_0(t_b)$, $z_0(t_b)$ и составляющие его вектора скорости $\dot{x}_0(t_b)$, $\dot{y}_0(t_b)$, $\dot{z}_0(t_b)$ в инерциальной геоцентрической системе координат $OX_0Y_0Z_0$ на момент времени t_b шкалы МДВ. Данные начальные условия вычисляются путем пересчета передаваемых в навигационном сообщении координат $x(t_b)$, $y(t_b)$, $z(t_b)$ и составляющих вектора скорости $\dot{x}(t_b)$, $\dot{y}(t_b)$, $\dot{z}(t_b)$ центра масс НКА в связанной с Землей системе координат ПЗ-90. Пересчет осуществляется по следующим формулам:

$$\begin{aligned}
x_0(t_b) &= x(t_b)\cos(S(t_b)) - y(t_b)\sin(S(t_b)),\\
y_0(t_b) &= x(t_b)\sin(S(t_b)) + y(t_b)\cos(S(t_b)),\\
z_0(t_b) &= z(t_b),\\
\dot{x}_0(t_b) &= \dot{x}(t_b)\cos(S(t_b)) - \dot{y}(t_b)\sin(S(t_b)) - \omega_з y_0(t_b),\\
\dot{y}_0(t_b) &= \dot{x}(t_b)\sin(S(t_b)) + \dot{y}(t_b)\cos(S(t_b)) + \omega_з x_0(t_b),\\
\dot{z}_0(t_b) &= \dot{z}(t_b),\\
S(t_b) &= \mathrm{GST} + \omega_з(t_b - 10800),
\end{aligned} \qquad (6.4)$$

где $\omega_з$ – средняя угловая скорость вращения Земли относительно точки весеннего равноденствия, $\omega_з = 7.2921151467 \times 10^{-5}\ рад/с$; GST – истинное звезное время по Гринвичу (в радианах).

Полученные в результате приближенного решения системы (6.1) – (6.3) полученные координаты центра масс $x_0(t_i)$, $y_0(t_i)$, $z_0(t_i)$ и составляющие его вектора скорости $\dot{x}_0(t_i)$, $\dot{y}_0(t_i)$, $\dot{z}_0(t_i)$ в инерциальной системе координат пересчиваются в связанную с Землей систему ПЗ-90 по следующим соотношениям:

$$\begin{aligned}
&x(t_i) = x_0(t_i)\cos(S(t_i)) + y_0(t_i)\sin(S(t_i)),\\
&y(t_i) = -x_0(t_i)\sin(S(t_i)) + y_0(t_i)\cos(S(t_i)),\\
&z(t_i) = z_0(t_i),\\
&\dot{x}(t_i) = \dot{x}_0(t_i)\cos(S(t_i)) + \dot{y}_0(t_i)\sin(S(t_i)) + \omega_з y(t_i),\\
&\dot{y}(t_i) = -\dot{x}_0(t_i)\sin(S(t_i)) + \dot{y}_0(t_i)\cos(S(t_i)) - \omega_з x(t_i),\\
&\dot{z}(t_i) = \dot{z}_0(t_i),\\
&S(t_i) = \mathrm{GST} + \omega_з(t_i - 10800).
\end{aligned} \qquad (6.5)$$

Как отмечено в [200] «вместо истинного звездного времени по Грнивичу GST, в формулах (6.4), (6.5) допускается использовать среднее звездное время по Гринвичу GMST», вычисляемое по формуле:

$$\begin{aligned}
\mathrm{GMST} = \mathrm{ERA} &+ 7.03270726 \times 10^{-8} + 0.0223603658710194 \times \mathrm{T}_\Delta +\\
&+ 6.7465784654 \times 10^{-6} \times \mathrm{T}_\Delta^2 - 2.1332 \times 10^{-12} \times \mathrm{T}_\Delta^3 -\\
&- 1.452308 \times 10^{-10} \times \mathrm{T}_\Delta^4 - 1.784 \times 10^{-13} \times \mathrm{T}_\Delta^5,
\end{aligned}$$

где ERA – угол поворота Земли, рад,

$$\mathrm{ERA} = 2\pi(0.7790572732640 + 1.00273781191135448 \times (\mathrm{JD0} - 2451545.0));$$

T_Δ – время от эпохи 2000 года 1 января, 12 ч (UTC(SU)) до текущей эпохи в юлианских столетиях по 36525 эфемеридных суток, $\mathrm{T}_\Delta = (JD0 - 2451545.0)/36525$.

6.3. Идентификация и хранение данных модели движения спутника по значениям эфемерид с применением кусочной интерполяции. В ИКД ГЛОНАСС в качестве примера численного метода, который может быть использован для интегрирования уравнений движения (6.1), приводится классический метод Рунге-Кутты 4-го порядка. В работе [203] отмечено, что в связи с необходимостью повышения точности расчета положения спутника ГЛОНАСС по данным эфемерид более подходящим для численного интегрирования соответствующих уравнений движения является метод Рунге-Кутты-Фельберга 5-го порядка, с учетом повышения времени расчета более чем на 50%. Ниже исследовано применение описанного в предыдущих главах кусочно-интерполяционного метода для интегрирования уравнений движения (6.1) с целью повышения точности приближения при условии снижения времени расчета, а также с целью построения непрерывного приближения координат и составляющих вектора скорости центра масс НКА на интервале $\pm 15\,мин$ относительно момента t_b.

Эфемериды НКА ГЛОНАСС записываются в формате RINEX *.yyG [204] с 30-минутным интервалом в виде векторных составляющих положения, скорости и ускорения спутника. Формат RINEX (Receiver Independent Exchange Format) был разработан для унификации измерений с различной спутниковой аппаратуры и последующей удобной обработки ГНСС-данных в программном обеспечении. Файлы формата RINEX, содержащие штатные эфемериды ГЛОНАСС, представлены в свободном доступе, в частности, в FTP-архиве прикладного потребительского центра государственной корпорации «Роскосмос», который функционирует на базе средств Информационно-аналитического центра координатно-временного и навигационного обеспечения АО «ЦНИИмаш» (ИАЦ КВНО): ftp://ftp.glonass-iac.ru. На рис. 6.2 представлен пример содержимого файла формата RINEX с эфемеридами ГЛОНАСС.

```
     2.10           GLONASS NAV DATA                         RINEX VERSION / TYPE
EPHEMCHECK 2.00     MCC E.BIKKININ      06-08-21 04:41       PGM / RUN BY / DATE
A new version of accumulating navigation files by IANC       COMMENT
                                                             END OF HEADER
 1 21 08 05 00 15  0.0 8.334871381521E-05 0.000000000000E+00 0.000000000000E+00
    2.485515820312E+04-7.984914779663E-01 1.862645149231E-09 0.000000000000E+00
    3.459438476562E+02-6.519222259521E-02-1.862645149231E-09 1.000000000000E+00
   -5.760185546875E+03-3.447617530823E+00-1.862645149231E-09 0.000000000000E+00
 2 21 08 05 00 15  0.0 4.885662347078E-04 9.094947017729E-13 0.000000000000E+00
    1.944737207031E+04 1.946067810059E+00 9.313225746155E-10 0.000000000000E+00
   -8.144775878906E+03-2.716627120972E-01-9.313225746155E-10-4.000000000000E+00
    1.435309472656E+04-2.804148674011E+00-2.793967723846E-09 0.000000000000E+00
 3 21 08 05 00 15  0.0 4.073698073626E-05 9.094947017729E-13 0.000000000000E+00
    2.335095703125E+03 3.113179206848E+00 0.000000000000E+00 0.000000000000E+00
   -1.108062109375E+04-2.569732666016E-01 9.313225746155E-10 5.000000000000E+00
    2.289445751953E+04-4.474649429321E-01-1.862645149231E-09 0.000000000000E+00
 4 21 08 05 00 15  0.0 9.883940219879E-05 1.818989403546E-12 0.000000000000E+00
   -1.577019482422E+04 2.542145729065E+00-1.862645149231E-09 0.000000000000E+00
   -7.564973144531E+03-7.650279998779E-02 2.793967723846E-09 6.000000000000E+00
    1.859590234375E+04 2.127021789551E+00 0.000000000000E+00 0.000000000000E+00
```

Рис. 6.2. Пример содержимого файла формата RINEX с обобщенными в ИАЦ КВНО бортовыми эфемеридами НКА ГЛОНАСС

Кусочно-интерполяционное решение задачи (6.1) – (6.3) выполняется на основе описанного в п. 6.2 алгоритма, на основе результатов численного эксперимента выполняется сравнение по точности, времени расчета и качествам полученных приближений с данными, рассчитанными по известным методам.

Обобщенные в ИАЦ КВНО бортовые эфемериды НКА ГЛОНАСС №730 (дата запуска 12/14/2009, дата ввода 01/30/2010) в системе ПЗ-90.11 на момент времени $t_b = 11700$ даты 05.08.2021 ($N_T = 583$, $N_4 = 7$) шкалы МДВ, пересчитанные по формулам (6.4) в инерциальную геоцентрическую систему координат представлены ниже:

$$\begin{aligned} x_0(t_b) &= \ \ 1.85671840\,522396\text{e}{+}07, \quad \dot{x}_0(t_b) = \ \ 5.72204100\,174071\text{e}{+}02, \\ y_0(t_b) &= -1.65274995\,936504\text{e}{+}07, \quad \dot{y}_0(t_b) = \ \ 1.84501010\,135317\text{e}{+}03, \\ z_0(t_b) &= -5.76018554\,687500\text{e}{+}06, \quad \dot{z}_0(t_b) = -3.44761753\,082300\text{e}{+}03. \end{aligned} \qquad (6.6)$$

Кусочно-интерполяционное приближение решения задачи (6.1) – (6.6) строится на интервалах $\pm 15\,мин$ относительно момента t_b ($t \in [t_b - \Delta t,\, t_b]$ и $t \in [t_b,\ t_b + \Delta t]$, $\Delta t = 900\ c$). Значения лунно-солнечных ускорений рассчитываются по описанному в п. 6.2 алгоритму при каждом вызове функции правой части (6.1) в процессе численного моделирования. Значения текущей юлианской даты $JD0$ и $GMST$ рассчитаны по описанному в п. 6.2 алгоритму и

для $N_T = 583$, $N_4 = 7$ имеют следующие значения: $JD0 = 2459431.5$; $GMST = 496929.057040$. С целью повышения быстродействия без потери точности приближения применяется модификация метода кусочно-интерполяционного решения задачи Коши для ОДУ, заключающаяся в замене автоматического подбора параметров пользовательским подбором и фиксированием параметров метода, подробно исследованная в главе 5 и в приложении к ней.

Результаты пересчета эфемерид НКА ГЛОНАСС в системе ПЗ-90.11 на момент времени $t_i = 12600$ с даты 05.08.2021 ($N_T = 583$, $N_4 = 7$) шкалы МДВ, полученные с помощью кусочно-интерполяционного метода с параметрами: $n = 8$, $\ell = 12$, представлены ниже.

$$x(t_i) = \quad 2.39489258119706\mathrm{e}+04, \quad \dot{x}(t_i) = -1.2100487088\,2318\mathrm{e}+00,$$
$$y(t_i) = \quad 3.40159756877465\mathrm{e}+02, \quad \dot{y}(t_i) = \quad 6.1365337375\,4929\mathrm{e}-02,$$
$$z(t_i) = -8.7971001572\,5756\mathrm{e}+03, \quad \dot{z}(t_i) = -3.29014462102794\mathrm{e}+00.$$

Код программы для расчета эфемерид НКА ГЛОНАСС в системе ПЗ-90.11 на на заданный момент времени t_i шкалы МДВ на 30-минутном интервале прогнозирования по данным эфемерид на основе кусочно-интерполяционного решения задачи (6.1) – (6.6) представлен в листинге 6.1.

Листинг 6.1

```
program fpi_glonass; {$APPTYPE CONSOLE} uses SysUtils, Math;
const Neq=6; {число уравнений в системе ОДУ}
N4=7; {номер эфемеридного четырехлетнего периода}
NT=583; {номер эфемеридных суток в эфемеридном четырехлетнем периоде}
tb=11700; {момент времени из оперативной информации ГЛОНАСС}
ti=12600; {заданный момент времени шкалы МДВ, на который необходимо пересчитать
координаты и составляющие вектора скорости НКА}
type vectNeq = array[1..Neq] of extended;
 const y00:vectNeq = (24855158.20312, 345943.8476562, -5760185.546875,
 -798.4914779663, -65.19222259521, -3447.617530823); {координаты и составляющие
вектора скорости центра масс НКА на момент времени tb в системе ПЗ-90.11 из
оперативной информации ГЛОНАСС}
procedure   pi_calculation_of_ephemeris_glonass(N4,NT:integer;   tb,ti:extended;
y00:vectNeq);
const Npol=8;{степень n интерполяционных полиномов}
k_iter=12; {граница числа итераций q}
e_l=0.054900489; e_c=0.016719; i_l=0.0898041080; w3=7.2921151467e-5;
J2=1082.62575e-6; a_c=1.49598e+11; GM=3.986004418e+14; ae=6378136;
G_l=4902.799e+9; G_c=13271244.0e+13; a_l=3.84385243e+8;
{FILEPATH='D:\coeffs.bin'; type Arr_coeff = array[0..29] of extended;
```

```
var f_coeff: file of Arr_coeff; dd: Arr_coeff;}
type Prc=array[1..8] of extended; matr= array[0..Npol,0..Npol] of integer;
var y0,yn,yn0,yr: vectNeq; JD0,GMST,ERA,dT,S:extended; i:byte;
 vPrc:array[0..Npol] of Prc;
procedure ls_acceleration_prep(x:extended; var KEKr_:Prc);
var sinv_l,cosv_l,sinv_c, cosv_c,w_c,eps,Gamma_,Ksi11,Ksi12,Eta11,Eta12, Kapa11,
Kapa12:extended; q_l,q_c, T, Omega_l, Ksi_, Eta_, Kapa_, El, Ec:extended;
function Ekk(q_k,e_k:extended):extended; var Ek_,Ek_new:extended; begin
Ek_new:=q_k; repeat Ek_:=Ek_new; Ek_new:=q_k+e_k*sin(Ek_); until abs(Ek_new-
Ek_)<1e-15; Ekk:=Ek_; end;
begin T:=(JD0+(x-10800)/86400-2451545.0)/36525;
q_l:=2.3555557435+8328.6914257190*T+0.0001545547*sqr(T);
q_c:=6.2400601269+628.3019551714*T-(2.6820e-6)*sqr(T);
Omega_l:=2.1824391966-33.7570459536*T+0.0000362262*sqr(T);
Gamma_:=1.4547885346+71.0176852437*T-0.0001801481*sqr(T);
w_c:=-7.6281824375+0.0300101976*T+(7.9741e-6)*sqr(T);
eps:=0.4090926006-0.0002270711*T; Ksi_:=1-sqr(cos(Omega_l))*(1-cos(i_l));
Eta_:=sin(Omega_l)*sin(i_l); Kapa_:=cos(Omega_l)*sin(i_l);
Ksi11:=sin(Omega_l)*cos(Omega_l)*(1-cos(i_l));
Ksi12:=1-sqr(sin(Omega_l))*(1-cos(i_l)); Eta11:=Ksi_*cos(eps)-Kapa_*sin(eps);
Eta12:=Ksi11*cos(eps)+Eta_*sin(eps); Kapa11:=Ksi_*sin(eps)+Kapa_*cos(eps);
Kapa12:=Ksi11*sin(eps)-Eta_*cos(eps); El:=Ekk(q_l,e_l); Ec:=Ekk(q_c,e_c);
sinv_l:=(sqrt(1-sqr(e_l))*sin(El))/(1-e_l*cos(El));
sinv_c:=(sqrt(1-sqr(e_c))*sin(Ec))/(1-e_c*cos(Ec));
cosv_l:=(cos(El)-e_l)/(1-e_l*cos(El)); cosv_c:=(cos(Ec)-e_c)/(1-e_c*cos(Ec));
KEKr_[1]:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Ksi11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Ksi12;
KEKr_[2]:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Eta11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Eta12;
KEKr_[3]:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Kapa11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Kapa12; KEKr_[7]:=a_l*(1-e_l*cos(El));
KEKr_[4]:=cosv_c*cos(w_c)-sinv_c*sin(w_c);
KEKr_[5]:=(sinv_c*cos(w_c)+cosv_c*sin(w_c))*cos(eps);
KEKr_[6]:=(sinv_c*cos(w_c)+cosv_c*sin(w_c))*sin(eps);
KEKr_[8]:=a_c*(1-e_c*cos(Ec)); end;
{подпрограмма-функция вычисления значений компонентов вектор-функции правой
части (6.1)} function f(eq:integer;x:extended;yy:vectNeq;KEKr:Prc):extended;
var r,GM_,y1_,y2_,y3_,ro,jx0_c,jy0_c,jz0_c,jx0_l,jy0_l,jz0_l:extended;
procedure ls_acceleration(y1,y2,y3:extended;KEKr:Prc; var jx0_c,jy0_c,jz0_c,
jx0_l, jy0_l,jz0_l:extended);
var delta_c,delta_l,x_c,y_c,z_c,x_l,y_l,z_l,Gl_,Gc_:extended;
begin x_l:=y1/KEKr[7]; y_l:=y2/KEKr[7]; z_l:=y3/KEKr[7]; Gl_:=G_l/sqr(KEKr[7]);
x_c:=y1/KEKr[8]; y_c:=y2/KEKr[8]; z_c:=y3/KEKr[8]; Gc_:=G_c/sqr(KEKr[8]);
delta_l:=Power(sqr(KEKr[1]-x_l)+sqr(KEKr[2]-y_l)+sqr(KEKr[3]-z_l),1.5);
delta_c:=Power(sqr(KEKr[4]-x_c)+sqr(KEKr[5]-y_c)+sqr(KEKr[6]-z_c),1.5);
jx0_l:=Gl_*((KEKr[1]-x_l)/delta_l-KEKr[1]); jy0_l:=Gl_*((KEKr[2]-y_l)/delta_l-
KEKr[2]); jz0_l:=Gl_*((KEKr[3]-z_l)/delta_l-KEKr[3]); jx0_c:=Gc_*((KEKr[4]-
x_c)/delta_c-KEKr[4]); jy0_c:=Gc_*((KEKr[5]-y_c)/delta_c-KEKr[5]);
jz0_c:=Gc_*((KEKr[6]-z_c)/delta_c-KEKr[6]);end;
begin case eq of 1: f:=yy[4]; 2: f:=yy[5]; 3: f:=yy[6]; else begin
r:=sqrt(yy[1]*yy[1]+yy[2]*yy[2]+yy[3]*yy[3]);
GM_:=GM/sqr(r); y1_:=yy[1]/r; y2_:=yy[2]/r; y3_:=yy[3]/r; ro:=ae/r;
ls_acceleration(yy[1],yy[2],yy[3],KEKr,jx0_c,jy0_c,jz0_c,jx0_l,jy0_l,jz0_l);
case eq of 4: f:=-GM_*y1_-1.5*J2*GM_*ro*ro*y1_*(1-5*sqr(y3_))+jx0_c+jx0_l;
5: f:=-GM_*y2_-1.5*J2*GM_*ro*ro*y2_*(1-5*sqr(y3_))+jy0_c+jy0_l;
6: f:=-GM_*y3_-1.5*J2*GM_*ro*ro*y3_*(3-5*sqr(y3_))+jz0_c+jz0_l; end; end; end;
end;
{кусочно-интерполяционное приближение решения задачи Коши (6.1) – (6.6)}
procedure RD(print:boolean;yin:vectNeq; A_nach,B_konech:extended; var
yout:vectNeq);const dp: array[1..8,1..8] of extended =
((1,-1/2,2/6,-6/24,24/120,-120/720,720/5040,-5040/40320),
(0,1/2,-3/6,11/24,-50/120,274/720,-1764/5040,13068/40320),
```

```
(0,0,1/6,-6/24,35/120,-225/720,1624/5040,-13132/40320),
(0,0,0,1/24,-10/120,85/720,-735/5040,6769/40320),
(0,0,0,0,1/120,-15/720,175/5040,-1960/40320),(0,0,0,0,0,1/720,-
21/5040,322/40320),(0,0,0,0,0,0,1/5040,-28/40320),(0,0,0,0,0,0,0,1/40320));
type Mmatr=array[0..Npol,0..Npol] of vectNeq; vect=array[0..Npol] of extended;
Mvect= array[0..Npol] of vectNeq;
var i,k,j,iter,r:byte; h:extended; x:vect; pp,s:vectNeq; y,fy,A:Mvect; dy:Mmatr;
begin h:=(B_konech-A_nach)/Npol;
for j:=0 to Npol do begin
x[j]:=A_nach+j*h; ls_acceleration_prep(x[j],vPrc[j]);end;
for j:=0 to Npol do for i:=1 to Neq do y[j,i]:=yin[i];
for i:=1 to Neq do begin fy[0,i]:=f(i,x[0],y[0],vPrc[0]); A[0,i]:=fy[0,i]; end;
for iter:=1 to k_iter do begin for j:=1 to Npol do for i:=1 to Neq do
fy[j,i]:=f(i,x[j],y[j],vPrc[j]);
for j:=0 to Npol-1 do for i:=1 to Neq do dy[1,j,i]:=fy[j+1,i]-fy[j,i];
for k:=2 to Npol do for j:=0 to Npol-k do for i:=1 to Neq do dy[k,j,i]:=dy[k-
1,j+1,i]-dy[k-1,j,i];
for k:=1 to Npol do begin for i:=1 to Neq do s[i]:=0; for j:=k to Npol do
for i:=1 to Neq do s[i]:=s[i]+dp[k,j]*dy[j,0,i];for i:=1 to Neq do A[k,i]:=s[i];
end;for j:=1 to Npol do begin for i:=1 to Neq do pp[i]:=A[Npol,i]/(Npol+1);
for r:=Npol-1 downto 0 do for i:=1 to Neq do pp[i]:=pp[i]*j+A[r,i]/(r+1);
for i:=1 to Neq do y[j,i]:=pp[i]*h*j+y[0,i]; end; end;
{сохранение массива коэффициентов полиномов, аппроксимирующих координаты центра
масс НКА на интервале прогнозирования, в типизированный файл (массив состоит из
одной строки с коэффициентами полиномиальных приближений компонентов решения
системы, соответствующих координатам x, y и z)}
{if print then begin AssignFile(f_coeff,FILEPATH); rewrite(f_coeff); for i:=1 to
3 do dd[(Npol+2)*(i-1)]:=y[0,i];
for k:=1 to Npol+1 do for i:=1 to 3 do dd[k+(Npol+2)*(i-1)]:=A[k-1,i]*h/k;
write(f_coeff,dd); Close(f_coeff); end;}
for i:=1 to Neq do yout[i]:=y[Npol,i]; end;
begin JD0:=1461*(N4-1)+NT+2450082.5; dT:=(JD0-2451545.0)/36525;
ERA:=2*PI*(0.7790572732640+1.00273781191135448*(JD0-2451545.0));
GMST:=ERA+7.03270726e-8+0.0223603658710194*dT+6.7465784654e-6*power(dT,2)
-2.1332e-12*power(dT,3)-1.452308e-10*power(dT,4)-1.784e-13*power(dT,5);
{расчет начальных условий для интегрирования системы (1) путем пересчета
координат и составляющих вектора скорости центра масс НКА в геоцентрическую
систему координат} S:=GMST+w3*(tb-10800); y0[1]:=y00[1]*cos(S)-y00[2]*sin(S);
y0[2]:=y00[1]*sin(S)+y00[2]*cos(S); y0[3]:=y00[3];
y0[4]:=y00[4]*cos(S)-y00[5]*sin(S)-w3*y0[2];
y0[5]:=y00[4]*sin(S)+y00[5]*cos(S)+w3*y0[1]; y0[6]:=y00[6];
RD(True,y0,tb,ti,yn0); {кусочно-интерполяционное приближение решения задачи
Коши}{пересчет результатов приближения координат и составляющих вектора скорости
центра масс НКА в момент времени ti в систему координат ПЗ-90.11}
S:=GMST+w3*(ti-10800);
yn[1]:=yn0[1]*cos(S)+yn0[2]*sin(S);          yn[2]:=-yn0[1]*sin(S)+yn0[2]*cos(S);
yn[3]:=yn0[3];
yn[4]:=yn0[4]*cos(S)+yn0[5]*sin(S)+w3*yn[2];                                yn[5]:=-
yn0[4]*sin(S)+yn0[5]*cos(S)-w3*yn[1]; yn[6]:=yn0[6];
writeln(ti);   writeln('   x=',yn[1],'          y=',yn[2],'          z=',yn[3]);
writeln('vx=',yn[4],'  vy=',yn[5],'  vz=',yn[6]);
RD(False,yn0,ti,tb,yr);  {расчет погрешности приближения путем обратного
интегрирования задачи Коши}
writeln; for i:=1 to Neq do write('Pogr',i,' =',Format(' %1.4e',[abs(yr[i]-
y0[i])]),'; '); end;
begin pi_calculation_of_ephemeris_glonass(N4,NT,tb,ti,y00); readln; end.
```

Результат работы программы:

```
  ti= 1.26000000000000E+0004
  x= 2.39489258119706E+0007    y= 3.40159756877465E+0005    z=-8.79710015725756E+0006
 vx=-1.21004870882318E+0003  vy= 6.13653373754929E+0001  vz=-3.29014462102794E+0003
P1= 0.000E+000; P2= 0.000E+000; P3= 0.000E+000; P4= 0.000E+000; P5= 0.000E+000; P6= 0.000E+000;
```

Помимо фиксирования параметров метода для дополнительного снижения временной сложности применяемого метода с учетом специфики задачи (6.1) – (6.6) и ограниченности длины отрезка интегрирования в программной реализации кусочно-интерполяционного метода, представленной в листинге 6.1 в виде подпрограммы *RD*, внесены некоторые изменения в сравнении с программными реализациями, представленными в главе 5 и в приложении к ней. Именно, отрезок интегрирования не разбивается интервалы и подынтервалы ($\beta_r - \alpha_r = t_i - t_b$, $k = 0$, см. глава 3, равенства (3.3), (3.4)). С учетом этого по представленному в главе 3 алгоритму для каждого компонена вектор-функции правой части (6.1) строится интерполяционный полином Ньютона, который приводится к виду

$$P_n(t) = a_0 + \sum_{k=1}^{n} a_k \left(\frac{t - t_b}{h} \right)^k,\ a_0 = u(t_b),\ a_k = \sum_{j=k}^{n} \Delta^j u_0 \frac{d_{jk}}{j!}, \qquad (6.7)$$

где $u(t)$ – интерполируемая функция, $\Delta^j u_0$ – конечная разность j-го порядка в узле t_b, $h = (t_i - t_b)/n$. Целочисленные коэффициенты d_{jk}, $j = \overline{1, n}$, $k = \overline{0, j}$, не зависящие, как было отмечено в п. 2.1, ни от интерполируемой функции, ни от области приближения, априори рассчитаны с помощью процедуры *Viet* (листинг 2.1) для степени $n = 10$, также априори вычислены факториалы из (6.7) и значения $d_{jk} / j!$, $j = \overline{1, n}$, $k = \overline{0, j}$, которые представлены в приведенной выше программе в качестве числового двумерного массива *dp*. Далее без изменений относительно описания метода, приведенного в главе 5, на основе первообразоной

$$P_{(\text{int})(n+1)}(t) = u(t_b) + \int_0^{(t-t_b)h^{-1}} P_n(t)\,dt \text{ равной } u(t_b) + h \sum_{k=0}^{n} a_k / (k+1) \left(\frac{t - t_b}{h} \right)^{k+1}$$

восстанавливается приближение искомого решения, которое итерационно уточняется. Итерации $P_n^{(\ell)}(t) \approx f(t, P_{(\text{int})(n+1)}^{(\ell-1)}(t))$, $P_{(\text{int})(n+1)}^{(\ell)}(t) = u(t_b) + \int_0^{(t-t_b)h^{-1}} P_n^{(\ell)}(t)\,dt$,

$\ell = 1, 2, \ldots$, продолжаются до заданной границы $\ell \le q = \text{const}$, которая является параметром метода.

Следует также отметить важную особенность применения кусочно-интерполяционного метода, которая относится именно к решению задачи (6.1) – (6.6). При кусочно-интерполяционном приближении решения рассматриваемой задачи большая часть числа обращений к функциям правой части (6.1) производится в процессе выполнения итераций. При этом моменты времени, определяемые в качестве узлов интерполяции остаются неизменными в процессе итераций. В (6.1) большое число параметров, в частности направляющие косинусы и удаление возмущающего тела, с использованием значений которых определяются ускорения от лунно-солнечных гравитационных возмущений, зависит только от времени T в юлианских столетиях по 36525 эфемеридных суток от эпохи 2000 года, 1 января, 12 ч (UTC(SU)) до момента времени t. Значения данных параметров рассчитываются до начала процесса итерационного уточнения, что дополнительно снижает время построения требуемых приближений эфемерид НКА ГЛОНАСС. Аналогичный подход к снижению времени расчета при численном интегрировании системы (6.1) на основе явных методов Рунге-Кутты не применим вследствие особенностей вычислительного алгоритма разностных методов (обращение к функции правой части дифференциальной системы выполняется при различных значениях независимой переменной на каждом шаге метода).

Точность приближения, согласно методике [37, 195], рассчитывается путем обратного интегрирования задачи (6.1) – (6.6) на отрезке $[t_i,\ t_b]$ или при $t_i < t_b$ на отрезке $[t_b,\ t_i]$ с начальными данными, полученными при решении прямой задачи в момент времени t_i. По данной методике для кусочно-интерполяционного приближения получены «нулевые» в формате вывода данных *extended* значения абсолютной погрешности приближения компонентов системы:

$$\Delta x = 0.000\text{e}+00, \quad \Delta y = 0.000\text{e}+00, \quad \Delta z = 0.000\text{e}+00,$$
$$\Delta \dot{x} = 0.000\text{e}+00, \quad \Delta \dot{y} = 0.000\text{e}+00, \quad \Delta \dot{z} = 0.000\text{e}+00.$$

Точность кусочно-интерполяционного приближения задачи (6.1) – (6.6) совпала с точностью представления начальных данных (6.6), при этом время решения составило 0.224 *мс*. Здесь и ниже погрешность приближения координат НКА указана в метрах, составляющих вектора скорости центра масс НКА – в метрах в секунду, значение времени решения по аналогии с [203] приводится среднее для 100000 последовательных решений на компьютере *Intel Core i5-2500*.

Разностное приближение решения этой же задачи с помощью метода Рунге-Кутты 4-го порядка с шагом интегрирования $h = 1c$ также характеризуется «нулевыми» значениями погрешности вследствие ограниченной точности начальных данных (6.6), однако время решения существенно возрастает – 36.072 *мс*.

В табл. 6.1 представлены результаты численных экспериментов по расчету эфемерид НКА ГЛОНАСС №730 на основе численного решения задачи (6.1), (6.6) с помощью метода Рунге-Кутты 4-го порядка с величиной шага интегрирования $h = 1c$ и кусочно-интерполяционного метода со значениями параметров $n = 8$, $\ell = 12$. В качестве погрешности указана каноническая норма абсолютной погрешности приближения компонентов системы (6.1).

Таблица 6.1

Погрешность и время приближенного решения задачи (6.1), (6.6) с высокой точностью приближения

Δt	*Runge-Kutta_4*		*FPI*	
	$h = 1c$		$n = 8$, $\ell = 12$	
5 мин	0.000**e+00**	12.143 *мс*	0.000**e+00**	0.224 *мс*
10 мин	0.000**e+00**	24.121 *мс*	0.000**e+00**	
15 мин	0.000**e+00**	36.072 *мс*	0.000**e+00**	

Применение предложенного метода позволяет существенно ускорить процесс расчета координат и составляющих вектора скорости центра масс НКА (более чем в 54 раза) при условии достижения высокой точности приближения.

С целью снижения трудоемкости в [200, 203] рекомендуется в практических расчетах применять метод Рунге-Кутты 4-го порядка с величиной шага интегрирования $60\,c$. Для рассматриваемой задачи это позволит выполить ресчет эфемерид НКА ГЛОНАСС №730 на момент времени $t_i = 12600$ со следующими значениями погрешности

$$\Delta x = 3.959\text{e}-05,\quad \Delta y = 1.003\text{e}-04,\quad \Delta z = 1.925\text{e}-04,$$
$$\Delta \dot{x} = 3.704\text{e}-08,\quad \Delta \dot{y} = 1.367\text{e}-09,\quad \Delta \dot{z} = 6.953\text{e}-09,$$

время решения при этом составит $0.522\,мс$. Здесь и ниже погрешность приближения рассчитывается сравнением с эталонным высокоточным приближением решения задачи (6.1) – (6.6), рассчитанным с применением кусочно-интерполяционного метода с параметрами $n = 8$, $\ell = 12$.

Трудоемкость построения кусочно-интерполяционного приближения также можно снизить изменив параметры метода. При значениях параметров: $n = 5$, $\ell = 7$, приближение решения задачи (6.1) – (6.6) характеризуются значениями погрешности:

$$\Delta x = 3.111\text{e}-06,\quad \Delta y \approx 4.055\text{e}-06,\quad \Delta z \approx 7.271\text{e}-06$$
$$\Delta \dot{x} = 3.775\text{e}-09,\quad \Delta \dot{y} \approx 1.635\text{e}-10,\quad \Delta \dot{z} \approx 1.503\text{e}-09$$

при времени решения $0.080\,мс$.

В табл. 6.2 представлены результаты численных экспериментов по расчету эфемерид НКА ГЛОНАСС №730 на основе численного решения задачи (6.1) – (6.6) с помощью метода Рунге-Кутты 4-го порядка с рекомендованной в [200, 203] величиной шага интегрирования ($h = 1\,мин$) и кусочно-интерполяционного метода с соответствующими по достигаемой точности приближения значениями параметров.

Таблица 6.2

Погрешность и время приближенного решения задачи (6.1) – (6.6) с рекомендованными значениями параметров методов

Δt	*Runge-Kutta_4* $h = 60c$		*FPI* $n = 5,\ \ell = 7$	
5 мин	6.441**e-05**	0.175 *мс*	3.574**e-09**	0.080 *мс*
10 мин	1.286**e-04**	0.347 *мс*	4.055**e-07**	
15 мин	1.925**e-04**	0.522 *мс*	7.271**e-06**	

Применение кусочно-интерполяционного метода с параметрами $n = 5$, $\ell = 7$, для всех указанных в табл. 6.2 интервалов интегрирования характеризуется существенным снижением времени решения (54% и более) при повышении точности приближения на 2 – 4 десятичных порядка. На рис. 6.2 представлены графики изменения погрешностей приближения координат центра масс НКА ГЛОНАСС №730 – $x_0(t_i)$, $y_0(t_i)$, $z_0(t_i)$, при $t_i \in [t_b,\ t_b + \Delta t]$, $\Delta t = 900\ c$, полученных в инерциальной системе координат $OX_0Y_0Z_0$ на основе интегрирования системы (6.1) – (6.6) с применением метода Рунге-Кутты 4-го порядка с $h = 1\,мин$ и кусочно-интерполяционного метода с параметрами $n = 5$, $\ell = 7$.

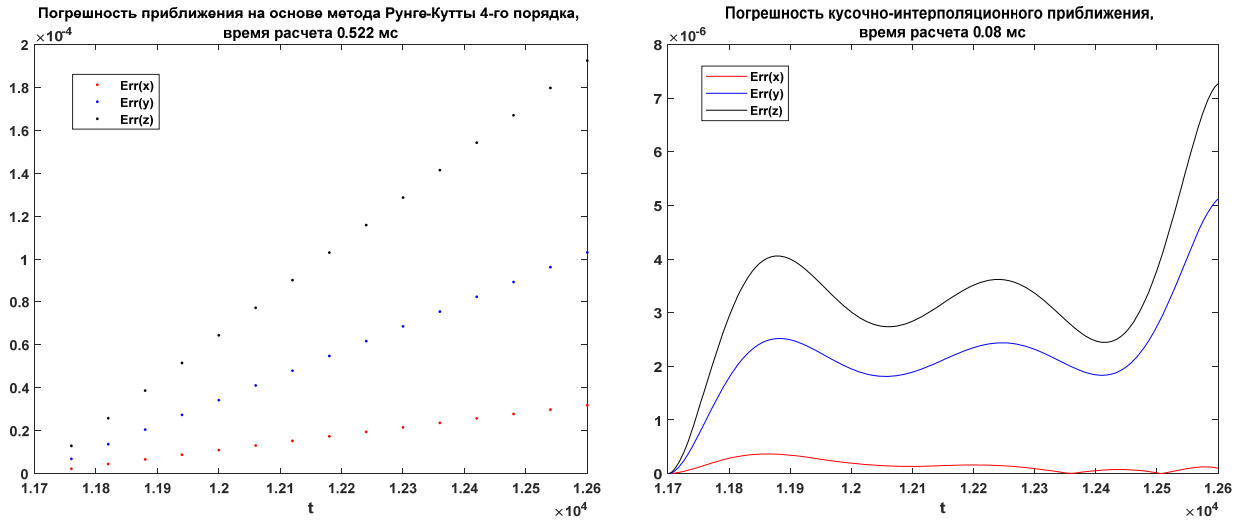

Рис. 6.2. Погрешность приближения координат центра масс НКА ГЛОНАСС №730 на 15-минутном интервале прогнозирования

Из рис. 6.2, в частности, видно, что приближение координат положения НКА с применением метода Рунге-Кутты 4-го порядка может быть получено с

погрешностью, не превышающей 2.0×10^{-4}, за время $0.522\,мс$. Применение кусочно-интерполяционного метода позволяет сократить время расчета координат до $0.08\,мс$ с одновременным повышением точности приближения до 10^{-6}.

С целью сравнения методов при минимально возможном значении времени расчета эфемерид с погрешностью не превышающей 1 см в табл. 6.3 представлены результаты численного эксперимента на основе метода Рунге-Кутты 4-го порядка с шагом интегрирования $h=150\,c$ (наибольшим значением шага, позволяющим получить приближения с указанной точностью) и предложенного метода с соответствующими значениями параметров.

Таблица 6.3

Погрешность и время приближенного решения задачи (6.1) – (6.6) при условии минимизации времени решения

Δt	*Runge-Kutta_4*		*FPI*	
	$h=150\,c$		$n=4,\ \ell=5$	
5 мин	2.526**e-03**	0.071 *мс*	1.847**e-06**	0.049 *мс*
10 мин	5.046**e-03**	0.140 *мс*	2.376**e-04**	
15 мин	7.557**e-03**	0.208 *мс*	4.088**e-03**	

Для интервала интегрирования $5\,мин$ применение предложенного метода ускоряет время расчета на 31% при одновременном повышении точности приближения на 3 десятичных порядка. При величине интервала интегрирования $10\,мин$ время расчета снижается на 65% при снижении погрешности на один десятичный порядок. На интервале $15\,мин$ достигается ускорение процесса расчета значений эфемерид на $\approx 76\%$ с сохранением порядка точности приближения.

Отрезкам приближения $[t_b-\Delta t,\, t_b]$, $\Delta t\in\{300\,c,\ 600\,c,\ 900\,c\}$, соответствуют аналогичные представленным в табл. 6.1 – 6.3 для отрезков $[t_b,\, t_b+\Delta t]$ соотношения погрешности и времени приближения (см. приложение к главе 6, П6.1).

Замечание 6.1. Как было отмечено выше, повышение порядка разностного метода с целью снижения погрешности приближения приводит к повышению времени расчета, в частности, применение метода Рунге-Кутты-Фельберга 5-го порядка повышает время расчета более чем на 50% при незначительном снижении погрешности приближения [203]. Приближение решения задачи (6.1), (6.6) на отрезке $t \in [t_b, t_b + \Delta t]$, $\Delta t = 900\ c$, с помощью метода Дормана-Принса 8-го порядка аппроксимации с шагом $60c$ характеризуется значениями погрешности

$$\Delta x = 1.240\text{e-}08,\ \Delta y = 5.325\text{e}-10,\quad \Delta z = 4.733\text{e-}09,$$
$$\Delta \dot{x} = 2.759\text{e}-11,\quad \Delta \dot{y} = 2.172\text{e-}12,\quad \Delta \dot{z} = 1.060\text{e}-11,$$

время решения 1.746 *мс* (в сравнении с методом Рунге-Кутты 4-го порядка – в 3 раза больше (табл. 6.2)); при значении шага $h = 1c$ точность приближения аналогично результатам, представленным в табл. 6.1, совпадает с точностью начальных данных (6.6):

$$\Delta x = 0.000\text{e}+00,\ \Delta y = 0.000\text{e}+00,\quad \Delta z = 4.548\text{e}-13,$$
$$\Delta \dot{x} = 5.551\text{e}-17,\ \Delta \dot{y} = 1.110\text{e}-16,\quad \Delta \dot{z} = 0.000\text{e}+00.$$

при этом время решения составляет 121.344 *мс*. Листинг программы для расчета координат и составляющих вектора скорости НКА ГЛОНАСС на заданный момент времени по данным эфемерид на основе метода Рунге-Кутты 4 порядка и метода Дормана-Принса 8 порядка представлен в приложении к главе 6, П6.2.

Для удобства прочтения представленные выше табл. 6.1 – 6.3 соединены в одну таблицу сравнения погрешности и времени приближенного решения задачи (6.1), (6.6) при различных значениях параметров методов (табл. 6.4).

Таблица 6.4

Сравнение погрешности и времени приближенного решения задачи (6.1), (6.6) при различных значениях параметров методов

Δt	*Runge-Kutta_4*		*FPI*	
	$h = 1c$		$n = 8,\ \ell = 12$	
5 мин	0.000**e+00**	12.143 *мс*	0.000**e+00**	0.224 *мс*
10 мин	0.000**e+00**	24.121 *мс*	0.000**e+00**	
15 мин	0.000**e+00**	36.072 *мс*	0.000**e+00**	
	$h = 60c$		$n = 5,\ \ell = 7$	
5 мин	6.441**e-05**	0.175 *мс*	3.574**e-09**	0.080 *мс*
10 мин	1.286**e-04**	0.347 *мс*	4.055**e-07**	
15 мин	1.925**e-04**	0.522 *мс*	7.271**e-06**	
	$h = 150c$		$n = 4,\ \ell = 5$	
5 мин	2.526**e-03**	0.071 *мс*	6.071**e-07**	0.049 *мс*
10 мин	5.046**e-03**	0.140 *мс*	1.566**e-04**	
15 мин	7.557**e-03**	0.208 *мс*	4.038**e-03**	

На основе результатов численных экспериментов, в частности, представленных в табл. 6.4, можно сделать вывод о том, что в программном обеспечении навигационной аппаратуры потребителей может быть реализовано в зависимости от требований к точности и быстродействию три варианта расчета координат и составляющих вектора скорости центра масс НКА ГЛОНАСС с применением кусочно-интерполяционного метода: со значениями параметров ($n = 8,\ \ell = 12$), ($n = 5,\ \ell = 7$) и ($n = 4,\ \ell = 5$).

Для большей наглядности результаты дополнительных численных экспериментов представлены на рис. 6.3. в виде диаграммы «точность – время расчета» для приближенного решения задачи (6.1), (6.6) на отрезке $t \in [t_b,\ t_b + \Delta t],\ \ \Delta t = 900\ c$ на основе метода Рунге-Кутты 4-го порядка с различными значениями шага интегрирования и на основе кусочно-интерполяционного метода с различными значениями параметров метода.

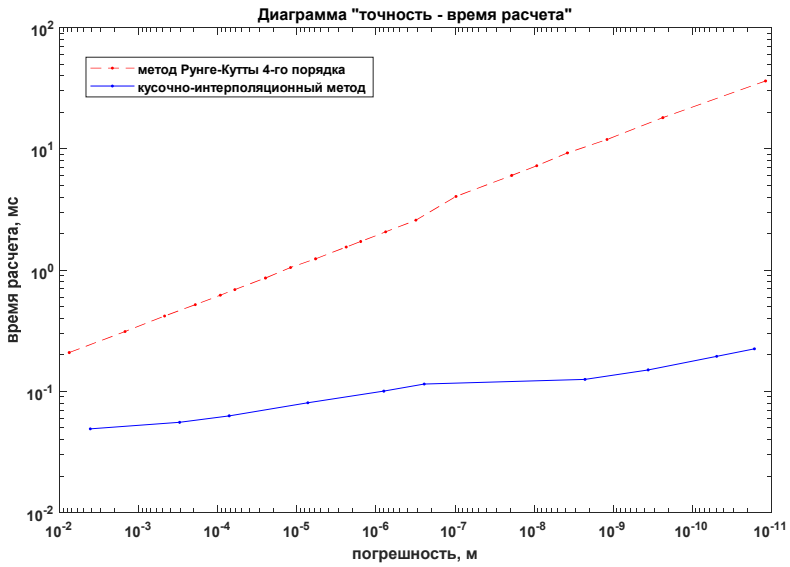

Рис. 6.3. Диаграмма «точность – время расчета» для приближения эфемерид НКА ГЛОНАСС на 15-минутном интервале на основе решения задачи (6.1), (6.6) методом Рунге-Кутты 4-го порядка и кусочно-интерполяционным методом

Из рис. 6.3 видно, что применение кусочно-интерполяционного метода для приближения эфемерид НКА ГЛОНАСС на основе решения задачи (6.1) – (6.6) позволяет достигать требуемых порядков точности за время меньшее на 1 – 2 десятичных порядка в сравнении с применением метода Рунге-Кутты 4-го порядка. В частности, граница абсолютной погрешности порядка 10^{-6} достигается на основе метода Рунге-Кутты за время $\approx 1 мс$, при использовании предложенного метода – за время $\approx 0.1 мс$.

Таким образом, применение кусочно-интерполяционного метода для интегрирования уравнений движения НКА ГЛОНАСС (6.1) позволило существенно ускорить процесс расчета координат и составляющих вектора скорости центра масс НКА при одновременном повышении точности приближения в сравнении с приближением, полученным на основе разностного метода Рунге-Кутты 4-го порядка.

6.4. Сравнительные аспекты идентификации данных модели движения спутника ГЛОНАСС. Численное моделирование уравнений движения (6.1) с применением кусочно-интерполяционного метода позволяет более чем на 54% ускорить процесс расчета координат и составляющих вектора

скорости центра масс НКА ГЛОНАСС на заданный момент времени t_i шкалы МДВ на 30-минутном интервале прогнозирования по данным эфемерид при одновременном повышении точности приближения на 2 – 4 десятичных порядка в сравнении с методом Рунге-Кутты 4-го порядка с рекомендованным в [200, 203] значением величины шага интегрирования. В дополнение к этому применение предложенного метода характеризуется следующими особенностями.

Расчет координат и составляющих вектора скорости центра масс НКА потребителем с момента t_b шкалы МДВ на заданный момент времени t_i той же шкалы ($|t_i - t_b| \le 15$ *мин*) реализован с помощью процедуры *pi_calculation_of_ephemeris_glonass* (см. листинг 6.1) со стандартными параметрами *(N4, NT, tb, ti, y00)*, значения которых задают исходные данные, необходимые для пересчета эфемерид в соответствии с алгоритмом, представленным в интерфейсном контрольном документе ГЛОНАСС: *N4* – номер эфемеридного четырехлетнего периода; *NT* – номер эфемеридных суток в эфемеридном четырехлетнем периоде; момент времени *tb* из оперативной информации ГЛОНАСС; *y00* – вектор координат и составляющих вектора скорости центра масс НКА на момент времени *tb* из оперативной информации ГЛОНАСС; *ti* – заданный момент времени шкалы МДВ, на который необходимо пересчитать координаты и составляющие вектора скорости НКА. Данные значения параметров могут поступать напрямую с датчиков данных в оцифрованном виде без специальной обработки, то есть процесс расчета эфемерид НКА с применением предложенного метода может быть полностью автоматизирован с сохранением высокого быстродействия и требуемой точности расчетов, что не представляется возможным реализовать с помощью известных разностных методов. В частности, возможность построения требуемых приближений в заданный момент времени на основе явных методов Рунге-Кутты напрямую зависит от величины шага интегрирования, а программный код процедуры *pi_calculation_of_ephemeris_glonass* не зависит от

заданного момента времени t_i (значение t_i может быть указано в программе в секундах с любой требуемой точностью). Кроме того, как показано в табл. 6.4, момент времени t_i при использовании предложенного метода также не влияет на время расчета требуемых приближений, которое остается минимальным и фиксированным. При построении разностных приближений время расчета существенно зависит от величины интервала интегрирования (табл. 6.4).

Далее, следует отметить, что в результате кусочно-интерполяционного приближения помимо значений координат и составляющих вектора скорости центра масс НКА в требуемый момент времени t_i шкалы МДВ одновременно выполняется построение непрерывного и непрерывно дифференцируемого приближения компонентов системы (6.1) на интервале $[t_b,\ t_i]$ или $[t_i,\ t_b]$, $\left| t_i - t_b \right| \le 15\ мин$. В отличие от дискретных значений приближения, получаемых при численном интегрировании уравнений движения (6.1) такое построение позволяет выполнить расчет значений координат и составляющих вектора скорости центра масс НКА в произвольно фиксированные моменты времени из отрезка интегрирования. Подобный пересчет нужен, в частности, во время сбоя в работе НКА. Кроме того, свойство непрерывной дифференцируемости приближения компонентов системы (6.1) в отличие от разностных приближений позволяет выполнить расчет ускорений НКА в произвольно фиксированные моменты времени.

Предложенный способ интегрирования уравнений движения (6.1) позволяет также сохранить коэффициенты полиномиальных приближений (в листинге 6.1 зафиксировано значение $k=0$) компонентов системы (6.1) после однократного решения задачи Коши на 30-минутном интервале прогнозирования по данным эфемерид. В дальнейшем расчет координат и составляющих вектора скорости центра масс НКА в произвольный момент времени t_i из данного интервала может выполняться за существенно меньшее время, небходимое только для расчета значений полиномов по схеме Горнера и

пересчета полученных координат центра масс $x_0(t_i)$, $y_0(t_i)$, $z_0(t_i)$ и составляющих его вектора скорости $\dot{x}_0(t_i)$, $\dot{y}_0(t_i)$, $\dot{z}_0(t_i)$ из инерциальной геоцентрической системы координат в связанную с Землей систему ПЗ-90 по формулам (6.5).

Дополнительные возможности ускорения численного моделирования возможны за счет применения кусочной интерполяции суперпозиций функций в правой части (6.1) с помощью интерполяционных полиномов Лагранжа или Ньютона, преобразованных к виду алгебраических полиномов с числовыми коэффициентами, подробно исследованной в главе 5, п. 5.3. Как было отмечено в п. 5.3, метод является инвариантным относительно вида функции и диапазона независимой переменной, характеризуется высокой точностью приближения и одновременно малой временной сложности. В частности, для класса повторяющихся функций после сохранения коэффициентов интерполяционных полиномов в памяти временная сложность сконструированного алгоритма измеряется количеством шагов схемы Горнера, то есть степенью полинома, – $n(t_y + t_c)$, t_y, t_c – время бинарного умножения и сложения. Так, с применением предложенного алгоритма кусочной интерполяции (см. глава 5, п. 5.5) можно построить кусочно-интерполяционные приближения направляющих косинусов и удалений Луны и Солнца – ξ_k, η_k, $\Im_k$, r_k (k – индекс возмущающего тела), которые рассчитываются по формулам (6.3) и позволяют определить ускорения от лунно-солнечных гравитационных возмущений. Данные функции зависят только от времени T в юлианских столетиях от эпохи 2000 года до требуемого момента времени t, при этом для численного моделирования движения НКА системы ГЛОНАСС вплоть до 2047 года диапазон значений T не выходит из отрезка $[0, 0.5]$. Вследствие этого можно априори вычислить коэффициенты интерполяционных полиномов для приближенного вычисления ξ_k, η_k, $\Im_k$, r_k при $T \in [0, 0.5]$, и хранить их в постоянном запоминающем устройстве вычислительной системы, что существенно сократит время вычисления

ускорений от лунных и солнечных гравитационных возмущений. Например, значения функции

$$\xi_л = (\sin \vartheta_л \cos \Gamma' + \cos \vartheta_л \sin \Gamma')\xi_{11} + (\cos \vartheta_л \cos \Gamma' - \sin \vartheta_л \sin \Gamma')\xi_{12},$$

где $\sin \vartheta_k = \dfrac{\sqrt{1-e_k^2}\sin \mathrm{E}_k}{1-e_k \cos \mathrm{E}_k}$, $\mathrm{E}_k = q_k + e_k \sin \mathrm{E}_k$ – уравнение Кеплера для эксцентрической аномалии;

$\xi_{11} = \sin \Omega_л \cos \Omega_л (1-\cos i_л)$, $\xi_{12} = 1 - \sin^2 \Omega_л (1-\cos i_л)$,

$q_л = 2.3555557435 + 8328.6914257190 \times T + 0.0001545547 \times T^2$,

$\Omega_л = 2.1824391966 - 33.7570459536 \times T + 0.0000362262 \times T^2$,

при $T \in [0, 0.5]$ могут быть приближены полиномами Ньютона шестой степени с границей абсолютной погрешности не превышающей порядка 10^{-15} (при разбиении отрезка приближения на 2^{16} подынтервалов равной длины). Программная реализация и результаты численного эксперимента представлены в приложении к главе 6, П6.3. Таким образом, значения направляющих косинусов и удалений Луны и Солнца, необходимые для определения ускорений от лунно-солнечных гравитационных возмущений по формулам (6.2), могут быть вычислены на отрезке $T \in [0, 0.5]$ с точностью не ниже порядка 10^{-15} за время $6(t_y + t_c)$.

Дополнительная минимизация временной сложности при вычислении функций общего вида, как было отмечено в главе 2, может быть достигнута за счет преобразования набора предложенных алгоритмов к параллельной форме (п. 2.1.2), параллелизм кусочно-инетерполяционного решения системы ОДУ с учетом применения к процессу моделирования обсуждается в п. 2.7.

Навигационные космические аппараты ГЛОНАСС, как было описано выше, движутся на околоземной орбите с номинальной высотой 19100 *км* и периодом обращения 11 *ч* 15 *мин* 44 *с*, что позволило создать устойчивую орбитальную систему, не требующую в отличие от остророзонансных орбит

для своего поддержания корректирующих импульсов практически в течение всего срока активного существования, так что движение КА ГЛОНАСС можно охарактеризовать как периодический автоколебательный процесс. Проделанное рассмотрение этого процесса не выводит из границ темы представленной диссертации.

6.5. Об архивации обработанных данных и прогнозировании траектории движения НКА. При использовании кусочно-интерполяционного метода непрерывное приближение решения задачи (6.1) – (6.6) и непрерывное приближение его производной оказываются хранимыми и восстаналиваются в произвольной точке изменения независимой переменной по схеме Горнера вычисления значения полинома с оценкой времени $T = n(t_y + t_c)$ [199]. При этом направление изменения независимой переменной может быть прямым, обратным и, вообще говоря, произвольным. Если требуется пересчитать значение решения или производной в любой точке отрезка приближения, то исходную задачу не требуется целиком решать заново – достаточно обратиться к соответственной строке типизированного файла или массива. Если пересчет требуется выполнить M раз, то рассматриваемая обработка (постобработка) выполнится в $\approx M \times \tau$ раз быстрее (τ – время решения задачи (6.1) – (6.6), чем с M -кратным использованием повторного численного интегрирования.

Время повторного расчета (в случае его необходимости) эфемерид НКА для произвольного момента времени из интервала прогнозирования $[t_b,\ t_b \pm \Delta t]$, $\Delta t = 900\,c$, можно существенно снизить посредством отмеченной возможности хранения коэффициентов непрерывного кусочно-интерполяционного приближения решения задачи (6.1) – (6.6). После однократного построения приближения в последующем оно восстанавливается как значение полинома с хранимыми коэффициентами, вычисляемое по схеме Горнера.

Для формирования типизированного файла с коэффициентами полиномов, аппроксимирующих координаты центра масс НКА на интервале

прогнозирования, в программе, представленной в листинге 6.1, нужно раскомментировать соответствующие блоки операторов. Далее, восстановление значений эфемерид НКА для произвольного момента времени из интервала прогнозирования и их пересчет из инерциальной геоцентрической системы координат в систему ПЗ-90.11 в соответствии с документацией из [200] выполняется представленной ниже программой:

```
program calc_fpi; {$APPTYPE CONSOLE} uses SysUtils, Math;
const N4=7; {номер эфемеридного четырехлетнего периода}
NT=583; {номер эфемеридных суток в эфемеридном четырехлетнем периоде}
tb=11700; ti=12600; {границы интервала прогнозирования эфемерид НКА}
tpr=12600; {момент времени из интервала прогнозирования}
FILEPATH='D:\coeffs.bin';  n=9;  h=112.5;  {файл  с  коэффициентами  полиномов,
аппроксимирующих координаты центра масс НКА и параметры приближения}
type Arr_coeff=array[0..29] of extended;
var f_coeff:file of Arr_coeff;dd:Arr_coeff; const w3=7.2921151467e-5;
type vectNeq = array[1..6] of extended; var i:integer; yn0,yn:vectNeq;
JD0,ERA,GMST,dT,S:extended;
function calc(num:byte;tpr:extended):extended;
type Coeff=array[0..n] of extended; var d:Coeff; j:integer; t:extended;
function gorner(const d:Coeff): extended; var pp:extended; i:integer;
 begin pp:=d[n]; for i:=1 to n do pp:=pp*t+d[n-i]; gorner:=pp; end;
function gorner_pr (const d:Coeff): extended;  var pp:extended; i:integer;
begin pp:=d[n]*n/h; for i:=1 to n-1 do pp:=pp*t+d[n-i]*(n-i)/h;
 gorner_pr:=pp; end;
begin  t:=(tpr-tb)/h;  case  num  of    1,2,3:  begin  for  j:=0  to  n  do
d[j]:=dd[j+(n+1)*(num-1)]; calc:=gorner(d); end;
4,5,6: begin for j:=0 to n do d[j]:=dd[j+(n+1)*(num-3-1)]; calc:=gorner_pr(d);
end; end; end;
begin JD0:=1461*(N4-1)+NT+2450082.5; dT:=(JD0-2451545.0)/36525;
ERA:=2*PI*(0.7790572732640+1.00273781191135448*(JD0-2451545.0));
GMST:=ERA+7.03270726e-8+0.0223603658710194*dT+6.7465784654e-6*power(dT,2)
-2.1332e-12*power(dT,3)-1.452308e-10*power(dT,4)-1.784e-13*power(dT,5);
AssignFile(f_coeff, FILEPATH); Reset(f_coeff); read(f_coeff,dd);
for i := 1 to 6 do yn0[i]:=calc(i,tpr);
S:=GMST+w3*(tpr-10800); yn[1]:=yn0[1]*cos(S)+yn0[2]*sin(S);
yn[2]:=-yn0[1]*sin(S)+yn0[2]*cos(S);yn[3]:=yn0[3];
yn[4]:=yn0[4]*cos(S)+yn0[5]*sin(S)+w3*yn[2];
yn[5]:=-yn0[4]*sin(S)+yn0[5]*cos(S)-w3*yn[1]; yn[6]:=yn0[6];
CloseFile(f_Coeff); writeln('tpr=',tpr);
writeln(' x=',yn[1],'   y=',yn[2],'   z=',yn[3]);
writeln('vx=',yn[4],'  vy=',yn[5],'  vz=',yn[6]); readln;end.
```

Результат работы программы:

```
  tpr=12600
  x= 2.39489258119706E+0007   y= 3.40159756877465E+0005   z=-8.79710015725756E+0006
 vx=-1.21004870882318E+0003  vy= 6.13653373754929E+0001  vz=-3.29014462102794E+0003
```

Метод позволяет хранить лишь коэффициенты полиномов, аппроксимирующих координаты центра масс НКА: значения составляющих вектора скорости восстанавливаются без потери точности как значения производных от полиномов, аппроксимирующих координаты центра масс (процедура `gorner_pr`).

В табл. 6.5 представлены результаты численных экспериментов по расчету эфемерид НКА ГЛОНАСС №730 на основе численного решения задачи (1) – (5) с помощью метода Рунге-Кутты 4-го порядка с шагом $60c$ и на основе восстановления кусочно-интерполяционного приближения решения этой задачи из типизированного файла.

Таблица 6.5

Погрешность и время повторного расчета эфемерид НКА с учетом хранимости коэффициентов кусочной интерполяции

Δt	*Runge-Kutta_4* $h = 60c$		*FPI (calc_fpi)* $n = 8,\ q = 12$	
5 мин	6.441**e-05**	0.175 *мс*	0.000**e+00**	0.025 *мс*
10 мин	1.286**e-04**	0.347 *мс*	0.000**e+00**	
15 мин	1.925**e-04**	0.522 *мс*	0.000**e+00**	

Согласно табл. 6.5 применение хранимого кусочно-интерполяционного приближения решения задачи (6.1) – (6.6) на интервале прогнозирования снижает время восстановления решения примерно в 7 раз для $\Delta t = 5 мин$, в 12 раз для $\Delta t = 10 мин$, в 20 раз для $\Delta t = 15 мин$. При этом точность восстановленного приближения существенно превышает допустимое значение.

Замечание 6.2. При расчете эфемерид НКА ГЛОНАСС на основе восстановления кусочно-интерполяционного приближения решения задачи Коши из типизированного файла основной составляющей времени расчета (табл. 6.5) является время отрытия и закрытия типизированного файла.

Замечание 6.3. Без учета времени отрытия и закрытия типизированного файла расчет эфемерид НКА ГЛОНАСС для произвольного момента времени из интервала прогнозирования занимает время $0.0007 мс$.

Согласно изложенному с помощью кусочно-интерполяционного интегрирования уравнений движения центра масс НКА достигается снижение погрешности прогнозирования положения НКА при одновременном ускорении вычислительной обработки. Кроме того, в результате кусочно-интерполяционного решения задачи (6.1) – (6.6) достигается непрерывность приближения координат, а также составляющих вектора скорости центра масс

НКА на интервале прогнозирования. Эти приближения могут быть хранимыми для всего интервала прогнозирования в виде типизированных файлов, что позволяет выполнять пересчет положения НКА в любых точках интервала без повторного решения задачи (6.1) – (6.6).

Целесообразно пояснить значение предложенной разновидности кусочной интерполяции решения задачи (6.1) – (6.6) для архивации прогнозирования. Поскольку отрезок $[t_b,\ t_b + \Delta t]$ $([t_b - \Delta t, t_b])$, $\Delta t = 900\ c$ остается постоянным для численного решения этой задачи, то при любом выборе разбиения для $[a, b] = [t_b - \Delta t,\ t_b + \Delta t]$ двумерный массив коэффициентов рассматриваемой кусочной интерполяции будет постоянным на всем этом отрезке. Необходима оговорка, что речь идет о решении задачи Коши с фиксированными начальными значениями на 30-минутном интервале прогнозирования. При изменении начальных значений соответственно изменится массив коэффициентов. При фиксированных начальных значениях задачи (6.1) – (6.6) приближенное решение со всеми отмеченными качествами с единственностью будет представлено одним и только одним двумерным массивом коэффициентов, сформированным в процессе прогнозирования путем численного моделирования. Такой массив естественно хранить в виде типизированного файла. С учетом того, что каждый компонент решения (и его первых двух производных) восстанавливается по строке коэффициентов, соответствующей номеру подынтервала, для произвольной точки $t \in [a, b]$, получается удобный вид хранения всего процесса прогнозирования на данном отрезке. При этом аналитические качества восстановленного приближения сохраняются и практически дают всю информацию о поведении решения (о поведении центра масс НКА) в любой момент времени на данном интервале прогнозирования. Так, для представленного в программах `fpi_glonass` и `calc_fpi` примера рассматриваемый массив состоит всего из одной строки, которая содержит набор из 10 коэффициентов для каждого интерполируемого компонента решения задачи (6.1) – (6.6). И именно такой массив

(типизированный файл) включает всю требуемую информацию для последующей обработки архива. Для нескольких соседних интервалов прогнозирования понадобится число хранимых файлов, равное числу интервалов, однако, их можно объединить в один файл с сохранением его структуры: начальные данные для каждого интервала изменят значение коэффициентов, но не повлияют на их взаимное расположение и на способ их считывания.

При запоминании коэффициентов интерполирующих полиномов в структуре типизированного файла метод позволяет не только рассчитать требуемые выходные данные для интервала прогнозирования, но и реализует возможность восстановления из файла, практически без потери времени, данных для прогноза в любой точке интервала прогнозирования. Восстановление возможно также на произвольном множестве точек интервала. Последовательное восстановление требуемых значений в каждой точке каждого интервала восстанавливает весь отрезок линии движения НКА на интервале прогнозирования (учитывается непрерывность кусочно-интерполяционного приближения). Отсюда возникают предпосылки прогнозирования в заданное время не только одной точки местоположения НКА (объекта), но возможно прогнозировать полный отрезок траектории движения объекта. Вместе с прогнозированием отрезка траектории НКА (объекта) предсказывается скорость движения (производная интерполирующего полинома), а также ускорение (его вторая производная).

Предлагаемый способ архивации позволит повысить эффективность и точность решения пользователями координатно-временных задач в апостериорном режиме моделирования движения НКА по сравнению с традиционным применением апостериорной высокоточной эфемеридно-временной информации в дискретных узловых точках [205]. В [205] рекомендуется выполнять расчет движения центра масс НКА на любой заданный момент времени по рассчитанным узловым точкам с использованием

интерполяционного полинома, а расчет текущей скорости движения НКА путем дифференцирования такого полинома. Это влечет накопление погрешности интерполяции и дифференцирования, что исключается в случае применения предложенного способа архивации. Передача потребителям вместо узловых точек со значениями эфемерид НКА массива с коэффициентами непрерывного кусочно-полиномиального приближения траектории (и скорости), которое хранит высокоточные значения приближенного решения в каждой точке интервала прогнозирования, поэтому не требует интерполяции и какой-либо дополнительной вычислительной обработки, позволит существенно ускорить и повысить точность потребительского расчета рассматриваемых эфемерид.

Содержание предложенного подхода соотносится с тем, что аналогичный подход с целью высокоточной быстродействующей обработки применяется в программе ORBGEN, входящей в состав Bernese GNSS (высокоточного программного обеспечения для постобработки наблюдений глобальных навигационных спутниковых систем, разработанного в астрономическом институте Бернского университета) [206]. Приближенное решение уравнений движения НКА строится в виде полинома, приближение сохраняется в форме файла с коэффициентами аппроксимирующих полиномов. Постобработка выполняется на основе метода коллокации (разновидность неявного метода Рунге-Кутты), что, в отличие от предложенного подхода, ограничивает снижение погрешности и временной сложности.

Таким образом, в главе представлен комплекс программ обработки данных эфемерид на модели движения НКА ГЛОНАСС и выполнено моделирование движения НКА по оперативной эфемеридной информации из навигационного сообщения. Кусочно-интерполяционное приближение координат непрерывно, сходится к траектории НКА ГЛОНАСС и приближает скорость НКА на всем интервале прогнозирования. Минимизация временной сложности предложенного метода позволила более чем на 54% ускорить

процесс расчета координат и составляющих вектора скорости центра масс НКА ГЛОНАСС в произвольно заданный момент времени из 30-минутного интервала прогнозирования по данным эфемерид в сравнении со стандартным алгоритмом расчета с превышением границ требуемой точности. Как отмечалось, процесс расчета реализован в виде процедуры с параметрами определяемыми из навигационного сообщения и может быть полностью автоматизирован с сохранением быстродействия, точности и гладкости обработки данных. На каждом подынтервале интервала прогнозирования интерполирующий компонент траектории полином является алгебраическим полиномом фиксированной степени с числовыми коэффициентами. Это дает возможность сохранять в памяти компьютера приближение траектории как типизированный файл коэффициентов соответственных подынтервалам разбиения. Непрерывное и гладкое приближение траектории оказывается возможным хранить и восстанавливать – без повторного вычисления траектории – в произвольной точке интервала прогнозирования простым алгоритмом считывания как соответственное значение полинома. Процесс прогнозирования на этой основе достаточно просто архивируется. Возникают предпосылки прогнозирования полного отрезка траектории движения НКА с расширением интервала прогнозирования. Представлены результаты численного эксперимента, подтверждающие улучшение качества прогнозирования параметров движения НКА. Полученные в главе результаты актуальны для повышения точности и скорости определения координат местоположения и составляющих вектора скорости движения потребителя системы ГЛОНАСС.

6.6. Выводы

1. Разработан комплекс программ кусочно-интерполяционной обработки данных эфемерид на модели движения НКА ГЛОНАСС. Точность и гладкость предложенного метода обработки данных позволяют получить координаты и составляющие вектора скорости центра масс НКА с превышением границ требуемой точности в произвольно заданные моменты времени из 30-минутного интервала прогнозирования, при этом время расчета примерно вдвое меньше времени известных методов.

2. Процесс расчета координат и составляющих вектора скорости центра масс НКА потребителем на заданный момент времени шкалы МДВ реализован в виде процедуры со стандартными параметрами, значения которых определяются по оперативной эфемеридной информации из навигационного сообщения. На этой основе процесс расчета эфемерид НКА с применением предложенного метода может быть полностью автоматизирован с сохранением быстродействия, точности и гладкости обработки данных.

3. Кусочно-интерполяционное приближение координат непрерывно, сходится к траектории НКА ГЛОНАСС и приближает скорость НКА на всем интервале прогнозирования. На каждом подынтервале интервала прогнозирования интерполирующий компонент траектории полином имеет вид алгебраического полинома фиксированной степени с числовыми коэффициентами. Это позволяет сохранять в памяти компьютера приближение компонента траектории в виде типизированного файла коэффициентов полиномов соответственных подынтервалам разбиения. Непрерывное и гладкое приближение каждого компонента траектории оказывается хранимым и восстанавливается без повторного вычисления траектории в произвольной точке интервала прогнозирования простым алгоритмом считывания как соответственное значение полинома, что качественно отличается от известных методов. На данной основе процесс прогнозирования естественным образом архивируется. Создаются предпосылки прогнозирования не отдельных точек

траектории, а всего непрерывного отрезка траектории движения НКА с расширением интервала прогнозирования. Приводится программа обработки данных и результаты численного эксперимента, подтверждающие улучшение качества прогнозирования параметров движения НКА.

4. Предложен способ дополнительного ускорения обработки данных на модели движения НКА ГЛОНАСС за счет применения кусочной интерполяции суперпозиций функций в правой части уравнений движения с помощью интерполяционных полиномов Ньютона, преобразованных к виду алгебраических полиномов с числовыми коэффициентами. На этой основе временная сложность вычисления значений направляющих косинусов и удалений Луны и Солнца, необходимых для определения ускорений от лунно-солнечных гравитационных возмущений, измеряется числом шагов схемы Горнера, что интерпретируется как оценка времени единичного порядка.

ЗАКЛЮЧЕНИЕ

В рамках диссертационной работы были проведены исследования, результатом которых стала разработка высокоточных быстродействующих алгоритмов и программ обработки данных в ИВС для моделей периодических процессов, позволяющих повышать качество моделирования, уточнять физико-технологические и динамические характеристики моделей процессов в реальном времени.

В частности, следующие результаты отличаются новизной:

1. Предложен метод разностно-полиномиальной обработки данных в ИВС с программным выбором варьируемых параметров для моделей периодических процессов. Метод отличается от аналогов обработкой данных на временных подынтервалах интерполяционными полиномами Ньютона, программно преобразуемыми в форму алгебраических полиномов с числовыми коэффициентами, разностной обработкой узловых значений, применением итерационного уточнения, автоматизированным выбором параметров, что позволяет повысить точность и уменьшить время обработки данных относительно известных методов, а также улучшить качество моделирования исследуемых процессов.

2. Выполнено исследование, показана сходимость, дана оценка скорости сходимости предложенного метода. Для корректного применения метода достаточно двукратной дифференцируемости правой части дифференциальной системы, что положительно отличается от условий применения аналогов и улучшает качество моделирования в ИВС периодических процессов с быстро меняющейся динамикой.

3. Как развитие метода обработки данных в ИВС с автоматизированным выбором варьируемых параметров и итерационным уточнением для моделей периодических процессов предложена модификация, не использующая разностные схемы. Модифицированный метод отличается от аналогов

кусочной интерполяцией на временных подынтервалах правой части дифференциальной системы и интегральным приближением решения в виде алгебраических полиномов с числовыми коэффициентами, а также выполнением итераций на подынтервалах, аналогичных интегральным приближениям Пикара. Метод отличается повышением точности обработки данных на больших отрезках времени относительно известных методов, а также улучшением качества численного моделирования.

4. Показана сходимость предложенной модификации кусочно-интерполяционного метода обработки данных и сходимость итерационного уточнения, даны оценки скорости сходимости. Качества равномерной сходимости и аналитический вид приближения числовых данных отличают предложенную модификацию от известных аналогов.

5. Предложенный метод реализован в виде стандартной программы ИВС, на вход которой поступает правая часть дифференциальной системы, моделирующей процесс, начальные данные и параметры, включающие границы временного отрезка обработки данных модели. Автоматический выбор параметров обеспечивает наибольшую точность при наименьшем времени обработки. Это отличительное качество программной реализации позволяет превышать точность аналогов на два десятичных порядка, при этом варьируемые параметры программно адаптируются к структуре модели и реализуют динамическую коррекцию начальных данных. В результате достигается вычислительная устойчивость, высокая точность, гладкость аналитического приближения данных на отрезке произвольной длины, что составляет отличие метода для широкого класса моделей периодических процессов в ИВС, в числе которых модели жестких задач и задач с неустойчивыми по Ляпунову решениями.

6. Предложен аналог метода обработки данных в ИВС для моделей с частными производными. Более точно, разработана варьируемая кусочно-интерполяционная обработка числовых данных модели переноса, которая

отличается от известных применением интерполяционного полинома Ньютона от двух переменных, программно преобразуемого в алгебраический полином с числовыми коэффициентами. Варьируемая интерполяция данных выполняется в каждой прямоугольной подобласти, на которые в зависимости от значений параметров автоматически делится исходная прямоугольная область. Применяется итерационное уточнение обработанных данных. На этой основе получается гладкое приближение данных в прямоугольной подобласти, которое отличает предложенный метод от известных, кроме того, метод отличается значительно более высокой точностью. Согласно численному эксперименту достигается превышение точности известных методов моделирования переноса на несколько десятичных порядков, что позволяет улучшить качество моделирования волновых процессов в ИВС.

7. Показана сходимость кусочно-интерполяционной обработки данных модели переноса, даны оценки скорости сходимости. Для случая прямоугольной области показана сходимость и оценивается скорость сходимости итерационного уточнения. Для квазилинейной модели переноса предложена схема вывода аналогичных оценок.

8. Выполнена алгоритмизация и программная реализация обработки данных модели переноса, реализован автоматический выбор параметров, обеспечивающий наибольшую точность при наименьшем времени обработки. Программная реализация отличается устойчивостью и отмеченной точностью обработки, что в сочетании с кусочной гладкостью приближения данных позволяет детализировать и повысить качество моделирования процесса переноса.

9. Предложен метод создания библиотеки стандартных программ в ИВС на основе кусочно-интерполяционной обработки данных. Разновидность кусочной интерполяции позволяет приближать функции с точностью до 10^{-20} на произвольном временном отрезке, при этом стандартные и специальные функции приближаются за время единичного порядка, что положительно

отличает метод от известных. Реализованы варианты хранения коэффициентов в разделе констант программы, в типизированном файле, в постоянном запоминающем устройстве, дан алгоритм считывания. Вычисления функции взаимно независимы по значениям аргумента, что влечет параллелизм метода. На основе хранимых коэффициентов любая функция библиотеки может параллельно воспроизводиться на произвольном множестве точек фиксированной области. На аналогичной основе разработано приближение производных в ИВС с точностью до 10^{-16}.

10. Показана возможность существенного снижения трудоемкости без потери точности в предложенном методе обработки данных дифференциальной модели при замене автоматического выбора параметров их пользовательским подбором и фиксированием. Согласно численному и программному эксперименту предложенная модификация превосходит известные методы по быстродействию, при этом достигает более высокой точности обработки данных. На этой основе разработаны быстродействующие алгоритмы воспроизведения с помощью хранимых коэффициентов специальных и стандартных функций с гладкостью приближения. В частности, функция Бесселя и гипергеометрическая функция, применяемые в моделях периодических процессов, реализуются гладким приближением на больших временных промежутках с быстродействием и точностью, превосходящими характеристики известных аналогов.

11. Разработан комплекс программ обработки данных в ИВС на основе предложенного метода для моделей жестких и нежестких задач, включающий автоматический и пользовательский выбор параметров для адаптации к классам моделей. С помощью комплекса выполнено моделирование периодических автоколебательных реакций (реакции Белоусова-Жаботинского, релаксационных автоколебаний в системе гликолиза, гетерогенной колебательной реакции окисления молекулярного водорода). Результаты отличаются сравнительно высокой точностью и гладкостью обработки данных

моделей в допустимых границах трудоемкости, что позволяет уточнить физико-химические параметры автоколебаний и повысить качество моделирования. Комплекс включает программы моделирования процессов переноса, согласно эксперименту точность обработки данных с помощью этих программ по сравнению с аналогами повышается в среднем на восемь десятичных порядков, отличается на порядок меньшим временем обработки и гладкостью приближения, что позволяет уточнить форму и динамику перемещения волн.

12. С помощью разработанного комплекса программ выполнено моделирование в ИВС движения КА, рассчитано уточненное время необходимое для вывода КА на устойчивую периодическую орбиту при управлении, соответствующем внешнему воздействию гравитационных сил. Получены уточненные параметры орбиты устойчивого движения КА. Достаточно малая временная сложность предложенной обработки данных позволяет применять метод для управления движением КА в режиме реального времени. Выполнено моделирование возмущенного движения ИСЗ, выведенного на низкую околоземную орбиту. Сравнительно высокая точность предложенного метода обработки данных в сочетании с гладкостью и малой временной сложностью дают координаты ИСЗ с требуемой точностью в произвольный момент времени, что необходимо для измерений с помощью лазерного дальномера от пункта наблюдения до ИСЗ и позволяет предупреждать аварийный сход с орбиты.

13. С помощью разработанного программного комплекса выполнена обработка данных эфемерид на модели движения НКА ГЛОНАСС. Точность и гладкость предложенного метода обработки данных позволяют получить координаты и составляющие вектора скорости центра масс НКА с превышением границ требуемой точности в произвольно заданные моменты времени из 30-минутного интервала прогнозирования, при этом время расчета примерно вдвое меньше времени известных методов. Процесс расчета реализован в виде процедуры с параметрами определяемыми из

навигационного сообщения и может быть полностью автоматизирован с сохранением быстродействия, точности и гладкости обработки данных. Предложен способ дополнительного ускорения обработки данных на модели движения НКА ГЛОНАСС за счет хранения коэффициентов кусочной интерполяции траектории и динамики движения для задач постобработки данных.

14. Кусочно-интерполяционное приближение координат непрерывно, сходится к траектории НКА ГЛОНАСС и приближает скорость НКА на всем интервале прогнозирования. На каждом подынтервале интервала прогнозирования интерполирующий компонент траектории полином является алгебраическим полиномом фиксированной степени с числовыми коэффициентами. Это дает возможность сохранять в памяти компьютера приближение траектории как типизированный файл коэффициентов соответственных подынтервалам разбиения. Непрерывное и гладкое приближение траектории оказывается возможным хранить и восстанавливать – без повторного вычисления траектории – в произвольной точке интервала прогнозирования простым алгоритмом считывания как соответственное значение полинома, что качественно отличается от известных методов. Процесс прогнозирования на этой основе достаточно просто архивируется. Возникают предпосылки прогнозирования полного отрезка траектории движения НКА с расширением интервала прогнозирования. Приводится программа обработки данных и результаты численного эксперимента, подтверждающие улучшение качества прогнозирования параметров движения НКА.

15. Обработка данных использует многообразие методов, в числе которых вычисление интеграла. Предложен метод обработки интегральных данных, представляющий собой разновидность формул Ньютона-Котеса с коэффициентами, не зависящими от подынтегральной функции и промежутка интегрирования, полученными на основе кусочной интерполяции. Вычисление интегралов с хранимыми в памяти ИВС коэффициентами реализовано

программно. Пользовательский интерфейс программ стандартизируется до задания подынтегральной функции, промежутка интегрирования, как вариант может включать степень полинома и число подынтервалов. Показана сходимость метода, даны оценки скорости сходимости. Интегральная обработка данных реализуется с высокой точностью на временных интервалах большой длины. На промежутках стандартной для обработки данных длины достигается нулевая граница погрешности. Метод распространяется на приближение первообразной функции, в этом случае погрешность имеет порядок 10^{-19}. Одновременно с минимизацией погрешности минимизируется время вычисления интегралов и первообразных, что отличает метод от известных.

Работа включает следующие **научные результаты**.

1. Предложен метод разностно-полиномиальной обработки данных в ИВС с программным выбором варьируемых параметров для моделей периодических процессов, построенный на основе обработки данных на временных подынтервалах интерполяционными полиномами Ньютона, программно преобразуемыми в форму алгебраических полиномов с числовыми коэффициентами. Метод использует разностную обработку узловых значений, применяет итерационное уточнение и обеспечивает повышение точности при уменьшении времени обработки данных для улучшения качества моделирования исследуемых процессов.

2. Даны обоснование сходимости и оценки скорости сходимости предложенного метода разностно-полиномиальной обработки данных на произвольном промежутке времени.

3. Предложена модификация метода обработки данных в ИВС с автоматизированным выбором варьируемых параметров для моделей периодических процессов, исключающая использование разностных схем. Модификация построена на основе кусочной интерполяции на временных подынтервалах правой части дифференциальной системы и интегральном

приближении решения в виде алгебраических полиномов с числовыми коэффициентами, использует итерационные уточнения аналогичные интегральным приближениям Пикара. Модификация отличается повышенной точностью обработки данных на больших отрезках времени и позволяет улучшать качество моделирования.

4. Выполнено обоснование сходимости и оценки скорости сходимости предложенной модификации кусочно-интерполяционного метода обработки данных с итерационным уточнением на произвольном временном промежутке.

5. Представлена программная реализация предложенного метода с автоматизированным выбором параметров, адаптирующихся к структуре модели и реализующих динамическую коррекцию начальных данных, что позволяет достигать наибольшей точности гладкого аналитического приближения данных на отрезке произвольной длины при минимальном времени обработки. При этом достигается вычислительная устойчивость обработки данных в ИВС для широкого класса моделей периодических процессов, в числе которых модели жестких задач и задач с неустойчивыми по Ляпунову решениями.

6. Предложен метод варьируемой кусочно-интерполяционной обработки числовых данных модели переноса с итерационным уточнением, построенный на основе интерполяционного полинома Ньютона от двух переменных, программно преобразуемого в алгебраический полином с числовыми коэффициентами. Применяется двумерное итерационное уточнение обработанных данных, что позволяет получить кусочно гладкое аналитическое приближение движения волны в прямоугольной области со сравнительно высокой точностью и повысить качество моделирования процесса переноса в ИВС.

7. Даны обоснование сходимости и оценки скорости сходимости кусочно-интерполяционной обработки данных с итерационным уточнением для модели переноса в прямоугольной области.

8. Выполнена алгоритмизация и программная реализация метода обработки данных модели переноса с автоматизированным выбором параметров, обеспечивающим наибольшую точность при наименьшем времени обработки и вычислительную устойчивость, что в сочетании с кусочной гладкостью приближения данных позволяет детализировать и повысить качество моделирования процесса переноса.

9. Предложен метод создания библиотеки стандартных программ в ИВС на основе кусочно-интерполяционной обработки данных с хранением полиномиальных коэффициентов, позволяющий параллельно воспроизводить приближения стандартных и специальных функций в моделях периодических процессов с точностью порядка 10^{-20} на произвольном множестве точек фиксированной области за время единичного порядка.

10. Представлена модификация кусочно-интерполяционной обработки данных дифференциальной модели, позволяющая существенно повысить быстродействие без потери точности обработки за счет замены автоматического выбора параметров их пользовательским подбором и фиксированием. Разработанные на этой основе с помощью хранимых коэффициентов быстродействующие алгоритмы высокоточного воспроизведения специальных функций и их производных позволяют в моделях периодических процессов получать приближения данных со свойством гладкости на больших интервалах времени.

11. Разработан комплекс программ кусочно-интерполяционной обработки данных в ИВС для моделей жестких и нежестких задач, включающий автоматический и пользовательский выбор параметров для адаптации к классам моделей. Комплекс позволяет выполнять быстродействующее высокоточное моделирование периодических автоколебательных реакций, что необходимо для отладки технологических процессов на их основе. Комплекс также включает моделирование процессов переноса для уточнения форм и динамики перемещения волн.

12. Разработан комплекс программ для моделирования в ИВС движения КА, позволяющий рассчитывать время и параметры вывода КА на устойчивую периодическую орбиту при управлении соответствующем внешнему воздействию гравитационных сил. Сравнительно малая временная сложность предложенной обработки данных позволяет применять комплекс для управления движением КА в режиме реального времени, кроме того программный комплекс позволяет рассчитать с повышенной точностью координаты ИСЗ, выведенного на низкую околоземную орбиту, в произвольный момент времени, что необходимо для измерений с помощью лазерного дальномера от пункта наблюдения до ИСЗ.

13. Представлены алгоритмы и программы для моделирования движения НКА ГЛОНАСС по данным эфемерид, позволяющие существенно ускорить процесс расчета координат и составляющих вектора скорости центра масс НКА ГЛОНАСС с превышением требуемой точности в произвольно заданные моменты времени из 30-минутного интервала прогнозирования. Предложенные алгоритмы и программы позволяют сохранять гладкое аналитическое приближение координат траектории и скорости движения НКА в памяти компьютера и восстанавливать его без повторного вычисления траектории в произвольной точке интервала прогнозирования за время единичного порядка без снижения точности приближений. На данной основе улучшается процесс прогнозирования и его архивация, создаются предпосылки прогнозирования непрерывного отрезка траектории движения НКА и расширения интервала прогнозирования.

14. Предложен метод обработки интегральных данных, представляющий собой разновидность формул Ньютона-Котеса с коэффициентами, не зависящими от подынтегральной функции и промежутка интегрирования, полученными на основе кусочной интерполяции. Вычисление интегралов с хранимыми в памяти ИВС коэффициентами реализовано программно. Пользовательский интерфейс программ стандартизирован до задания

подынтегральной функции, промежутка интегрирования, как вариант может включать степень полинома и число подынтервалов. Интегральная обработка данных реализуется с высокой точностью на временных интервалах большой длины. На промежутках стандартной для обработки данных длины достигается нулевая граница погрешности. Одновременно с минимизацией погрешности минимизируется время вычисления интегралов.

Научную и практическую значимость в области обработки данных в ИВС моделей периодических процессов имеют полученные в диссертации высокоточные быстродействующие методы и программная реализация обработки данных дифференциальных моделей, включая модели периодических автоколебательных реакций, модели процессов переноса и модели прогнозирования движения космических аппаратов. Научную значимость в области численных методов имеют представленные в диссертации кусочно-интерполяционные методы приближенного решения дифференциальных систем, отличающиеся от известных способом построения, инвариантностью вычислительного алгоритма относительно задач различных классов, возможностью достижения высокой точности и гладкости полученного приближения при одновременной минимизации временной сложности. Значимы также представленные в работе методы варьируемого кусочно-интерполяционного вычисления функций, отличающиеся малой погрешностью и гладкостью непрерывного приближения с минимизацией временной сложности, и аналог подхода Ньютона-Котеса для произвольной степени интерполяционного полинома, реализованный на основе кусочно-интерполяционного приближения подынтегральных функций. Помимо того, научно значимым является метод кусочно-интерполяционного решения уравнения переноса на основе интерполяционного полинома Ньютона от функции двух переменных варьируемой степени с итерационным уточнением. Метод позволяет достигать высокой точности и кусочной непрерывности приближения решения линейного и квазилинейного уравнения переноса и уточнить на этой основе параметры волнового процесса в результате обработки

данных модели переноса. Все предложенные методы основаны на инвариантном программном переводе интерполяционных полиномов в форму алгебраических полиномов с числовыми коэффициентами с помощью алгоритма отличного от формул Виета и уравнений Ньютона для симметрических функций.

Практическая значимость диссертационного исследования заключается в прикладном характере предложенных методов обработки данных, включающих кусочно-интерполяционные решения ОДУ и ДУ в частных производных, которые применяются для компьютерной реализации моделей колебательных динамических процессов, включая химические и биохимические реакции (реакция Белоусова-Жаботинского, релаксационные автоколебания в системе гликолиза, колебательная реакция окисления молекулярного водорода), электрическое равновесие автогенератора с внутренней обратной связью, движение КА с управлением при помощи «малой тяги», возмущенное движение КА, прогнозирование положения НКА системы ГЛОНАСС, а также для моделирования переноса волны. Результаты моделирования необходимы для отладки технологических процессов на основе периодических реакций, для эффективного управления движением КА в режиме реального времени, для прогнозирования движения и расчета координат ИСЗ в произвольно заданные моменты времени, повышения точности и скорости расчета пространственных данных об объектах для ГИС. На основе инвариантных кусочно-интерполяционных методов разработан программный комплекс, реализующий методы как с автоматическим, так и с пользовательским подбором параметров. Комплекс программ предназначен для решения актуальных задач обработки данных моделей, связанных с исследованием динамических систем и автоколебательных процессов. В аспекте практического применения существенно сочетание в инвариантных кусочно-интерполяционных методах высокой точности, быстродействия и гладкости приближения, позволяющее применять методы для повышения качества численного моделирования, в том

числе в режиме реального времени. Данное сочетание характеристик сохраняется при численном моделировании процессов, описываемых жесткими и нежесткими системами ОДУ, и в целом характеризует разработанный комплекс программ. Практическую значимость имеет в частности то, что точность и гладкость предложенного метода обработки данных позволяют получить координаты и составляющие вектора скорости центра масс НКА ГЛОНАСС с превышением требуемой точности в произвольно заданные моменты времени из 30-минутного интервала прогнозирования, при этом время расчета примерно вдвое меньше времени известных методов. Кроме того, значима возможность сохранять в памяти компьютера приближение компонента траектории в виде типизированного файла. Хранимым оказывается непрерывное и гладкое приближение каждого компонента траектории, при этом траектория восстанавливается без повторного вычисления в произвольной точке интервала прогнозирования. Процесс прогнозирования естественным образом архивируется, создаются предпосылки прогнозирования всего непрерывного отрезка траектории движения НКА с увеличением существующего интервала прогнозирования.

Внедрение и использование результатов работы. Полученные в работе результаты приняты к использованию:

1. В АО НКБ ВС для моделирования движения и автоматического управления подвижными объектами; для повышения точности численного моделирования автоколебательных процессов при автоматизированном управлении движением объектов; при расширении библиотеки стандартных программ вычисления элементарных, повторяющихся и специальных функций для бортовых вычислителей систем автоматического управления движением подвижными объектами и с целью вычисления композиций сложных функций с минимальной временной сложностью при высокой точности приближения функций. Разработанный программный комплекс используется для численного моделирования процессов управления с широко варьируемыми параметрами с

целью уточнения физико-технологических и динамических характеристик моделируемых процессов.

2. В АО «ВНИИЖТ» для повышения точности прогнозирования положения объекта, а также для моделирования навигационного управления; для расширения библиотеки стандартных программ вычисления элементарных, часто повторяющихся и специальных функций с целью ускорения процесса численного моделирования, а также уточнения значения параметров в моделях дистанционного зондирования с помощью спутников и БПЛА. Разработанный программный комплекс принят к использованию с целью уточнения физико-технологических характеристик моделируемых процессов в режиме реального времени; программный комплекс для численного моделирования движения навигационных космических аппаратов ГЛОНАСС принят к использованию с целью ускорения и повышения точности расчета координат местоположения и скорости движения исследуемых объектов.

3. В учебном процессе кафедры информатики Таганрогского института имени А.П. Чехова (филиала) ФГБОУ ВО «РГЭУ (РИНХ)» в курсах «Специальные разделы информатики», «Современные инструментальные средства», «Программирование», «Компьютерное моделирование», «Алгоритмы численного интегрирования и анализа устойчивости», «Современные технологии программирования», «Абстрактная и компьютерная алгебра», «Численные методы в анализе данных», «Визуализация данных», «Математическое и имитационное моделирование», «Численные методы», «Математическое моделирование и численные эксперименты», «Современные методы построения программ» и «Параллельные алгоритмы».

Кроме того, полученные в работе результаты были положены в основу исследований по проектам, поддержанным Российским фондом фундаментальных исследований: 10-07-00178-а «Численная оптимизация на основе сортировки с приложением к анализу устойчивости, поиску и распознаванию»; 12-07-00143-а «Компьютерные методы численной оптимизации на основе сортировки с приложением к анализу устойчивости,

разностно-полиномиальному решению дифференциальных уравнений, распознаванию изображений и цифровой обработке сигналов»; 16-07-00100-А «Компьютерные методы варьируемого кусочно-полиномиального решения дифференциальных уравнений и анализа устойчивости».

ЛИТЕРАТУРА

1. Suli E. Numerical Solution of Ordinary Differential Equations. – Mathematical Institute, University of Oxford, 2014.– 82 p.
2. Butcher J.C. Numerical Methods for Ordinary Differential Equations. – Chichester. JohnWiley&Sons, Ltd., 2016.–xxiv+519 p.
3. Хайрер Э., Ваннер Г. Решение обыкновенных дифференциальных уравнений. Жесткие и дифференциально-алгебраические задачи. – М.: Мир, 1999. – 685 с.
4. Новиков Е.А., Шорников Ю.В. Компьютерное моделирование жестких гибридных систем: монография. – Новосибирск: Изд-во НГТУ, 2012. – 451 с.
5. Everhart E. A new method for integrating orbits // Bull. Amer. Astronom. Soc. – 1973. – Vol. 5. – P. 389.
6. Дзядык В.К. О применении линейных методов к приближению полиномами решений обыкновенных дифференциальных уравнений и интегральных уравнений Гаммерштейна // Известия АН СССР. Сер. матем. – 1970. – Т. 34. – Выпуск 4. – С. 827 – 848.
7. Плахов Ю.В., Мыценко А.В., Шельпов В.А. О методике численного интегрирования уравнений возмущенного движения ИСЗ в задачах космической геодезии // Известия вузов. Геодезия и аэрофотосъемка. – 1989. – №4. – С. 61 – 67.
8. Афанасьев А.П., Дзюба С.М., Кириченко М.А., Рубанов Н.А. Приближенное аналитическое решение систем обыкновенных дифференциальных уравнений с полиномиальной правой частью // Журнал вычислительной математики и математической физики. – 2013. – Т. 53. – № 2. – С. 321 – 328.
9. Афанасьев А.П., Дзюба С.М. Метод построения приближенных аналитических решений дифференциальных уравнений с полиномиальной правой частью // Журнал вычислительной математики и математической физики. – 2015. – Т. 55. – №10. – С. 1694 – 1702.

10. Awoyemi D.O., Kayode S.J., Adoghe L.O. A Five-Step P-Stable Method for the Numerical Integration of Third Order Ordinary Differential Equations // American Journal of Computational Mathematics. – 2014. – № 4. – P. 119 – 126.
11. Fatimah B.O., Senapon W.A., Adebowale A.M. Solving Ordinary Differential Equations with Evolutionary Algorithms // Open Journal of Optimization. – 2015. – № 4. – P. 69 – 73.
12. Березин И.С., Жидков Н.П. Методы вычислений. Т.2. – М.: Физматгиз, 1962. – 640 с.
13. Everhart E. Implicit single sequence methods for integrating orbits // Celest. Mech. – 1974. – Vol. 10. – P. 35 – 55.
14. Everhart E. An efficient integrator that uses Gauss-Radau spacings // Dynamics of Comets: Their Origin and Evolution. Proc. Of IAU Colloq. 83.Italy, 1984 / Eds. A Carusiand G.B. Valsecchi.Dordrecht: Reidel, Astrophys. And Space Science Library Rame, 1985. – Vol. 115. – P. 185 – 202.
15. Татевян С.К., Сорокин Н.А., Залеткин С.Ф. Численное интегрирование обыкновенных дифференциальных уравнений на основе локальных многочленных приближений // Вычислительные методы и программирование. – 2000. – Т. 1. – С. 28 – 61.
16. Авдюшев В.А. Интегратор Гаусса-Эверхарта // Вычислительные технологии. – 2010. – Т. 15. – № 4. – С. 31 – 47.
17. Арушанян О.Б., Волченскова Н.И., Залеткин С.Ф. Метод решения задачи Коши для обыкновенных дифференциальных уравнений с использованием рядов Чебышёва // Вычислительные методы и программирование. – 2013. – Т. 14. – С. 203 – 214.
18. Арушанян О.Б., Залеткин С.Ф. Приближенное решение задачи Коши для обыкновенных дифференциальных уравнений методом рядов Чебышева // Вычислительные методы и программирование. – 2016. – Т. 17, выпуск 2. – С. 121 – 131.
19. Касти Дж., Калаба Р. Методы погружения в прикладной математике. Монография. – М.: Мир, 1976. – 224 с.

20. Рихтмайер Р., Мортон К. Разностные методы решения краевых задач. – М.: Мир, 1972. – 420 с.
21. Калиткин Н.Н. Численные методы. – М.: Наука, 1978. – 512 с.
22. Самарский А.А. Теория разностных схем. – М.: Наука, 1989. – 616 с.
23. Митчелл Э., Уэйт Р. Метод конечных элементов для уравнений с частными производными. – М: Мир, 1981. – 216 с.
24. Zienkiewicz O.C., Taylor R.L. The finite element method. – John Wiley & Sons Inc, 2000. – Vol. 1: The Basis. – 708 p.
25. Дегтярев Л.М., Дроздов В.В., Иванова Т.С. Метод адаптивных к решению сеток в одномерных краевых задачах с пограничным слоем. – М., 1986. – 26 с. (Препринт / АН СССР. ИПМатем.; № 164).
26. Галанин М.П., Еленина Т.Г. Тестирование разностных схем для линейного уравнения переноса. − М.: ИПМ им. М.В. Келдыша РАН, 1999.– препринт № 40. – 42 с.
27. Жамбалова Д.Б., Черный С.Г. Метод интерполяционного профиля решения уравнений переноса // Вестник НГУ. Серия: Информационные технологии. – 2012. – Том 10. – Выпуск 1. – С. 33 – 54.
28. Рогов Б.В., Михайловская М.Н. Монотонная высокоточная компактная схема бегущего счета для квазилинейных уравнений гиперболического типа // Математическое моделирование. – 2011. – Т. 23. – № 12. – С. 65 – 78.
29. Сиковский Д.Ф. Методы вычислительной теплофизики. – Новосибирск: Изд-во Новосиб. гос. ун-та, 2013. – 98 с.
30. Barth T.J., Deconinck H. (eds.) High-order methods for computational physics. – Springer, Berlin, 1999. – 587 p.
31. Liu X.-D., Osher S. Nonoscillatory high order accurate self-similar maximum principle satisfying shock capturing schemes I // SIAM J. Numer. Anal. – 1996. – V. 33.– № 2.– P. 760 – 779.
32. Макаров Е.Г. Инженерные расчеты в Mathcad 15. – СПб.: Питер, 2011. – 400 с.
33. Thompson I. Understanding Maple. – Cambridge, 2017. – 236 p.

34. Корн Г.А., Корн Т.М. Справочник по математике для научных работников и инженеров: Определения. Теоремы. Формулы: Пер. с англ. под ред. И.Г. Арамановича. – 6. изд. – СПб.: Лань, 2003. – 831 с.
35. Хайрер Э., Нёрсетт С., Ваннер Г. Решение обыкновенных дифференциальных уравнений. Нежесткие задачи. – М.: Мир, 1990. – 512 с.
36. Shampine L.F., Gordon M.K. Computer Solution of Ordinary Differential Equations.– W.H. Freeman & Co., 1975.
37. Авдюшев В.А. Эффективные методы численного моделирования околопланетной орбитальной динамики: диссертация … доктора физико-математических наук: 01.03.01. – Томск, 2009. – 210 с.
38. Сальникова Т.В., Степанов С.Я. Математическая модель образования космических пылевых облаков Кордылевского // Доклады РАН. – 2015. – Т. 463. – № 2. – С. 164 – 167.
39. Laskar J., Robutel P. High order symplectic integrators for perturbed Hamiltonian systems // Cel. mech. – 2001. – V. 80. – P. 39 – 62.
40. Chambers J.E. A Hybrid Symplectic Integrator that Permits Close Encounters between Massive Bodies // Monthly Notices of the Royal Astronomical Society. – 1999. – V. 304. – P. 793 – 799.
41. Авдюшев В.А. О численном интегрировании орбит с короткопериодическими возмущениями // Изв. вузов. Физика. 2006а. Приложение. – Т. 49. – Вып. 2. – С. 31 – 43.
42. Бордовицына Т.В. Современные численные методы в задачах небесной механики. – М.: Наука, 1984. – 136 с.
43. Rocher P., Chapront J. Observations and Ephemerides of the Faint Satellites of Jupiter // Astron. Astrophys. – 1996. – V. 311. – P. 710 – 714.
44. Новиков Е.А., Исаева С.И. Численное сравнение методов Мерсона и Эверхарта на гравитационной задаче двух тел // Системы управления и информационные технологии.– 2011.– Т. 43.– № 1.– С. 25 – 29.

45. Борисов В.Г. Модифицированная версия интегратора Гаусса-Эверхарта // Вестник Кемеровского госуд. университета. – 2015. – Т. 5. – № 2 (62). – С. 38 – 42.
46. Борунов В.П., Иванов В.А., Миронов С.В. Сравнение эффективности методов Фелберга, Дорманда-Принса и Эверхарта численного интегрирования систем обыкновенных дифференциальных уравнение.– М.: ВЦ РАН. – 2001.– С. 17 – 62.
47. Скворцов Л.М. Численное решение обыкновенных дифференциальных и дифференциально-алгебраических уравнений. – М.: ДМК Пресс, 2018. – 230 с.
48. Li X., Liao S. More than six hundred new families of Newtonian periodic planar collisionless three-body orbits // Sci. ChinaPhys. Mech. Astron. – 2017. –V.60. – P.129511.
49. Голиков А.Р. Численно-аналитическая теория THEONA движения искусственных спутников небесных тел // Космические исследования.– 2012.– Т. 50.– № 6.– С. 480 – 489.
50. Черноусько Ф.Л., Акуленко Л.Д., Соколов Б.Н. Управление колебаниями. – М.: Наука, 1980. – 384 с.
51. Гроздовский Г.Л., Иванов Ю.Н., Токарев В.В. Механика космического полета: Проблемы оптимизации. – М.: Наука, 1975. – 720 с.
52. Колесников А.А. Прикладная синергетика: основы системного синтеза. – Таганрог: Изд-во ТТИ ЮФУ, 2007. – 384 с.
53. Можаев Г.В. Синтез орбитальных структур спутниковых систем: теоретико-групповой подход. – М.: Машиностроение, 1989. – 303 с.
54. MATLAB. version 9.3.0.713579 (R2017b). Natick, Massachusetts: The MathWorks Inc.; 2017.
55. Кутимская М.А. Биофизические основы иммунной системы человека в свете современного состояния природы и метасоциума // САНВШ, В-Спектр.– 2007. – С. 326 – 331.
56. Чуличков А.И. Математические методы нелинейной динамики. – М.: Физматлит, 2000. – 296 с.

57. Жаботинский А.М., Филд Р., Огмер Х. Колебания и бегущие волны в химических системах. – М.: Мир, 1988. – 720 с.
58. Воробьев А.А. Микробиология и иммунология. – М.: Медицина, 1999. – 464 с.
59. Мюррей Дж. Математическая биология. Том I. Введение. – М.-Ижевск: НИЦ «Регулярная и хаотическая динамика», Институт компьютерных исследований, 2009. – 776 с.
60. Ризниченко Г.Ю. Математические модели в биофизике и экологии. – М.-Иж.: ИКИ, 2003.– 184 с.
61. Белоусов Б.П. Периодически действующая реакция и её механизмы: сборник рефератов по радиационной медицине за 1958 год. – М. – С. 145.
62. Tyson J.J. What everyone should know about the Belousov-Zhabotinsky reaction // Frontiers in Mathematical Biology, V. 100 of Lect. Notes in Biomathematics, P. 569 – 587. Springer-Verlag, Berlin-Heidelberg-New York. – 1994.
63. Field R.J., Noyes R.M. Oscillations in Chemical Systems. IV. Limit Cycle Behavior in a Model of a Real Chemical Reaction // J. Chem. Phys. – 1974. – V. 60. – № 5. – P. 1877 – 1884.
64. Ромм Я.Е. Компьютерно-ориентированный анализ устойчивости на основе рекуррентных преобразований разностных решений обыкновенных дифференциальных уравнений // Кибернетика и системный анализ. – 2015. – Т. 51. – № 3. – С. 107 – 124.
65. Ромм Я.Е., Джанунц Г.А. Компьютерный анализ устойчивости по Ляпунову на основе мультипликативных и аддитивных преобразований решений обыкновенных дифференциальных уравнений /ТГПИ. – Таганрог, 2014. – 49 с. – ДЕП в ВИНИТИ 17.02.2014, №53-В2014.
66. Бахвалов Н.С., Лапин А.В., Чижонков Е.В. Численные методы в задачах и упражнениях. – М.: Высшая школа, 2000. – 195 с.
67. Dormand J.R., Prince P.J. A family of embedded Runge-Kutta formulae // J. Comp. Appl. Math. – 1980.– V. 6.– P. 19 – 26.

68. Амосов А.А., Дубинский Ю.А., Копченова Н.В. Вычислительные методы для инженеров: Учеб. пособие. – М.: Высш. шк., 1994. – 544 с.
69. Де Борк К. Практическое руководство по сплайнам. – М.: Радио и связь, 1985. – 304 с.
70. Hairer E. A Runge-Kutta method of order 10 // J. Inst. Math. Applics.– 1978. – V. 21.– P. 47 – 59.
71. Prince P.J., Dormand J.R. High order embedded Runge-Kutta formulae // J. Comp. Appl. Math. – 1981.– V. 7 – P. 67 – 75.
72. Бахвалов Н.С., Жидков Н.П., Кобельков Г.М. Численные методы. – 6-е изд. – М.: БИНОМ. Лаборатория знаний, 2008. – 636 с.
73. Аксайская Л.Н. Разработка и исследование параллельных схем цифровой обработки сигналов на основе минимизации временной сложности вычисления функций: диссертация… кандидата технических наук: 05.13.17, 05.13.18.– Таганрог: ЮФУ.– 2008. – 154 с.
74. Ромм Я.Е. Бесконфликтные и устойчивые методы детерминированной параллельной обработки / автореферат диссертации … доктора технических наук: 05.13.17, 05.13.13. – Таганрог: ТРТУ. – 1998. – 42 с.
75. Ромм Я.Е. Локализация и устойчивое вычисление нулей многочлена на основе сортировки. II // Кибернетика и системный анализ. – 2007. – № 2. – С. 161 – 174.
76. Джанунц Г.А., Ромм Я.Е. Варьируемое кусочно-интерполяционное решение задачи Коши для обыкновенных дифференциальных уравнений с итерационным уточнением // Журнал вычислительной математики и математической физики.– 2017. – Т.57. – №10. – С. 1641 – 1660.
77. Калиткин Н.Н. Численные методы решения жестких систем // Математическое моделирование. – 1995. – Т. 7. – № 5. – С. 8 – 11.
78. Ракитский Ю.В., Устинов С.М., Черноруцкий И.Г. Численные методы решения жестких систем. – М.: Наука, 1979. – 208 с.
79. Lambert J.D. Computational methods in ordinary differential equations. – John Wiley & Sons Inc., New York, 1973.

80. Новиков Е.А. Явные методы для жестких систем. – Новосибирск. Наука. Сиб. предприятие РАН, 1997. – 195 с.
81. Butcher J.C. Implicit Runge-Kutta processes //Math. Comp. – 1964. – V. 18. – № 85. – P. 50 – 64.
82. Rosenbrock H.H. Some general implicit processes for the numerical solution of differential equations // The Comput. J. – 1963. – V. 5. – № 4. – P. 329 – 330.
83. Калиткин Н.Н. Полуявные схемы для задач большой жесткости // ЭНТП, Серия Б. – Т.VII-1, Ч.1. – М.: Янус-К, 2008. – С. 153 – 171.
84. Холодов А.С. Численные методы решения уравнений и систем гиперболического типа // Энциклопедия низкотемпературной плазмы. – Т.VII-1.4.2. – М.: Янус-К, 2008. – С.141 – 174.
85. Рождественский Б.Л., Яненко Н.Н. Системы квазилинейных уравнений и их приложения к газовой динамике. – М.: Наука, 1968. – 546 с.
86. Магомедов К.М., Холодов А.С. Сеточно-характеристические численные методы. – М.: Наука, 1988. – 287 с.
87. Куликовский А.Г., Погорелов Н.В., Семенов А.Ю. Математические вопросы численного решения гиперболических систем уравнений. – М.: Наука, 2001. – 608 с.
88. Седов Л.И. Механика сплошной среды. – М.: Наука, 1977. – 1060 с.
89. Лайтхилл Дж. Волны в жидкостях. – М.: Мир, 1981. – 598 с.
90. LeVeque R.J. Finite Volume Methods for Hyperbolic Problems. – Cambridge University Press, 2002. – 558 p.
91. Зайцев А.И., Пелиновский Е.Н. Прогноз высот волн цунами на Российском побережье Чёрного моря // Океанология. – 2011. – Т. 51. – № 6. – С. 965 – 973.
92. Родин, А.А. Влияние эффектов обрушения на трансформацию и накат длинных волн на берег: специальность 01.02.05 «Механика жидкости, газа и плазмы»: диссертация на соискание ученой степени кандидата физико-математических наук / Родин Артём Александрович; Нижегор. гос. техн. ун-т им Р.Е. Алексеева. – Нижний Новгород, 2013. – 121 с.

93. Руденко О.В., Солуян С.И. Теоретические основы нелинейной акустики. – М.: Наука, 1975. – 384 с.
94. LeVeque R.J. A Well-Balanced Path-Integral f-wave Method for Hyperbolic Problems with Source Terms // Journal of Scientific Computing. – 2010. – V. 48. – P. 209 – 226.
95. .Lax P.D., Wendroff B. Systems of conservation laws // Commun. Pure Appl Math. – 1960. – V. 13(2). – P. 217 – 237.
96. Pulliam T.H., Zingg D.W. Fundamental algorithms in computational fluid dynamics. – Switzerland: Springer. – 2014. – 211 p.
97. Уизем Дж. Линейные и нелинейные волны. – М.: Мир, 1977.
98. Куликовский А.Г., Свайникава Е.И. Нелинейные волны в упругих средах. – М.: Моск. лицей, 1998. – 412 с.
99. Гужев Д.С., Калиткин Н.Н. Уравнение Бюргерса – тест для численных методов // Матем. моделирование. 1995. – Т. 7. – № 4. – С. 99 – 127.
100. Барахнин В.Б., Карамышев В.Б. TVD-Схема на подвижной адаптивной сетке // Вычислительные технологии. – 2000. – Т. 5. – № 1. – С. 19 – 30.
101. Зайцев В.Ф., Полянин А.Д. Справочник по дифференциальным уравнениям с частными производными первого порядка. – М.: ФИЗМАТЛИТ, 2003. – 416 с.
102. Лисковец О.А. Метод прямых // Дифференц. уравнения. – 1965. – Т. 1. – № 12. – С. 1662 – 1678.
103. Rothe E. Zweidimensionale parabolische Randwertaufgaben als Grenzfall eindimensionaler Randwertaufgaben // Math. Ann. –1930. –V.102. –P. 650 – 670.
104. Fernández Pablo. Entropy-Stable Hybridized Discontinuous Galerkin Methods for Large-Eddy Simulation of Transitional and Turbulent Flows: Ph.D. in Computational Science and Engineering. – Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, 2019. – 212 p.
105. Deardorff J.W. A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers. // J. Fluid Mech. – 1970. –V. 41. – P. 453 – 480.

106. Deardorff J.W. The use of subgrid transport equations in a three-dimensional model of atmospheric turbulence // Journal of Fluids Engineering. – 1973. – V. 9. – P. 429 – 438.

107. Волков К.Н. Моделирование крупных вихрей в турбулентной струе, истекающей в затопленное пространство или спутный поток // Прикладная механика и техническая физика. – 2011. – Т. 52. – № 1. – С. 60 – 70.

108. Wilmer W.Nichols, Michael F.O'Rourke. Blood flow in arteries. – London, Edward Arnold. – 1990.

109. Соснин, Н. В. Линейный анализ распространения пульсовых волн в сердечно-сосудистой системе: специальность 05.13.18 «Математическое моделирование, численные методы и комплексы программ»: диссертация на соискание ученой степени доктора физико-математических наук / Соснин Николай Васильевич; Московский государственный университет имени М.В. Ломоносова. – Москва, 2008. – 313 с.

110. Кошелев В.Б., Мухин С.И., Соснин Н.В., Фаворский А.П. Математические модели квазиодномерной гемодинамики: Методическое пособие. – М.: МАКС-Пресс, 2010. – 114 с.

111. Мухин С.И., Соснин Н.В., Фаворский А.П., Хруленко А.Б. Линейный анализ волн давления и скорости в системе эластичных сосудов. Препринт – М.: МАКС-Пресс, 2001. – 40 с.

112. Численное моделирование распространения гемодинамических импульсов [Текст] / А. П. Фаворский, М. А. Тыглиян, Н. Н. Тюрина [и др.] // Математическое моделирование. – 2009. – Т. 21, № 12. – С. 21-34.

113. Фаварский А.П., Тыглиян М.А., Тюрина Н.Н., Галанина А.М., Исаков В.А. Численное моделирование распространения гемодинамических импульсов // Матем. моделирование. – 2009. – Т. 21. – № 12. – С. 21 – 34.

114. Solin P. Partial Differential Equations and the Finite Element Method. – NJ: J. Wiley & Sons, 2005.

115. Рогов Б.В., Михайловская М.Н. Монотонные бикомпактные схемы для линейного уравнения переноса // Математическое моделирование. – 2011. – Т. 23. – № 6. – С. 98 – 110.

116. Ferm L., Lötstedt P. Space-Time Adaptive Solution of First Order PDEs. – J Sci Comput. – 2006. –V. 26. – P. 83 – 110.

117. Демидович Б.П., Марон И.А., Шувалова Э.З. Численные методы анализа. Приближение функций, дифференциальные и интегральные уравнения. – СПб.: Лань, 2010. – 400 с.

118. Ромм Я.Е., Джанунц Г.А. Схема разностного решения обыкновенных дифференциальных уравнений с повышенной точностью на основе интерполяционного полинома Ньютона // Известия ЮФУ. Технические науки. – 2009. – № 5. – С. 46 – 52.

119. Ромм Я.Е., Джанунц Г.А. Кусочная линеаризация задачи Коши для обыкновенных дифференциальных уравнений // Известия ЮФУ. Технические науки. – 2011. – № 2. – С. 26 –32.

120. Ромм Я.Е., Джанунц Г.А. Компьютерный метод разностно-аналитического решения обыкновенных дифференциальных уравнений на основе интерполяционного полинома Ньютона. – Таганрог, 2009. – 40 с. – Деп. в ВИНИТИ 18.06.09, № 379-В2009.

121. Ромм Я.Е., Джанунц Г.А. Компьютерная схема преобразования разностных решений обыкновенных дифференциальных уравнений в кусочно-полиномиальную форму // Прикладная информатика и математическое моделирование. – Москва: МГУП, 2009. – С. 55 – 60.

122. Romm Ya.E., Dzhanunts G.A. Difference-polynomial solutions of Cauchy problem for the ordinary differential equations using the parallel recovery coefficients of the polynomial from its roots // International Conference Parallel Computer Algebra 2010; Tambov, June 29 – July 3, 2010 / Tambov State University named after G.R. Derzhavin, 2010. – C. 17 – 18.

123. Джанунц Г.А., Ромм Я.Е. Компьютерное моделирование жестких систем на основе кусочно-полиномиальной аппроксимации решений

обыкновенных дифференциальных уравнений. – Таганрог, 2011. – 20 с. – Деп. в ВИНИТИ 31.08.2011, № 405-В2011.

124. Березин И.С., Жидков Н.П. Методы вычислений. Т.1. – М.: Наука, 1966. – 632 с.

125. Ромм Я.Е., Голиков А.Н., Распараллеливаемые кусочно-полиномиальные схемы аппроксимации функций, производных и вычисления определённых интегралов с повышенной точностью / ТГПИ. – Таганрог, 2010. – 139 с. – Деп. в ВИНИТИ 27.04.2010, № 230-В2010.

126. Ромм Я.Е., Джанунц Г.А., Кусочно-полиномиальный метод численного решения систем обыкновенных дифференциальных уравнений и уравнений в частных производных / ТГПИ. – Таганрог, 2011. – 59 с. – Деп. в ВИНИТИ 20.07.2011, № 353-В2011.

127. Джанунц Г.А. Компьютерный метод кусочно-полиномиального приближения решений обыкновенных дифференциальных уравнений в применении к моделированию автоколебательных реакций: диссертация … кандидата технических наук: 05.13.18. – Таганрог: ЮФУ, 2012. – 152 с.

128. Ромм Я.Е., Джанунц Г.А. Компьютерный метод разностно-полиномиального решения задачи Коши для обыкновенных дифференциальных уравнений / ТГПИ. – Таганрог, 2011. – 45 с. – Деп. в ВИНИТИ 25.03.2011, № 141-В2011.

129. Ромм Я.Е., Джанунц Г.А., Повышение точности разностных решений обыкновенных дифференциальных уравнений на основе кусочно-полиномиальной интерполяции / ТГПИ. – Таганрог, 2010. – 103 с. – Деп. в ВИНИТИ 25.01.2010, № 20-В2010.

130. Ромм Я.Е., Джанунц Г.А. Кусочно-полиномиальная аппроксимация решения задачи Коши для систем обыкновенных дифференциальных уравнений на основе преобразования интерполяционного полинома Ньютона / ТГПИ. – Таганрог, 2010. – 37 с. – Деп. в ВИНИТИ 25.05.2010, № 305-В2010.

131. Ромм Я.Е., Джанунц Г.А. Кусочно-полиномиальные приближения функций и решений дифференциальных уравнений в применении к

моделям периодических реакций / Монография. – Изд-во ТГПИ имени А.П. Чехова. – Таганрог, 2013. – 240 с.

132. Ромм Я.Е., Джанунц Г.А. Компьютерный метод варьируемой кусочно-полиномиальной аппроксимации функций и решений обыкновенных дифференциальных уравнений // Кибернетика и системный анализ, Киев. – 2013. – № 3. – С. 95 – 112.
133. Матвеев Н.М. Методы интегрирования обыкновенных дифференциальных уравнений. – Л.: Изд-во Ленинградского университета, 1955. – 655 с.
134. Никуличев Ю.В. Численные методы на основе эрмитовых сплайнов решения задачи Коши для систем обыкновенных дифференциальных уравнений, аппроксимации кривых и поверхностей, в задачах оптимального управления / автореферат диссертации … доктора физико-математических наук: 05.13.18. – Новосибирск: ИВТ СО РАН, 2005. – 34 с.
135. Shampine L.F. and Reichelt M.W. The MATLAB ODE Suite // SIAM Journal on Scientific Computing. – 1997. – V. 18. – P. 1 – 22.
136. Нуньес-Иглесиас Х., Уолт Ш., Дэшноу Х. Элегантный SciPy. – ДМК Пресс, 2018. – 266 с.
137. Gear C.W. Numerical initial value problems in ordinary differential equations. – Englewood Cliffs: Prentice Hall, 1971.
138. England R. Error estimates for Runge–Kutta type solutions to systems of ordinary differential equations // The Computer Journal. – 1969. –V.12. –№ 2. – P. 166 –170.
139. Maruyama K. On the Parallel Evaluation of Polynomials // IEEE Trans. on Computers. – 1973. – V. 22. – № 1. – P. 2 – 5.
140. Ромм Я.Е., Джанунц Г.А. Итерационное уточнение кусочно-полиномиального приближения решения задачи Коши для обыкновенных дифференциальных уравнений. – Таганрог, 2015. – 60 с. – Деп. в ВИНИТИ 07.12.2015, № 201-В2015.

141. Ромм Я.Е., Джанунц Г.А. Компьютерный эксперимент по кусочно-интерполяционной реализации дифференциальных моделей // Новые информационные технологии и системы: Сборник научных статей XVI Международной научно-технической конференции. – Пенза, Пензенский государственный университет, 2019. – С. 250 – 256.

142. Джанунц Г.А. Численное интегрирование обыкновенных дифференциальных уравнений с использованием кусочно-полиномиальной аппроксимации // Вестник Таганрогского института имени А.П. Чехова. – Таганрог, 2020. – № 2. – С. 6 – 15.

143. Ромм Я.Е., Джанунц Г.А. Кусочно-интерполяционное решение обыкновенных дифференциальных уравнений с приложением к задачам численного моделирования // Актуальные проблемы прикладной математики, информатики и механики: сборник трудов Международной научной конференции, Воронеж, 7 – 9 декабря 2020 г. – Воронеж, 2021. – С. 1038 – 1045.

144. Демидович Б.П., Марон И.А. Основы вычислительной математики. – СПб.: Лань. – 2016. – 672 с.

145. Ромм Я.Е., Джанунц Г.А. Компьютерное кусочно-интерполяционное решение одноточечной и двухточечной задачи Коши для обыкновенных дифференциальных уравнений / Ин-т им. А.П. Чехова (филиал) «РГЭУ (РИНХ)». – Таганрог. 49 с. – Деп. в ВИНИТИ 05.04.16, № 57-В2016.

146. Ромм Я.Е., Джанунц Г.А. Кусочно-интерполяционное решение двухточечной задачи Коши для обыкновенных дифференциальных уравнений с итерационным уточнением // Фундаментальные исследования. – 2016. – № 6 (часть 2). – С. 308 – 317.

147. Попов И.В. Построение разностной схемы повышенного порядка аппроксимации для нелинейного уравнения переноса с использованием адаптивной искусственной вязкости // Препринты ИПМ им. М.В. Келдыша. – 2017. – № 68. – 21 с.

148. Самарский А.А., Вабищевич П.Н. Разностные схемы для уравнения переноса // Дифференц. ур-ния. – 1998. – Т. 34. – № 12. – С. 1675 – 1685.

149. Самарский А.А., Вабищевич П.Н. Нелинейные монотонные схемы для уравнения переноса // Докл. АН СССР. – 1998. – Т. 361. – № 1. – С. 21 – 23.

150. Ромм Я.Е., Джанунц Г.А. Кусочно-интерполяционное решение задачи Коши для уравнения переноса / Ин-т им. А.П. Чехова (филиал) «РГЭУ (РИНХ)». – Таганрог, 2019. –40 с. – Деп. в ВИНИТИ 20.08.19, № 68.

151. Ромм Я.Е., Джанунц Г.А. Кусочно-интерполяционное решение дифференциальных уравнений в частных производных // Известия ЮФУ. Технические науки. Тематический выпуск: «Компьютерные и информационные технологии в науке, инженерии и управлении». – 2011. – № 5. – С. 146 – 153.

152. Ромм Я.Е., Джанунц Г.А. Варьируемое кусочно-интерполяционное решение уравнения переноса // Итоги науки и техники. Серия «Современная математика и ее приложения. Темататические обзоры». – Т. 166. – ВИНИТИ РАН, М. – 2019. – С. 77 – 86.

153. Ромм Я.Е., Джанунц Г.А. Варьируемое кусочно-интерполяционное решение задачи Коши для уравнения переноса с итерационным уточнением // Современные наукоемкие технологии. – 2020. – № 1. – С. 21 – 46.

154. Romm Ya.E., Dzhanunts G.A. Piecewise interpolation solution of the Cauchy problem for the transport equation with iterative refinement // Journal of Physics: Conference Series. Applied Mathematics, Computational Science and Mechanics: Current Problems. – 2020. – V. 1479. – p. 012110.

155. Ромм Я.Е., Джанунц Г.А. Компьютерный метод разностно-полиномиального решения задачи Коши для уравнений в частных производных // Обозрение прикладной и промышленной математики. – Т. 18. – Редакция «ОПиПМ», Москва. – 2011. – С. 141.

156. Ромм Я.Е., Джанунц Г.А. Кусочно-интерполяционные схемы решения обыкновенных дифференциальных уравнений и уравнений в частных производных // Высокие технологии, образование, промышленность. Т. 1: сборник статей XI международной научно-практической конференции

«Фундаментальные и прикладные исследования, разработка и применение высоких технологий в промышленности». 27 – 29 апреля 2011 г., Санкт-Петербург, Россия / под ред. А.П. Кудинова. – СПб.: Изд-во Политехн. университета, 2011. – С. 127 – 128.

157. Ромм Я.Е., Джанунц Г.А. Разностно-полиномиальный метод численного решения систем обыкновенных дифференциальных уравнений и уравнений в частных производных. – Таганрог, 2011. – 59 с. – Деп. в ВИНИТИ 20.07.2011, № 353-В2011.
158. Ромм Я.Е., Джанунц Г.А. Варьируемая непрерывная кусочно-полиномиальная аппроксимация функций одной и двух переменных и решений ОДУ с оценками скорости сходимости. – Таганрог, 2013. – 87 с. – Деп. в ВИНИТИ 14.01.2013, № 8-В2013.
159. Ромм Я.Е., Джанунц Г.А. Компьютерный метод кусочно-полиномиального приближения решений обыкновенных дифференциальных уравнений и уравнений в частных производных. – Таганрог, 2015. – 100 с. – Деп. в ВИНИТИ 27.02.2015, № 43-В2015.
160. Ромм Я.Е., Джанунц Г.А. Кусочно-интерполяционное приближение решения задачи Коши для уравнений в частных производных. – Таганрог, 2017. – 41 с. – Деп. в ВИНИТИ 14.08.2017, № 88-В2017.
161. Ромм Я.Е., Джанунц Г.А. Кусочно-полиномиальное приближение решения задачи Коши для гиперболического уравнения / В кн.: Фундаментальные исследования, методы и алгоритмы прикладной математики в технике, медицине и экономике. – Новочеркасск: Южно-Российск. госуд. политехнич. университет (НПИ) имени М.И. Платова, 2017. – С. 9 – 15.
162. Ромм Я.Е., Джанунц Г.А. Оценка скорости сходимости кусочно-интерполяционного решения задачи Коши с итерационным уточнением для уравнения переноса. – Таганрог, 2018. – 79 с. – Деп. в ВИНИТИ 12.02.2018, №20-В2018.
163. Ромм Я.Е., Джанунц Г.А. Варьируемое кусочно-интерполяционное решение уравнения переноса // Актуальные проблемы прикладной

математики: Материалы IV Международной научной конференции. – Нальчик: ИПМА КБНЦ РАН, 2018. – С. 218.

164. Джанунц Г.А. Кусочно-интерполяционное решение уравнения переноса с итерационным уточнением // Фундаментальные исследования, методы и алгоритмы прикладной математики в технике, медицине и экономике: материалы 17-ой Междунар. науч.-практ. конф., г. Новочеркасск, 6-7 сентября 2018г. / Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова. – Новочеркасск: Лик, 2018. – С. 52 – 58.
165. Джанунц Г.А. Численное моделирование распространения гемодинамических импульсов с применением кусочно-интерполяционного метода // Электронный научный журнал «Вестник молодёжной науки России». – 2019. – № 2.
166. Ромм Я.Е., Джанунц Г.А. Кусочно-интерполяционное решение задачи Коши для уравнения переноса с итерационным уточнением // Актуальные проблемы прикладной математики, информатики и механики: сборник трудов Международной научной конференции, Воронеж, 11 – 13 ноября 2019 г. – Воронеж, 2020. – С. 1056 – 1065.
167. Gasca M., Sauer T. On the history of multivariate polynomial interpolation // Jornal of Computational and Applied Mathematics. – 2000. – V. 122. – P. 23 – 35.
168. Kurganov A., Tadmor E. New high-resolution central schemes for nonlinear conservation laws and convection-diffusion equations // J. Comput. Phys. – 2000. – V. 160. – P. 241 – 282.
169. Якимов А.С. Аналитический метод решения краевых задач. 2-е изд., доп. – Томск: Изд-во Том. ун-та, 2011. – 199 с.
170. Bülbül B., Sezer M. Taylor polynomial solution of hyperbolic type partial differential equations with constant coefficients // Int J Computer Math. – 2011. – V. 88. – № 3. – P. 533 – 544.
171. Nazir T, Abbas M., Yaseen M. / Lai Sh. (Reviewing Editor) Numerical solution of second-order hyperbolic telegraph equation via new cubic

trigonometric B-splines approach // Cogent Mathematics. – 2017. – V. 4. – № 1.

172. Kumar R., Choudhary A., Baskar S.Modified cubic B-spline quasi-interpolation numerical scheme for hyperbolic conservation laws // Applicable Analysis. – 2018.

173. Буланов С.Г., Джанунц Г.А. Программный анализ устойчивости систем обыкновенных дифференциальных уравнений на основе мультипликативных преобразований разностных схем и кусочно-полиномиальных приближений решения // Промышленные АСУ и контроллеры. – 2015. – № 2. – С. 10 – 20.

174. Romm Ya.E., Dzhanunts G.A. Piecewise interpolation solution of ordinary differential equations with application to numerical modeling problems // Journal of Physics: Conference Series. Applied Mathematics, Computational Science and Mechanics: Current Problems. – 2021. – V.1902(1). – p. 0121130.

175. Буланов С.Г., Джанунц Г.А. Компьютерный анализ устойчивости модели автогенератора на основе кусочно-полиномиального метода // Перспективы развития технических наук / Сборник научных трудов по итогам международной научно-практической конференции. – Челябинск, 2015. – № 2. – С. 11 – 15.

176. Буланов С.Г., Джанунц Г.А. Исследование автоколебательных цепей с использованием кусочно-полиномиального метода и компьютерных схем анализа устойчивости // Материалы III международной научно-практической конференции «Инновационные технологии и дидактика в обучении». – Таганрог: Издательство ЮФУ, 2015. – С. 118 – 123.

177. Буланов С.Г., Джанунц Г.А. Численное моделирование и компьютерный анализ устойчивости орбитального движения космических аппаратов // Вестник Таганрогского института имени А.П. Чехова. Физико-математические и естественные науки. – Таганрог, 2015. – №1. – С. 55 – 63.

178. Ромм Я.Е., Джанунц Г.А. Кусочная интерполяция функций, производных и интегралов с приложением к решению обыкновенных

дифференциальных уравнений // Современные наукоемкие технологии. – 2020. – № 12 (часть 2). – С. 291 – 316.

179. Ромм Я.Е., Джанунц Г.А., Медведкин Н.А. О библиотеке стандартных программ вычисления функций на основе кусочной интерполяции // Современные наукоемкие технологии. – 2022. – № 11. – С. 57 – 70.

180. Ромм Я.Е., Джанунц Г.А. О воспроизводимости функций, производных и решений дифференциальных уравнений с помощью хранимых коэффициентов кусочной интерполяции // Современные наукоемкие технологии. – 2023. – № 1. – С. 44 – 63.

181. Ромм Я.Е., Джанунц Г.А. О стандартизации программ вычисления интегралов на основе кусочной интерполяции подынтегральных функций // Современные наукоемкие технологии. – 2023. – № 4. – С. 71 – 92. DOI 10.17513/snt.39582

182. Ланда П.С. Нелинейные колебания и волны. – М.: Наука. Физматлит, 1997. – 496 с.

183. Слинько М.Г., Слинько М.М. Автоколебания скорости гетерогенных каталитических реакций // Успехи химии. – 1980. – Т. 49. – № 4. – С. 561 – 587.

184. Холодов А.С., Лобанов А.И., Евдокимов А.В. Разностные схемы для решения жестких обыкновенных дифференциальных уравнений в пространстве неопределенных коэффициентов: Учебно-методическое пособие. – М.: МФТИ, 2001. – 48 с.

185. Чумаков Г.А., Слинько М.Г., Беляев В.Д. Сложные изменения скорости гетерогенных каталитических реакций // ДАН СССР. – 1980. – Т. 253. – № 3. – С. 653 – 658.

186. Икрамов, Р.Д. Моделирование и численное исследование динамики колебательных химических реакций полунеявными методами: специальность 02.00.04 «Физическая химия»: диссертация на соискание ученой степени кандидата физико-математических наук / Икрамов Рустам Джамолович; Башкир. гос. ун-т. – Уфа, 2016. – 181 с.

187. Потапов В.И. О бифуркациях в динамической системе Чумакова-Слинько // Вестник Нижегородского университета им. Н.И. Лобачевского. – 2011. – № 2 (1). – С. 146 – 155.

188. Кузнецов А.П., Кузнецов С.П., Рыскин Н.М. Нелинейные колебания. Учеб. пособие для вузов. – М.: Издательство физико-математической литературы, 2002. – 292 с.

189. Пилипенко А.М., Бирюков В.Н. Исследование эффективности современных численных методов при анализе автоколебательных цепей // Журнал радиоэлектроники. – 2013. – № 9.

190. Аксёнов Е.П. Теория движения искусственных спутников Земли. – М.: Наука, 1977. – 368 с.

191. Крылов В.И. Основы теории движения ИСЗ (часть вторая: возмущенное движение): учебное пособие. – М.: МИИГАиК, 2016. – 67 с.

192. Параметры Земли 1990 года (ПЗ-90.11). – М.: «27 ЦНИИ» Минобороны России, 2020. – 30 с.

193. Space-track.org. URL: https://www.space-track.org (дата обращения: 19.07.2021).

194. David A. Vallado, Paul Crawford, Richard Hujsak, and T.S. Kelso, «Revisiting Spacetrack Report #3», presented at the AIAA/AAS Astrodynamics Specialist Conference, Keystone, CO, 2006 August 21 – 24.

195. Иванов Д.С., Трофимов С.П., Широбоков М.Г. Численное моделирование орбитального и углового движения космических аппаратов. – М.: ИПМ им. М.В. Келдыша, 2016. – 118 с.

196. Аксенов Е.П. Специальные функции в небесной механике. – М.: Наука. Гл. ред. физ.-мат. лит., 1986. – 320 с.

197. Мелешко И.Н., Нифонтова Д.А., Сорокин В.В. К приближенному интегрированию сильно осциллирующих функций // Наука и техника. – 2017. – Т. 16. – № 4. – С. 343 – 347.

198. Зубов В.И. Функции Бесселя: Учебно-методическое пособие. – М.: МФТИ, 2007. – 51 с.

199. Ромм Я.Е., Джанунц Г.А. Моделирование движения навигационных спутников системы ГЛОНАСС на основе кусочно-интерполяционного решения задачи Коши для дифференциальной системы // Современные наукоемкие технологии. – 2023. – № 2. – С. 88 – 101.

200. ГЛОНАСС. Интерфейсный контрольный документ. Общее описание системы с кодовым разделением сигналов. Редакция 1.0. Сайт ОАО «Российские космические системы». http://russianspacesystems.ru/wp-content/uploads/2016/08/IKD.-Obshh.-opis.-Red.-1.0-2016.pdf, дата обращения: 20.02.2023.

201. Дубошин Г.Н. Небесная механика: Основные задачи и методы. – М.: Наука, 1975.

202. Абалакин В.К. Основы эфемеридной астрономии. – М.: Наука, 1979.

203. Maciuk K. (2016). Different approaches in GLONASS orbit computation from broadcast ephemeris. Geodetski vestnik. 60. 437 – 448. DOI:10.15292/geodetski-vestnik.2016.03.437-448.

204. Gurtner W., Estey L. (2007): «RINEX: The Receiver Independent Exchange Format Version 3.00». ftp://igscb.jpl.nasa.gov/igscb/data/format/rinex300.pdf, дата обращения 20.02.2023.

205. ГЛОНАСС. Интерфейсный контрольный документ. Система высокоточного определения эфемерид и временных поправок. Редакция 3.0. 2011. Сайт ПМК СВОЭВП. URL: http://www.glonass-svoevp.ru/DATA/Documents/IKD_SVO.pdf. (дата обращения: 03.02.2023).

206. Dach R., Lutz S., Walser P., Fridez P. (Eds). Bernese GNSS Software Version 5.2. User manual, Astronomical Institute. University of Bern. Bern Open Publishing. 2015. DOI:10.7892/boris.72297.

ПРИЛОЖЕНИЕ К ГЛАВЕ 2

П2.1. Приложение к п. 2.1. Вычисление действительных функций одной действительной переменной выполняется с априори заданной границей абсолютной погрешности при условии минимизации временной сложности. Отрезок приближения разбивается на подынтервалы равной длины, на каждом из которых строится интерполяционный полином Ньютона, программно преобразуемый в форму полинома с числовыми коэффициентами. Степень полинома и число подынтервалов программно варьируются и фиксируются при достижении требуемой границы погрешности с минимальной временной сложностью. Схема построения метода и его математическое обоснование представлены в главе 2. Непосредственно ниже приводится программная реализация метода и результаты численного эксперимента на примере вычисления значений функции $u(x)=\sqrt[3]{\operatorname{arctg}\left(e^{\cos\sqrt[5]{(1/x)}}\right)}$ при $x\in[\,0.5,1\,]$ с фиксированным значением границы абсолютной погрешности $\varepsilon=10^{-18}$.

```
program vpi_f_Newton; {$APPTYPE CONSOLE} Uses SysUtils, Math;
const abs_error=1.0e-18; nn=20;
type M=array[0..nn] of extended; MM=array[0..nn,0..nn] of extended;
var n_min,k_min:byte;a0,b0:extended; d: MM; k_output:integer;
function u(x:extended):extended;
begin u:=Power(arctan(exp(cos(Power(1/x,1/5)))),1/3) end;
procedure Konech_Raznoct(n:byte;a00,h: extended; var dy:MM); var i,k:byte;x:M;
begin for i:=0 to n do x[i]:=a00+i*h;
for i:=0 to n-1 do dy[1,i]:=u(x[i+1])-u(x[i]);
for k:=2 to n do for i:=0 to n-k do dy[k,i]:=dy[k-1,i+1]-dy[k-1,i] end;
procedure Viet(n:byte;var d:MM); var q:MM;k,i,j:byte;
begin q[1,1]:=1; q[1,0]:=0; for k:=2 to n do begin
q[k,0]:=-q[k-1,0]*(k-1); for i:=1 to k-1 do
q[k,k-i]:=q[k-1,k-i-1]-q[k-1,k-i]*(k-1); q[k,k]:=q[k-1,k-1] end;
 for j:=1 to n do  for i:=0 to j do d[i,j]:=q[j,i] end;
procedure pi_Newton(print:boolean;n,k:byte; var cond:boolean);
var xpr,t,hpr,s,p,h:extended; factorial:integer;
    dy: MM; a00, b00, vel_podint: extended; i, j, l: byte; b,y,a: M;kk:int64;
begin vel_podint:=(b0-a0)/exp(k*ln(2)); a00:=a0; b00:=a00+vel_podint; kk:=0;
while a00<=b0-vel_podint do begin
h:=(b00-a00)/n; hpr:=h/3; Konech_Raznoct(n,a00,h,dy); xpr:=a00;
while xpr<=b00 do begin
t:=(xpr-a00)/h; factorial:=1;
for j:=1 to n do begin factorial:=factorial*j; b[j]:=dy[j,0]/factorial; end;
a[0]:=u(a00); for l:=1 to n do begin s:=0;
for j:=l to n do s:=s+d[l,j]*b[j]; a[l]:=s end; p:=a[n];
for i:=n-1 downto 0 do p:=p*t+a[i];
if print then begin inc(kk);
```

```
if kk=k_output then begin writeln(xpr,' ',p,' ',abs(p-u(xpr)));kk:=0;end; end;
if abs(p-u(xpr))>abs_error then cond:=false; xpr:=xpr+hpr end;
a00:=a00+vel_podint; b00:=a00+vel_podint end; end;
procedure vpi_f(a0,b0:extended; var n_min,k_min:byte);
var n,k:byte; cond:boolean;
begin n:=1; repeat k:=0; repeat cond:=true; pi_Newton(false,n,k,cond);
if cond=true then begin n_min:=n;k_min:=k; pi_Newton(true,n,k,cond);
writeln; writeln('n_min = ',n_min,'  k_min = ',k_min); exit; end
else k:=k+1; until k>17; n:=n+1 until n>15; end;
begin k_output:=50000; a0:=1/2; b0:=1; Viet(nn,d); vpi_f(a0,b0,n_min,k_min);
readln; end.
```

Результат работы программы (в 1-м столбце – аргумент функции, во 2-м – вычисленное значение функции, в 3-м – абсолютная погрешность кусочно-интерполяционного приближения):

```
 5.31788508097331E-0001  9.96862645372149E-0001  3.25260651745651E-0019
 5.63577651977539E-0001  9.98675242936278E-0001  0.00000000000000E+0000
 5.95366795857747E-0001  1.00034513563823E+0000  2.16840434497101E-0019
 6.27155939737956E-0001  1.00189023820867E+0000  2.16840434497101E-0019
 6.58945083618164E-0001  1.00332546262406E+0000  0.00000000000000E+0000
 6.90734227498372E-0001  1.00466333593303E+0000  2.16840434497101E-0019
 7.22523371378581E-0001  1.00591446948345E+0000  1.08420217248550E-0019
 7.54312515258789E-0001  1.00708792002208E+0000  0.00000000000000E+0000
 7.86101659138997E-0001  1.00819147093145E+0000  0.00000000000000E+0000
 8.17890803019206E-0001  1.00923185367284E+0000  1.08420217248550E-0019
 8.49679946899414E-0001  1.01021492390171E+0000  0.00000000000000E+0000
 8.81469090779622E-0001  1.01114580282816E+0000  0.00000000000000E+0000
 9.13258234659831E-0001  1.01202899164833E+0000  0.00000000000000E+0000
 9.45047378540039E-0001  1.01286846490704E+0000  0.00000000000000E+0000
 9.76836522420247E-0001  1.01366774722825E+0000  1.08420217248550E-0019

 n_min = 2  k_min = 17
```

Таким образом, с помощью варьируемого кусочно-интерполяционного метода функция $u(x)=\sqrt[3]{\operatorname{arctg}\left(e^{\cos\sqrt[5]{(1/x)}}\right)}$ вычислена с точностью порядка 10^{-19} полиномами Ньютона 2-й степени при количестве подынтервалов $P=2^{17}$ (последняя строка вывода: `n_min = 2  k_min = 17`).

Значения функций из библиотеки стандратных программ, например, $u(x)=\sin(x)$ при $x\in[0,\ 1]$ может быть вычислены на основе представленной выше программы с фиксированным значением границы абсолютной погрешности $\varepsilon=10^{-19}$. Результат работы программы:

```
 1.55944824218750E-0002  1.55938503647020E-0002  0.00000000000000E+0000
 3.08532714843750E-0002  3.08483767205612E-0002  0.00000000000000E+0000
 4.77634006076389E-0002  4.77452419002790E-0002  3.38813178901720E-0021
 6.47176106770833E-0002  6.46724432632043E-0002  0.00000000000000E+0000
 8.16718207465278E-0002  8.15810552855052E-0002  0.00000000000000E+0000
 9.86260308159722E-0002  9.84662177835779E-0002  0.00000000000000E+0000
 1.15580240885417E-0001  1.15323077314111E-0001  0.00000000000000E+0000
 1.32534450954861E-0001  1.32146788569152E-0001  1.35525271560688E-0020
 1.49488661024306E-0001  1.48932515768833E-0001  0.00000000000000E+0000
```

```
 1.66442871093750E-0001  1.65675434051363E-0001  0.00000000000000E+0000
 1.83397081163194E-0001  1.82370730859872E-0001  1.35525271560688E-0020
 2.00351291232639E-0001  1.99013607325731E-0001  0.00000000000000E+0000
 2.17305501302083E-0001  2.15599279647927E-0001  0.00000000000000E+0000
 2.34259711371528E-0001  2.32122980468112E-0001  0.00000000000000E+0000
 2.51091851128472E-0001  2.48461719695437E-0001  0.00000000000000E+0000
 2.66350640190972E-0001  2.63212521581714E-0001  2.71050543121376E-0020
 2.81609429253472E-0001  2.77902040716229E-0001  0.00000000000000E+0000
 2.96868218315972E-0001  2.92526856995146E-0001  2.71050543121376E-0020
 3.12127007378472E-0001  3.07083565379146E-0001  0.00000000000000E+0000
 3.27385796440972E-0001  3.21568776686205E-0001  2.71050543121376E-0020
 3.42644585503472E-0001  3.35979118380688E-0001  0.00000000000000E+0000
 3.57903374565972E-0001  3.50311235358561E-0001  2.71050543121376E-0020
 3.73162163628472E-0001  3.64561790728548E-0001  2.71050543121376E-0020
 3.88420952690972E-0001  3.78727466589046E-0001  2.71050543121376E-0020
 4.03679741753472E-0001  3.92804964800621E-0001  0.00000000000000E+0000
 4.18938530815972E-0001  4.06791007753897E-0001  0.00000000000000E+0000
 4.34197319878472E-0001  4.20682339132672E-0001  2.71050543121376E-0020
 4.49456108940972E-0001  4.34475724672076E-0001  2.71050543121376E-0020
 4.64714898003472E-0001  4.48167952911584E-0001  0.00000000000000E+0000
 4.79973687065972E-0001  4.61755835942738E-0001  2.71050543121376E-0020
 4.95232476128472E-0001  4.75236210151369E-0001  2.71050543121376E-0020
 5.10491265190972E-0001  4.88605936954170E-0001  2.71050543121376E-0020
 5.25750054253472E-0001  5.01861903529441E-0001  5.42101086242752E-0020
 5.41008843315972E-0001  5.15001023541836E-0001  5.42101086242752E-0020
 5.56267632378472E-0001  5.28020237860942E-0001  0.00000000000000E+0000
 5.71526421440972E-0001  5.40916515273521E-0001  5.42101086242752E-0020
 5.86785210503472E-0001  5.53686853189263E-0001  0.00000000000000E+0000
 6.02043999565972E-0001  5.66328278339860E-0001  5.42101086242752E-0020
 6.17302788628472E-0001  5.78837847471265E-0001  0.00000000000000E+0000
 6.32561577690972E-0001  5.91212648028957E-0001  0.00000000000000E+0000
 6.47820366753472E-0001  6.03449798836059E-0001  5.42101086242752E-0020
 6.63079155815972E-0001  6.15546450764155E-0001  5.42101086242752E-0020
 6.78337944878472E-0001  6.27499787396636E-0001  0.00000000000000E+0000
 6.93596733940972E-0001  6.39307025684438E-0001  5.42101086242752E-0020
 7.08855523003472E-0001  6.50965416594011E-0001  5.42101086242752E-0020
 7.24114312065972E-0001  6.62472245747361E-0001  0.00000000000000E+0000
 7.39373101128472E-0001  6.73824834054031E-0001  0.00000000000000E+0000
 7.54631890190972E-0001  6.85020538334866E-0001  0.00000000000000E+0000
 7.69890679253472E-0001  6.96056751937406E-0001  0.00000000000000E+0000
 7.85149468315972E-0001  7.06930905342791E-0001  5.42101086242752E-0020
 8.00408257378472E-0001  7.17640466764009E-0001  0.00000000000000E+0000
 8.15667046440972E-0001  7.28182942735359E-0001  0.00000000000000E+0000
 8.30925835503472E-0001  7.38555878693003E-0001  0.00000000000000E+0000
 8.46184624565972E-0001  7.48756859546443E-0001  0.00000000000000E+0000
 8.61443413628472E-0001  7.58783510240824E-0001  0.00000000000000E+0000
 8.76702202690972E-0001  7.68633496309907E-0001  5.42101086242752E-0020
 8.91960991753472E-0001  7.78304524419592E-0001  5.42101086242752E-0020
 9.07219780815972E-0001  7.87794342901867E-0001  0.00000000000000E+0000
 9.22478569878472E-0001  7.97100742279059E-0001  0.00000000000000E+0000
 9.37737358940972E-0001  8.06221555778251E-0001  0.00000000000000E+0000
 9.52996148003472E-0001  8.15154659835768E-0001  5.42101086242752E-0020
 9.68254937065972E-0001  8.23897974591598E-0001  0.00000000000000E+0000
 9.83513726128472E-0001  8.32449464373636E-0001  0.00000000000000E+0000
 9.98772515190972E-0001  8.40807138171644E-0001  5.42101086242752E-0020

n_min = 3  k_min = 15
```

При этом, если сохранить коэффициенты полиномов в памяти компьютера, то временная сложность вычисления функции $u(x)=\sqrt[3]{\operatorname{arctg}\left(e^{\cos\sqrt[5]{(1/x)}}\right)}$ с точностью порядка 10^{-19} при использовании схемы Горнера – $T=2(t_c+t_y)$, где t_c, t_y – время бинарного сложения и умножения

соответственно, временная сложность вычисления функции $u(x)=\sin(x)$ с точностью порядка $\varepsilon=10^{-20}$ – $T=3(t_c+t_y)$.

П2.2. Приложение к п. 2.4. Листинг программы разностно-полиномиального решения системы ОДУ с вычислением начальных приближений узловых значений интерполяционных полиномов с помощью разностных методов Эйлера, Эйлера-Коши, Рунге-Кутты 4-го порядка, Бутчера 6-го порядка и Дормана-Принса 8-го порядка аппроксимации представлен непосредственно ниже. В программной реализации выбор разностного метода задается значением константы `method` от 1 до 5 соответственно порядку указанных выше методов. Правая часть системы ОДУ и начальные условия, заданные в программе, соответствуют задаче (2.47), интервал интегрирования – $x\in[\,1,10\,]$, для вывода абсолютной погрешности приближения использовано точное аналитическое решение задачи.

```
program DP_S2; {$APPTYPE CONSOLE} uses SysUtils;
const Anach=1; Bkonech=10; nn=13; method=5; //выборразностногометода
type matr=array[0..nn,0..nn] of extended;matr_C=array[-1..nn+1] of extended;
matric=array[0..10000] of matr_C; vect=array[0..nn] of extended;
var mm,p7,r,pod,iter,N_int:longint; m,i,j,l,n,Nmin,Kmin,k_:Byte;
 pp1,pp2,p1,p2: extended; ss1,ss2,t,xpr,tpr: extended;
 a00,b00,h,h0,h3,x3,y3_1,y3_2,lk1,lk2,a0,b0,max,min,Velpod,maximum:extended;
 bb1,bb2,X,Y1,Y2,A1,A2,fy1,fy2: vect; dy1,dy2,d: matr; Ck1,Ck2:matric;
procedure Viet; var k,l:byte; z:array[0..nn] of extended;
 e:array[0..nn,0..nn] of extended; begin for k:=0 to nn-1 do z[k]:=k;
 e[1,1]:=1; e[1,0]:=-z[0]; for k:=2 to nn do begin e[k,0]:=-e[k-1,0]*z[k-1];
 for l:=1 to k-1 do e[k,k-l]:=e[k-1,k-l-1]-e[k-1,k-l]*z[k-1]; e[k,k]:=e[k-1,k-1]
 end; for k:=1 to nn do for l:=0 to k do d[l,k]:=e[k,l] end;
function f1(x,y1,y2: extended):extended; begin f1:=x+2*y1/x-sqrt(y2) end;
function f2(x,y1,y2: extended):extended;begin f2:=2*sqrt(y2) end;
function fun1(x:extended):extended; begin fun1:=x+sqr(x) end;
function fun2(x:extended):extended; begin fun2:=sqr(x+1) end;
//вычисление значений конечных разностей для аппроксимации правой части
procedure Konech_Raznoct(var dy1,dy2:matr); var i,j:byte;
begin for j:=0 to n-1 do begin dy1[1,j]:=fy1[j+1]-fy1[j];
 dy2[1,j]:=fy2[j+1]-fy2[j] end; for i:=2 to n do for j:=0 to n-i do begin
dy1[i,j]:=dy1[i-1,j+1]-dy1[i-1,j]; dy2[i,j]:=dy2[i-1,j+1]-dy2[i-1,j] end; end;
function fapr(Koef:matr_C; xu:extended):extended;
var ll:Shortint; Si,tpr: extended; begin tpr:=(xu-Koef[-1])/h; Si:=Koef[n+1];
 for ll:=n downto 0 do Si:=tpr*Si+Koef[ll]; fapr:=Si end;
procedure Euler; var i:integer; begin for i:=1 to n do begin y1[i]:=y1[i-
1]+h*f1(x[i-1],y1[i-1],y2[i-1]); y2[i]:=y2[i-1]+h*f2(x[i-1],y1[i-1],y2[i-1]);
end; end;
procedure EulerCauchy; var i:integer;yT1,yT2:extended;
begin for i:=1 to n do begin yT1:= y1[i-1] + h*f1(x[i-1],y1[i-1],y2[i-1]);
```

```
yT2:=y2[i-1] + h*f2(x[i-1],y1[i-1],y2[i-1]); y1[i]:=y1[i-1]+h*(f1(x[i-1],y1[i-
1], y2[i-1])+f1(x[i],yT1,yT2))/2; y2[i]:= y2[i-1] + h*(f2(x[i-1],y1[i-1],y2[i-
1]) + f2(x[i],yT1,yT2))/2; end;end;
procedure RungeKutta;
var i:integer; z11,z12,z21,z22,z31,z32,z41,z42,x00,y100,y200:extended;
begin for i:=1 to n do begin z11:=h*f1(x[i-1],y1[i-1],y2[i-1]);z12:=h*f2(x[i-
1],y1[i-1],y2[i-1]); x00:=x[i-1]+h/2;y100:=y1[i-1]+z11/2;y200:=y2[i-1]+z12/2;
z21:=h*f1(x00,y100,y200); z22:=h*f2(x00,y100,y200); y100:=y1[i-1]+z21/2; y200:=
y2[i-1]+z22/2; z31:=h*f1(x00,y100,y200); z32:=h*f2(x00,y100,y200);x00:=x[i-1]+h;
y100:=y1[i-1]+z31; y200:=y2[i-1]+z32; z41:=h*f1(x00,y100,y200);
z42:=h*f2(x00,y100,y200); y1[i]:= y1[i-1] + 1/6 * (z11+2*z21+2*z31+z41);
y2[i]:= y2[i-1] + 1/6 * (z12+2*z22+2*z32+z42); end;end;
procedure Butcher; const c2=1/2; c3=2/3; c4=1/3; c5=5/6; c6=1/6; c7=1;
 b1=13/200; b2=0; b3=11/40; b4=11/40; b5=4/25; b6=4/25; b7=13/200; a21=1/2;
 a31=2/9; a32=4/9; a41=7/36; a42=2/9; a43=-1/12; a51=-35/144; a52=-55/36;
a53=35/48; a54=15/8; a61=-1/360; a62=-11/36; a63=-1/8; a64=1/2; a65=1/10;
a71=-41/260; a72=22/13; a73=43/156; a74=-118/39; a75=32/195; a76=80/39;
var i:integer;
G11,G12,G21,G22,G31,G32,G41,G42,G51,G52,G61,G62,G71,G72,z21,z22,z23:extended;
z31,z32,z33,z41,z42,z43,z51,z52,z53,z61,z62,z63,z71,z72,z73:extended;
begin for i:=1 to n do begin G11:=h*f1(x[i-1],y1[i-1],y2[i-1]); G12:=h*f2(x[i-
1],y1[i-1],y2[i-1]);    z21:=x[i-1]+c2*h;    z22:=y1[i-1]+a21*G11;    z23:=y2[i-
1]+a21*G12; G21:=h*f1(z21,z22,z23); G22:= h*f2(z21,z22,z23); z31:=x[i-1]+c3*h;
z32:=y1[i-1]+a31*G11+a32*G21; z33:=y2[i-1]+a31*G12+a32*G22;
G31:=h*f1(z31,z32,z33); G32:=h*f2(z31,z32,z33); z41:=x[i-1]+c4*h;
 z42:=y1[i-1]+a41*G11+a42*G21+a43*G31; z43:=y2[i-1]+a41*G12+a42*G22+a43*G32;
G41:=h*f1(z41,z42,z43); G42:=h*f2(z41,z42,z43); z51:=x[i-1]+c5*h;
z52:=y1[i-1]+a51*G11+a52*G21+a53*G31+a54*G41;   z53:=y2[i-1]+a51*G12+   a52*G22+
a53*G32+  a54*G42;  G51:=h*f1(z51,z52,z53);G52:=h*f2(z51,z52,z53);  z61:=x[i-1]+
c6*h;   z62:=   y1[i-1]+a61*G11+a62*G21+a63*G31+a64*G41+a65*G51;   z63:=y2[i-1]+
a61*G12+    a62*G22+a63*G32+a64*G42+a65*G52;    G61:=h*f1(z61,z62,z63);    G62:=
h*f2(z61,z62,z63); z71:=x[i-1]+c7*h; z72:=y1[i-1]+ a71*G11+ a72*G21+ a73*G31+
a74*G41+a75*G51+a76*G61; z73:=y2[i-1]+a71*G12+a72*G22+a73*G32+a74*G42+ a75*G52+
a76*G62; G71:=h*f1(z71,z72,z73); G72:=h*f2(z71,z72,z73);y1[i]:=y1[i-1]+b1*G11 +
b2*G21 +b3*G31+b4*G41+b5*G51+b6*G61+b7*G71; y2[i]:= y2[i-1] + b1*G12+b2*G22+
b3*G32+ b4*G42+b5*G52+b6*G62+b7*G72; end;end;
procedure Dormand_Prince; const c1=0; c2=1.0/18.0; c3=1.0/12.0; c4=1.0/8.0;
c5=5.0/16.0;   c6=3.0/8.0;   c7=59.0/400.0;   c8=93.0/200.0;   c9=5490023248.0/
9719169821.0; c10=13.0/20.0; c11=1201146811.0/1299019798.0; c12=1.0; c13=1.0;
a21=1.0/18.0; a31=1.0/48.0; a32=1.0/16.0; a41=1.0/32.0; a42=0; a43=3.0/32.0;
a51=5.0/16.0; a52=0; a53=-75.0/64.0; a54=75.0/64.0; a61=3.0/80.0; a62=0; a63=0;
a64=3.0/16.0;   a65=3.0/20.0;   a71=29443841.0/614563906.0;   a72=0;   a73=0;
a74=77736538.0/692538347.0;    a75=-28693883.0/1125000000.0;    a76=23124283.0/
1800000000.0;  a81=16016141.0/946692911.0;  a82=0;  a83=0;  a84=61564180.0/
158732637.0;  a85=22789713.0/633445777.0;  a86=545815736.0/2771057229.0;  a87=-
180193667.0 /1043307555.0;  a91=39632708.0/573591083.0;  a92=0;  a93=0;  a94=-
433636366.0/683701615.0;    a95=-421739975.0/2616292301.0;    a96=100302831.0/
723423059.0;    a97=790204164.0/839813087.0;    a98=800635310.0/3783071287.0;
a101=246121993.0/1340847787.0;    a102=0;    a103=0;    a104=-37695042795.0/
15268766246.0;  a105=-309121744.0/1061227803.0;  a106=-12992083.0/490766935.0;
a107=6005943493.0/2108947869.0; a108=393006217.0/1396673457.0; a109=123872331.0/
1001029789.0; a111=-1028468189.0/846180014.0; a112=0; a113=0; a114=8478235783.0/
508512852.0;  a115=1311729495.0/1432422823.0;  a116=-10304129995.0/1701304382.0;
a117=-48777925059.0/3047939560.0;    a118=15336726248.0/1032824649.0;    a119=-
45442868181.0/3398467696.0;  a1110=3065993473.0/597172653.0;  a121=185892177.0/
718116043.0;    a122=0;    a123=0;    a124=-3185094517.0/667107341.0;    a125=-
477755414.0/1098053517.0;   a126=-703635378.0/230739211.0;   a127=5731566787.0/
1027545527.0;  a128=5232866602.0/850066563.0;  a129=-4093664535.0/808688257.0;
a1210=3962137247.0/1805957418.0; a1211=65686358.0/487910083.0; a131=403863854.0/
491063109.0; a132=0; a133=0; a134=-5068492393.0/434740067.0; a135=-411421997.0/
543043805.0;    a136=652783627.0/914296604.0;    a137=11173962825.0/925320556.0;
a138=-13158990841.0/6184727034.0;    a139=3936647629.0/1978049680.0;    a1310=-
```

```
160528059.0/      685178525.0;      a1311=248638103.0/1413531060.0;      a1312=0;
b1=13451932.0/455176623.0; b2=0; b3=0; b4=0; b5=0; b6=-808719846.0/976000145.0;
b7=1757004468.0/5645159321.0;    b8=656045339.0/265891186.0;    b9=-3867574721.0/
1518517206.0; b10=465885868.0/322736535.0; b11=53011238.0/667516719.0; b12=2.0/
45.0; b13=0; var i:integer;
Q11,Q21,Q31,Q41,Q51,Q61,Q71,Q81,Q91,Q101,Q111,Q121,Q131:extended;
Q12,Q22,Q32,Q42,Q52,Q62,Q72,Q82,Q92,Q102,Q112,Q122,Q132:extended;
 yy1_4, yy1_5,yy1_6,yy1_7,yy1_8,yy1_9,yy1_10,yy1_11,yy1_12,yy1_13:extended;
 yy2_4, yy2_5,yy2_6,yy2_7,yy2_8,yy2_9,yy2_10,yy2_11,yy2_12,yy2_13:extended;
begin for i:=1 to n do begin
Q11:= h*f1(x[i-1],yy1[i-1],yy2[i-1]); Q12:= h*f2(x[i-1],yy1[i-1],yy2[i-1]);
Q21:= h*f1(x[i-1]+c2 *h, yy1[i-1]+ a21*Q11, yy2[i-1] +a21*Q12);
Q22:= h*f2(x[i-1]+c2 *h, yy1[i-1]+ a21*Q11, yy2[i-1] +a21*Q12);
Q31:= h*f1(x[i-1]+c3 *h, yy1[i-1]+ a31*Q11+a32*Q21, yy2[i-1]+ a31*Q12+a32*Q22);
Q32:= h*f2(x[i-1]+c3 *h, yy1[i-1]+ a31*Q11+a32*Q21, yy2[i-1]+ a31*Q12+a32*Q22);
yy1_4:=yy1[i-1]+a41*Q11+a42*Q21+a43*Q31;
yy2_4:=yy2[i-1]+a41*Q12+a42*Q22+a43*Q32;
Q41:= h*f1(x[i-1]+c4 *h, yy1_4, yy2_4); Q42:= h*f2(x[i-1]+c4 *h, yy1_4, yy2_4);
yy1_5:= yy1[i-1] + a51 *Q11 +a52 *Q21 +a53 *Q31 +a54 *Q41;
yy2_5:= yy2[i-1] + a51 *Q12 +a52 *Q22 +a53 *Q32 +a54 *Q42;
Q51:= h*f1(x[i-1]+c5 *h, yy1_5, yy2_5); Q52:= h*f2(x[i-1]+c5 *h, yy1_5, yy2_5);
yy1_6:= yy1[i-1] + a61 *Q11 +a62 *Q21 +a63 *Q31 +a64 *Q41 +a65 *Q51;
yy2_6:= yy2[i-1] + a61 *Q12 +a62 *Q22 +a63 *Q32 +a64 *Q42 +a65 *Q52;
Q61:= h*f1(x[i-1]+c6 *h, yy1_6, yy2_6);
Q62  := h*f2(x[i-1]+c6 *h, yy1_6, yy2_6);
yy1_7:= yy1[i-1] + a71 *Q11 +a72 *Q21 +a73 *Q31 +a74 *Q41 +a75 *Q51 +a76 *Q61;
yy2_7:= yy2[i-1] + a71 *Q12 +a72 *Q22 +a73 *Q32 +a74 *Q42 +a75 *Q52 +a76 *Q62;
Q71:= h*f1(x[i-1]+c7 *h, yy1_7, yy2_7);
Q72:= h*f2(x[i-1]+c7 *h, yy1_7, yy2_7);
yy1_8:=yy1[i-1]+a81*Q11+a82*Q21+a83*Q31+a84*Q41+a85*Q51+a86*Q61+a87*Q71;
yy2_8:=yy2[i-1]+a81*Q12+a82*Q22+a83*Q32+a84*Q42+a85*Q52+a86*Q62+a87*Q72;
Q81:= h*f1(x[i-1]+c8 *h, yy1_8, yy2_8);
Q82:= h*f2(x[i-1]+c8 *h, yy1_8, yy2_8);
yy1_9:=yy1[i-1] + a91 *Q11 +a92 *Q21 +a93 *Q31 +a94 *Q41 +a95 *Q51 +a96 *Q61
+a97 *Q71 +a98 *Q81; yy2_9 := yy2[i-1] + a91 *Q12 +a92 *Q22 +a93 *Q32 +a94 *Q42
+a95 *Q52 +a96 *Q62 +a97 *Q72 +a98 *Q82; Q91:= h*f1(x[i-1]+c9 *h, yy1_9, yy2_9);
Q92:= h*f2(x[i-1]+c9 *h, yy1_9, yy2_9); yy1_10:= yy1[i-1] + a101*Q11 +a102*Q21
+a103*Q31 +a104*Q41 +a105*Q51 +a106*Q61 +a107*Q71 +a108*Q81 +a109*Q91; yy2_10:=
yy2[i-1] + a101*Q12 +a102*Q22 +a103*Q32 +a104*Q42 +a105*Q52 +a106*Q62 +a107*Q72
+a108*Q82 +a109*Q92; Q101 := h*f1(x[i-1]+c10*h, yy1_10, yy2_10);
Q102:=  h*f2(x[i-1]+c10*h,  yy1_10,  yy2_10);  yy1_11:=  yy1[i-1]  +  a111*Q11
+a112*Q21 +a113*Q31 +a114*Q41 +a115*Q51 +a116*Q61 +a117*Q71 +a118*Q81 +a119*Q91
+a1110*Q101;  yy2_11:=  yy2[i-1]  +  a111*Q12  +a112*Q22  +a113*Q32  +a114*Q42
+a115*Q52 +a116*Q62 +a117*Q72 +a118*Q82 +a119*Q92 +a1110*Q102; Q111 := h*f1(x[i-
1]+c11*h, yy1_11, yy2_11); Q112:= h*f2(x[i-1]+c11*h, yy1_11, yy2_11); yy1_12:=
yy1[i-1] + a121*Q11 +a122*Q21 +a123*Q31 +a124*Q41 +a125*Q51 +a126*Q61 +a127*Q71
+a128*Q81  +a129*Q91  +a1210*Q101  +a1211*Q111;  yy2_12:=  yy2[i-1]  +  a121*Q12
+a122*Q22 +a123*Q32 +a124*Q42 +a125*Q52 +a126*Q62 +a127*Q72 +a128*Q82 +a129*Q92
+a1210*Q102  +a1211*Q112;  Q121  :=  h*f1(x[i-1]+c12*h,  yy1_12,  yy2_12);  Q122:=
h*f2(x[i-1]+c12*h,  yy1_12,  yy2_12);  yy1_13:=  yy1[i-1]  +  a131*Q11  +a132*Q21
+a133*Q31   +a134*Q41   +a135*Q51   +a136*Q61   +a137*Q71   +a138*Q81   +a139*Q91
+a1310*Q101  +a1311*Q111  +a1312*Q121;  yy2_13:=  yy2[i-1]  +  a131*Q12  +a132*Q22
+a133*Q32   +a134*Q42   +a135*Q52   +a136*Q62   +a137*Q72   +a138*Q82   +a139*Q92
+a1310*Q102 +a1311*Q112 +a1312*Q122;
Q131:=h*f1(x[i-1]+c13*h,yy1_13,yy2_13);Q132:=h*f2(x[i-1]+c13*h, yy1_13, yy2_13);
yy1[i]:= yy1[i-1] + b1*Q11+ b2*Q21+ b3*Q31+ b4*Q41+ b5*Q51+ b6*Q61+ b7*Q71+
b8*Q81+  b9*Q91+  b10*Q101+  b11*Q111+  b12*Q121+b13*Q131;  yy2[i]:=  yy2[i-1]  +
b1*Q12+  b2*Q22+  b3*Q32+  b4*Q42+  b5*Q52+  b6*Q62+  b7*Q72+  b8*Q82+  b9*Q92+
b10*Q102+ b11*Q112+ b12*Q122+ b13*Q132; end; end;
begin Viet; Velpod:=1; {величина интервала} Iter:=10; {количество итераций}
a0:=Anach; b0:=Anach+Velpod;N_int:=0; lk1:=2; lk2:=4; //начальныеусловия
while a0 < Bkonech do begin Min:=10; k_:=0;
```

```
repeat n:=1; repeat max:=0; h0:=(b0-a0)/exp(k_*ln(2)); a00:=a0; b00:=a00+h0;
y1[0]:=lk1; y2[0]:=lk2; x[0]:=a0; mm:=0; pod:=0;
while a00<=b0-h0 do begin h:=(b00-a00)/n; for j:=1 to n do begin mm:=mm+1;
x[j]:=a0+mm*h; end; case method of 1: Euler; 2: EulerCauchy;3: RungeKutta;4:
Butcher; 5: Dormand_Prince end;
for m:=1 to iter do begin for i:=0 to n do begin fy1[i]:=f1(x[i],y1[i],y2[i]);
fy2[i]:=f2(x[i],y1[i],y2[i]); end; Konech_Raznoct(dy1,dy2); p7:=1; for j:=1 to n
do begin p7:=p7*j; bb1[j]:=dy1[j,0]/p7; bb2[j]:=dy2[j,0]/p7; end; a1[0]:=fy1[0];
a2[0]:=fy2[0]; for l:=1 to n do begin ss1:=0;ss2:=0;
for j:=l to n do begin ss1:=ss1+d[l,j]*bb1[j]; ss2:=ss2+d[l,j]*bb2[j] end;
a1[l]:=ss1; a2[l]:=ss2; end;
for i:=1 to n do begin t:=(x[i]-x[0])/h; pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1);
for r:=n-1 downto 0 do begin pp1:=pp1*t+a1[r]/(r+1); pp2:=pp2*t+a2[r]/(r+1);
end; y1[i]:=pp1*h*t+y1[0]; y2[i]:=pp2*h*t+y2[0] end end;
h3:=h/3; x3:=a00; while x3<=b00 do begin t:=(x3-x[0])/h; pp1:=a1[n]/(n+1);
pp2:=a2[n]/(n+1); for r:=n-1 downto 0 do begin pp1:=pp1*t+a1[r]/(r+1);
pp2:=pp2*t+a2[r]/(r+1) end; y3_1:=pp1*h*t+y1[0]; y3_2:=pp2*h*t+y2[0]; p1:=a1[n];
p2:=a2[n]; for r:=n-1 downto 0 do begin p1:=p1*t+a1[r]; p2:=p2*t+a2[r]; end; if
(abs(p1-f1(x3,y3_1,y3_2))+abs(p2-f2(x3,y3_1,y3_2))) > max then begin
max:=abs(p1-f1(x3,y3_1,y3_2))+abs(p2-f2(x3,y3_1,y3_2)); end; x3:=x3+h3; end;
Y1[0]:=Y1[n]; Y2[0]:=Y2[n]; x[0]:=x[n]; pod:=pod+1; a00:=a00+h0; b00:=a00+h0;
end; if max<Min then begin Min:=max; Nmin:=n; Kmin:=k_ end; n:=n+1; until n>5;
k_:=k_+1; until k_>6;
n:=Nmin; k_:=Kmin; h0:=(b0-a0)/exp(k_*ln(2)); a00:=a0; b00:=a00+h0; y1[0]:=lk1;
y2[0]:=lk2; mm:=0; pod:=0; x[0]:=a0; while a00<=b0-h0 do begin h:=(b00-a00)/n;
for j:=1 to n do begin mm:=mm+1; x[j]:=a0+mm*h; end;
case method of 1: Euler; 2: EulerCauchy; 3: RungeKutta; 4: Butcher; 5:
Dormand_Prince end;
for m:=1 to iter do begin for i:=0 to n do begin fy1[i]:=f1(x[i],y1[i],y2[i]);
fy2[i]:=f2(x[i],y1[i],y2[i]); end; Konech_Raznoct(dy1,dy2);
 p7:=1; for j:=1 to n do begin p7:=p7*j; bb1[j]:=dy1[j,0]/p7;
bb2[j]:=dy2[j,0]/p7; end; a1[0]:=fy1[0]; a2[0]:=fy2[0];
for l:=1 to n do begin ss1:=0; ss2:=0;
for j:=l to n do begin ss1:=ss1+d[l,j]*bb1[j]; ss2:=ss2+d[l,j]*bb2[j] end;
a1[l]:=ss1; a2[l]:=ss2; end; for i:=1 to n do begin t:=(x[i]-x[0])/h;
pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1); for r:=n-1 downto 0 do begin
pp1:=pp1*t+a1[r]/(r+1); pp2:=pp2*t+a2[r]/(r+1); end;
y1[i]:=pp1*h*t+y1[0]; y2[i]:=pp2*h*t+y2[0]; end; end;
Ck1[pod,0]:=y1[0]; Ck2[pod,0]:=y2[0]; Ck1[pod,-1]:=x[0]; Ck2[pod,-1]:=x[0];
for i:=1 to n+1 do begin Ck1[pod,i]:=a1[i-1]*h/i; Ck2[pod,i]:=a2[i-1]*h/i end;
Y1[0]:=Y1[n]; Y2[0]:=Y2[n]; x[0]:=x[n]; pod:=pod+1; a00:=a00+h0; b00:=a00+h0;
end; pod:=trunc((b0-a0)/(h*n))-1; lk1:=fapr(Ck1[pod],b0);
lk2:=fapr(Ck2[pod],b0);
xpr:= a0+0.03*(a0-1); if xpr<>b0 then pod:=trunc((xpr-a0)/(h*n)) else
pod:=trunc((xpr-a0)/(h*n))-1; maximum:=abs(fapr(Ck1[pod],xpr)-fun1(xpr));
if abs(fapr(Ck2[pod],xpr) -fun2(xpr))>maximum then
maximum:=abs(fapr(Ck2[pod],xpr)-fun2(xpr));
writeln('i= ',N_int,' N= ',n,' k=',k_,' ',xpr,' ',maximum); a0:=a0+Velpod;
b0:=b0+Velpod; inc(N_int); end; writeln; readln; end.
```

Результат работы программы при значении константы `method=5`:

```
i=0 N= 2 k=2   1.00  0.00000000000000E+0000
i=1 N= 4 k=0   2.03  4.33680868994202E-0019
i=2 N= 4 k=0   3.06  8.67361737988404E-0019
i=3 N= 2 k=0   4.09  0.00000000000000E+0000
i=4 N= 1 k=1   5.12  0.00000000000000E+0000
i=5 N= 1 k=0   6.15  3.46944695195361E-0018
i=6 N= 1 k=0   7.18  6.93889390390723E-0018
i=7 N= 1 k=0   8.21  0.00000000000000E+0000
i=8 N= 1 k=0   9.24  0.00000000000000E+0000
```

В 1-й колонке – номер интервала i из разбиения отрезка интегрирования (длина интервала фиксирована и определяется значением переменной `Velpod`), во 2-й – степень n интерполяционных полиномов на i-м интервале разбиения, в 3-й – значение параметра k, определяющего число подынтервалов $P = 2^k$ на i-м интервале разбиения, в 4-й – аргумент, в 5-й – абсолютная погрешность разностно-полиномиального приближения. Значения абсолютных погрешностей приближения при разностном вычислении узловых значений по методам Эйлера, Эйлера-Коши, Рунге-Кутты и Бутчера приведены в табл. 2.3 (в программе соответственно заданы значения константы `method=1,2,3,4`). Погрешность приближения во всех разновидностях метода практически одинакова (табл. 2.3) и на один и более десятичный порядок ниже погрешности приближения, полученной по известным разностным методам (табл. 2.2). При этом на основе сравнения программно определеных значений варьируемых параметров n и k в случае `method=5` и значений тех же параметров, полученных при использовании метода Эйлера: $n = 1, 2, 3, 4$, $k = 2, 3, 4, 5$ (табл. 2.4) можно отметить, что при использовании в качестве входных приближений значений, полученных по методу высшего порядка, аналогичные порядки погрешности приближения достигаются при меньшем числе подынтервалов разбиения.

Приближенное решение задачи Коши [35]:

$$\left.\begin{aligned} &y_1' = 2x\,y_1 \ln\left(\max\left(y_2, 10^{-3}\right)\right),\ \ y_2' = -2x\,y_2 \ln\left(\max\left(y_1, 10^{-3}\right),\right.\\ &y_1(0) = 1,\ y_2(0) = e, \end{aligned}\right\} \qquad (\text{П}2.1)$$

с точным аналитическим решением $y_1 = e^{\sin(x^2)}$, $y_2 = e^{\cos(x^2)}$ на отрезке $x \in [0, 10]$ по представленной выше программе при значении константы `method=1` дает следующий результат:

```
i= 0 N= 6 k=8  0.00  0.00000000000000E+0000
i= 1 N= 6 k=9  1.03  2.16840434497101E-0018
i= 2 N= 8 k=8  2.06  4.60785923306339E-0019
i= 3 N= 8 k=8  3.09  1.08420217248550E-0018
i= 4 N= 8 k=8  4.12  8.13151629364128E-0019
i= 5 N= 8 k=9  5.15  4.11996825544492E-0018
```

```
i= 6 N= 8 k=9  6.18  3.36102673470506E-0018
i= 7 N= 8 k=9  7.21  2.98155597433514E-0018
i= 8 N= 8 k=9  8.24  4.77048955893622E-0018
i= 9 N= 8 k=9  9.27  5.42101086242752E-0019
```

Результат работы программы разностно-полиномиального решения той же задачи при использовании метода Дормана-Принса (`method=5`):

```
i= 0 N= 6 k=8  0.00  0.00000000000000E+0000
i= 1 N= 6 k=9  1.03  2.16840434497101E-0018
i= 2 N= 8 k=7  2.06  4.60785923306339E-0019
i= 3 N= 8 k=8  3.09  6.50521303491303E-0019
i= 4 N= 8 k=9  4.12  3.52365706057789E-0019
i= 5 N= 8 k=9  5.15  2.81892564846231E-0018
i= 6 N= 8 k=9  6.18  1.84314369322536E-0018
i= 7 N= 8 k=9  7.21  1.62630325872826E-0018
i= 8 N= 8 k=9  8.24  3.68628738645072E-0018
i= 9 N= 8 k=9  9.27  3.79470760369927E-0019
```

Использование разностного метода Дормана-Принса 8-го порядка для получения начальных приближений узловых значений интерполяционных полиномов с увеличением трудоемкости позволило лишь незначительно снизить погрешность приближения. При этом программа варьируемого разностно-полиномиального метода для всех значений константы `method` дает приближение решения задачи (П2.1) на отрезке $x \in [0, 10]$ с границей абсолютной погрешности порядка 10^{-18} (табл. П2.1).

Таблица П2.1

Погрешность варьируемого разностно-полиномиального решения задачи (П2.1)

x	`method=1`	`method=2`	`method=3`	`method=4`	`method=5`
1.03	2.168**e-18**	2.602**e-18**	2.602**e-18**	2.602**e-18**	2.168**e-18**
2.06	4.608**e-19**	8.674**e-19**	8.674**e-19**	8.674**e-19**	4.608**e-19**
…	…	…	…	…	…
7.21	2.982**e-18**	6.939**e-18**	6.939**e-18**	6.939**e-18**	1.626**e-18**
8.24	4.770**e-18**	2.168**e-18**	2.602**e-18**	2.602**e-18**	3.686**e-18**
9.27	5.421**e-19**	1.518**e-18**	1.464**e-18**	1.464**e-18**	3.795**e-19**

Для сравнения погрешность наилучшего приближения решения этой же задачи разностными методами, программные реализации которых представлены в п. П2.2 приложения, представлена в табл. П2.2.

Таблица П2.2

Погрешность решения задачи (П2.1) разностными методами

x	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*
	$h = 1.03\times10^{-8}$	$h = 1.03\times10^{-8}$	$h = 1.03\times10^{-5}$	$h = 1.03\times10^{-4}$	$h = 1.03\times10^{-3}$

1.03	2.034**e-08**	4.152**e-16**	3.383**e-17**	9.649**e-18**	7.156**e-18**
2.06	2.562**e-08**	3.327**e-16**	4.987**e-18**	5.313**e-18**	4.012**e-18**
…	…	…	…	…	…
7.21	6.819**e-06**	1.892**e-14**	1.175**e-16**	2.364**e-17**	3.068**e-17**
8.24	1.828**e-06**	5.476**e-14**	5.866**e-17**	2.439**e-17**	7.698**e-17**
9.27	2.011**e-06**	4.145**e-14**	4.949**e-17**	6.830**e-18**	4.215**e-17**

Граница абсолютной погрешности порядка 10^{-17} соответствует методам Бутчера 6-го и Дормана-Принса 8-го порядков аппроксимации, при этом дальнейшее уменьшение величины разностного шага приводит к снижению точности приближения.

Таким образом, представлены программные реализации разновидностей варьируемого разностно-полиномиального решения задачи Коши для системы ОДУ с разностным вычислением узловых значений при помощи методов Эйлера, Эйлера-Коши, Рунге-Кутты 4-го порядка, Бутчера (соответственно, Рунге-Кутты 6-го порядка), Дормана-Принса (соответственно, Рунге-Кутты 8-го порядка). Результаты экспериментов подтверждают, что точность приближения при вариации числа подынтервалов и степени интерполяционного полинома во всех данных разновидностях метода практически совпадает, при этом она более чем на один десятичный порядок превышает точность использованных разностных методов, программные реализации которых также представлены в приложении. Кроме того, дана программная реализация метода варьируемого кусочно-интерполяционного приближения функций одной действительной переменной. Представлен пример вычисления композиции стандартных функций с точностью порядка 10^{-19} на основе кусочной интерполяции полиномами Ньютона второй степени.

П2.3. Приложение к п. 2.5. Абсолютные погрешности приближения явными методами Рунге-Кутты различного порядка и время работы программ, приведенные в таблицах сравнений в главах 2, 3, 5 диссертации, рассчитаны с помощью представленных ниже программных реализаций и их видоизменений в соответствии с числом уравнений в системе ОДУ. Значения коэффициентов разностных методов, использованных в программных реализациях,

заимствованы из [35]. Коэффициенты метода Дормана-Принса 8-го порядка аппроксимации представлены непосредственно ниже.

Коэффициенты метода Дормана и Принса 8-го порядка

0													
$\frac{1}{18}$	$\frac{1}{18}$												
$\frac{1}{12}$	$\frac{1}{48}$	$\frac{1}{16}$											
$\frac{1}{8}$	$\frac{1}{32}$	0	$\frac{3}{32}$										
$\frac{5}{16}$	$\frac{5}{16}$	0	$\frac{-75}{64}$	$\frac{75}{64}$									
$\frac{3}{8}$	$\frac{3}{80}$	0	0	$\frac{3}{16}$	$\frac{3}{20}$								
$\frac{59}{400}$	$\frac{29443841}{614563906}$	0	0	$\frac{77736538}{692538347}$	$\frac{-28693883}{1125000000}$	$\frac{23124283}{1800000000}$							
$\frac{93}{200}$	$\frac{16016141}{946692911}$	0	0	$\frac{61564180}{158732637}$	$\frac{22789713}{633445777}$	$\frac{545815736}{2771057229}$	$\frac{-180193667}{1043307555}$						
$\frac{5490023248}{9719169821}$	$\frac{39632708}{573591083}$	0	0	$\frac{-433636366}{683701615}$	$\frac{-421739975}{2616292301}$	$\frac{100302831}{723423059}$	$\frac{790204164}{839813087}$	$\frac{800635310}{3783071287}$					
$\frac{13}{20}$	$\frac{246121993}{1340847787}$	0	0	$\frac{-37695042795}{15268766246}$	$\frac{-309121744}{1061227803}$	$\frac{-12992083}{490766935}$	$\frac{6005943493}{2108947869}$	$\frac{393006217}{1396673457}$	$\frac{123872331}{1001029789}$				
$\frac{1201146811}{1299019798}$	$\frac{-1028468189}{846180014}$	0	0	$\frac{8478235783}{508512852}$	$\frac{1311729495}{1432422823}$	$\frac{-10304129995}{1701304382}$	$\frac{-48777925059}{3047939560}$	$\frac{15336726248}{1032824649}$	$\frac{-45442868181}{3398467696}$	$\frac{3065993473}{597172653}$			
1	$\frac{185892177}{718116043}$	0	0	$\frac{-3185094517}{667107341}$	$\frac{-477755414}{1098053517}$	$\frac{-703635378}{230739211}$	$\frac{5731566787}{1027545527}$	$\frac{5232866602}{850066563}$	$\frac{-4093664535}{808688257}$	$\frac{3962137247}{1805957418}$	$\frac{65686358}{487910083}$		
1	$\frac{403863854}{491063109}$	0	0	$\frac{-5068492393}{434740067}$	$\frac{-411421997}{543043805}$	$\frac{652783627}{914296604}$	$\frac{11173962825}{925320556}$	$\frac{-13158990841}{6184727034}$	$\frac{3936647629}{1978049680}$	$\frac{-160528059}{685178525}$	$\frac{248638103}{1413531060}$	0	
	$\frac{13451932}{455176623}$	0	0	0	0	$\frac{-808719846}{976000145}$	$\frac{1757004468}{5645159321}$	$\frac{656045339}{265891186}$	$\frac{-3867574721}{1518517206}$	$\frac{465885868}{322736535}$	$\frac{53011238}{667516719}$	$\frac{2}{45}$	0

Метод Эйлера решения задачи Коши для системы двух ОДУ:

```
program EulerSystemof2ode; {$APPTYPE CONSOLE} uses SysUtils;
var x,y1,y2,k1,k2,maxErr,h,Anach,Bkonech:extended; kk,k:int64;
 hour,minut,sec,msec:word; tnach,tkonech:extended;
function f1(x,y1,y2: extended):extended; begin f1:=x+2*y1/x-sqrt(y2) end;
function f2(x,y1,y2: extended):extended; begin f2:=2*sqrt(y2) end;
function fun1(x:extended):extended; begin fun1:=x+sqr(x) end;
function fun2(x:extended):extended; begin fun2:=sqr(x+1) end;
begin tnach:=Gettime; Anach:=1; Bkonech:=10;y1:=2; y2:=4; h:=1.03e-9;
kk:=0; x:=Anach; k:=1; while x<Bkonech do begin
k1:=f1(x,y1,y2); k2:=f2(x,y1,y2);
y1:= y1 + h*k1; y2:= y2 + h*k2; x:=Anach+k*h; inc(k);
inc(kk); if kk=1e+9 then begin maxErr:= abs(y1-fun1(x));
if abs(y2-fun2(x))>maxErr then maxErr:= abs(y2-fun2(x)); writeln(x,' ',maxErr);
kk:=0; end; end; tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln;
 writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec); readln; end.
```

Метод Эйлера-Коши решения задачи Коши для системы двух ОДУ:

```
program EulerCauchySystemof2ode; {$APPTYPE CONSOLE} uses SysUtils;
var yt1,yt2,k1,k2:extended; x,y1,y2,maxErr,h,Anach,Bkonech:extended;
kk,k:int64; hour,minut,sec,msec:word; tnach,tkonech:extended;
function f1(x,y1,y2: extended):extended; begin f1:=x+2*y1/x-sqrt(y2) end;
function f2(x,y1,y2: extended):extended; begin f2:=2*sqrt(y2) end;
function fun1(x:extended):extended; begin fun1:=x+sqr(x) end;
function fun2(x:extended):extended; begin fun2:=sqr(x+1) end;
begin tnach:=Gettime; Anach:=1; Bkonech:=10;y1:=2; y2:=4; h:=1.03e-9; kk:=0;
x:=Anach; k:=1; while x<Bkonech do begin yT1:= y1 + h*f1(x,y1,y2);
```

```
yT2:= y2 + h*f2(x,y1,y2); k1:=f1(x,y1,y2)+f1(x+h,yT1,yT2);
k2:=f2(x,y1,y2)+f2(x+h,yT1,yT2); y1:= y1 + h*k1/2; y2:= y2 + h*k2/2;
x:=Anach+k*h; inc(k); inc(kk); if kk=1e+9 then begin maxErr:= abs(y1-fun1(x));
if abs(y2-fun2(x))>maxErr then maxErr:= abs(y2-fun2(x)); writeln(x,' ',maxErr);
kk:=0; end; end; tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln;
 writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec); readln; end.
```

Метод Рунге-Кутты 4-го порядка решения задачи Коши для системы двух ОДУ:

```
program RungeKuttSystemof2ode; {$APPTYPE CONSOLE} uses SysUtils;
var k11,k12,k21,k22,k31,k32,k41,k42,x00,y100,y200,tnach,tkonech:extended;
 x,y1,y2,maxErr,h,Anach,Bkonech:extended; kk,k:int64; hour,minut,sec,msec:word;
function f1(x,y1,y2: extended):extended; begin f1:=x+2*y1/x-sqrt(y2) end;
function f2(x,y1,y2: extended):extended; begin f2:=2*sqrt(y2) end;
function fun1(x:extended):extended; begin fun1:=x+sqr(x) end;
function fun2(x:extended):extended; begin fun2:=sqr(x+1) end;
begin tnach:=Gettime; Anach:=1; Bkonech:=10;y1:=2; y2:=4; h:=1.03e-5;
kk:=0; x:=Anach; k:=1;
while x<Bkonech do begin k11:=h*f1(x,y1,y2);
k12:=h*f2(x,y1,y2); x00:=x+h/2; y100:=y1+k11/2; y200:=y2+k12/2;
k21:=h*f1(x00,y100,y200); k22:=h*f2(x00,y100,y200); y100:=y1+k21/2;
y200:=y2+k22/2; k31:=h*f1(x00,y100,y200); k32:=h*f2(x00,y100,y200); x00:=x+h;
y100:=y1+k31; y200:=y2+k32; k41:=h*f1(x00,y100,y200); k42:=h*f2(x00,y100,y200);
y1:= y1 + 1/6 * (k11+2*k21+2*k31+k41); y2:= y2 + 1/6 * (k12+2*k22+2*k32+k42);
x:=Anach+k*h; inc(k); inc(kk); if kk=100000 then begin maxErr:= abs(y1-fun1(x));
if abs(y2-fun2(x))>maxErr then maxErr:= abs(y2-fun2(x)); writeln(x,' ',maxErr);
kk:=0; end; end; tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln;
writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec); readln; end.
```

Метод Бутчера 6-го порядка решения задачи Коши для системы двух ОДУ:

```
program ButcherSystemof2ode; {$APPTYPE CONSOLE} uses SysUtils;
const c2=0.5; c3=2.0/3.0; c4=1.0/3.0; c5=5.0/6.0; c6=1.0/6.0; c7=1.0;
b1=13.0/200.0;  b2=0.0;  b3=11.0/40.0;  b4=11.0/40.0;  b5=4.0/25.0;  b6=4.0/25.0;
b7=13.0/200.0;  a21=0.5;  a31=2.0/9.0;  a32=4.0/9.0;  a41=7.0/36.0;  a42=2.0/9.0;
a43=-1.0/12.0;  a51=-35.0/144.0;  a52=-55.0/36.0;  a53=35.0/48.0;  a54=15.0/8.0;
a61=-1.0/360.0; a62=-11.0/36.0; a63=-1.0/8.0; a64=0.5; a65=0.1; a71=-41.0/260.0;
a72=22.0/13.0; a73=43.0/156.0; a74=-118.0/39.0; a75=32.0/195.0; a76=80.0/39.0;
var Q11, Q12, Q21, Q22, Q31, Q32, Q41, Q42, Q51, Q52, Q61, Q62, Q71, Q72, z21,
z22, z23:extended; z31, z32, z33, z41, z42, z43, z51, z52, z53, z61, z62, z63,
z71, z72, z73:extended; x,y1,y2,maxErr,h,Anach,Bkonech:extended; kk,k:int64;
 hour,minut,sec,msec:word; tnach,tkonech:extended;
function f1(x,y1,y2: extended):extended; begin f1:=x+2*y1/x-sqrt(y2) end;
function f2(x,y1,y2: extended):extended; begin f2:=2*sqrt(y2) end;
function fun1(x:extended):extended; begin fun1:=x+sqr(x) end;
function fun2(x:extended):extended; begin fun2:=sqr(x+1) end;
begin tnach:=Gettime; Anach:=1; Bkonech:=10;y1:=2; y2:=4; h:=1.03e-3;
kk:=0; x:=Anach; k:=1;
while x<Bkonech do begin Q11:=h*f1(x,y1,y2);
Q12:=h*f2(x,y1,y2); z21:=x+c2*h; z22:=y1+a21*Q11; z23:=y2+a21*Q12;
Q21:=h*f1(z21,z22,z23); Q22:= h*f2(z21,z22,z23); z31:=x+c3*h;
z32:=y1+a31*Q11+a32*Q21; z33:=y2+a31*Q12+a32*Q22; Q31:=h*f1(z31,z32,z33);
Q32:=h*f2(z31,z32,z33); z41:=x+c4*h; z42:=y1+a41*Q11+a42*Q21+a43*Q31;
z43:=y2+a41*Q12+a42*Q22+a43*Q32; Q41:=h*f1(z41,z42,z43); Q42:=h*f2(z41,z42,z43);
z51:=x+c5*h; z52:=y1+a51*Q11+a52*Q21+a53*Q31+a54*Q41;
z53:=y2+a51*Q12+a52*Q22+a53*Q32+a54*Q42; Q51:=h*f1(z51,z52,z53);
Q52:= h*f2(z51,z52,z53);  z61:=x+c6*h;
z62:=y1+a61*Q11+a62*Q21+a63*Q31+a64*Q41+a65*Q51; z63:=y2+a61*Q12+a62*Q22+a63*Q32
+a64*Q42+a65*Q52; Q61:=h*f1(z61,z62,z63); Q62:=h*f2(z61,z62,z63); z71:=x+c7*h;
```

```
z72:=y1+a71*Q11+a72*Q21+a73*Q31+a74*Q41+a75*Q51+a76*Q61;
z73:=y2+a71*Q12+a72*Q22+a73*Q32+a74*Q42+a75*Q52+a76*Q62;
Q71:=h*f1(z71,z72,z73); Q72:=h*f2(z71,z72,z73);
y1:= y1 + b1*Q11+b2*Q21+b3*Q31+b4*Q41+b5*Q51+b6*Q61+b7*Q71;
y2:= y2 + b1*Q12+b2*Q22+b3*Q32+b4*Q42+b5*Q52+b6*Q62+b7*Q72;
x:=Anach+k*h; inc(k); inc(kk); if kk=1000 then begin maxErr:= abs(y1-fun1(x));
if abs(y2-fun2(x))>maxErr then maxErr:= abs(y2-fun2(x)); writeln(x,' ',maxErr);
kk:=0; end; end; tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec);writeln;
writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec); readln; end.
```

Метод Дормана-Принса 8-го порядка решения задачи Коши для системы двух ОДУ:

```
program DormanPrinceSystemof2ode; {$APPTYPE CONSOLE} uses SysUtils;
const  c1=0;  c2=1.0/18.0;  c3=1.0/12.0;  c4=1.0/8.0;  c5=5.0/16.0;  c6=3.0/8.0;
c7=59.0/400.0;   c8=93.0/200.0;   c9=5490023248.0/9719169821.0;   c10=13.0/20.0;
c11=1201146811.0/1299019798.0;  c12=1.0;  c13=1.0;  a21=1.0/18.0;  a31=1.0/48.0;
a32=1.0/16.0;  a41=1.0/32.0;  a42=0;  a43=3.0/32.0;  a51=5.0/16.0;  a52=0;  a53=-
75.0/64.0;    a54=75.0/64.0;    a61=3.0/80.0;    a62=0;    a63=0;    a64=3.0/16.0;
a65=3.0/20.0; a71=29443841.0/614563906.0; a72=0; a73=0;
a74=77736538.0/692538347.0; a75=-28693883.0/1125000000.0;
 a76=23124283.0/   1800000000.0;   a81=16016141.0/946692911.0;   a82=0;   a83=0;
a84=61564180.0/158732637.0; a85=22789713.0/633445777.0;
a86=545815736.0/2771057229.0; a87=-180193667.0/1043307555.0;
a91=39632708.0/573591083.0; a92=0; a93=0; a94=-433636366.0/683701615.0;
a95=-421739975.0/2616292301.0; a96=100302831.0/723423059.0;
a97=790204164.0/839813087.0; a98=800635310.0/ 3783071287.0;
a101=246121993.0/1340847787.0; a102=0; a103=0;
a104=-37695042795.0 /15268766246.0; a105=-309121744.0/1061227803.0;
a106=-12992083.0/ 490766935.0; a107=6005943493.0/2108947869.0;
a108=393006217.0/1396673457.0; a109=123872331.0/1001029789.0;
a111=-1028468189.0/846180014.0; a112=0; a113=0; a114=8478235783.0/508512852.0;
a115=1311729495.0/1432422823.0; a116=-10304129995.0/1701304382.0;
a117=-48777925059.0/3047939560.0; a118=15336726248.0/1032824649.0;
a119=-45442868181.0/3398467696.0; a1110=3065993473.0/597172653.0;
a121=185892177.0/718116043.0; a122=0; a123=0; a124=-3185094517.0/667107341.0;
a125=-477755414.0/1098053517.0; a126=-703635378.0/230739211.0;
a127=5731566787.0/1027545527.0; a128=5232866602.0/850066563.0;
a129=-4093664535.0/808688257.0; a1210=3962137247.0/1805957418.0;
a1211=65686358.0/487910083.0;   a131=403863854.0/491063109.0;   a132=0;   a133=0;
a134=-5068492393.0/434740067.0; a135=-411421997.0/543043805.0;
 a136=652783627.0/914296604.0; a137=11173962825.0/925320556.0;
a138=-13158990841.0/6184727034.0; a139=3936647629.0/1978049680.0;
a1310=-160528059.0/685178525.0; a1311=248638103.0/1413531060.0; a1312=0;
b1=13451932.0/455176623.0; b2=0; b3=0; b4=0; b5=0; b6=-808719846.0/976000145.0;
b7=1757004468.0/5645159321.0; b8=656045339.0/265891186.0;
b9=-3867574721.0/1518517206.0; b10=465885868.0/322736535.0;
b11=53011238.0/667516719.0; b12=2.0/45.0; b13=0;
var k11,k21,k31,k41,k51,k61,k71,k81,k91,k101,k111,k121,k131:extended;
k12,k22,k32,k42,k52,k62,k72,k82,k92,k102,k112,k122,k132:extended;
y1_4, y1_5,y1_6,y1_7,y1_8,y1_9,y1_10,y1_11,y1_12,y1_13: extended;
y2_4, y2_5,y2_6,y2_7,y2_8,y2_9,y2_10,y2_11,y2_12,y2_13:extended;
x,y1,y2,maxErr,h,Anach,Bkonech:extended; kk,k:int64;
hour,minut,sec,msec:word; tnach,tkonech:extended;
function f1(x,y1,y2: extended):extended;begin f1:=x+2*y1/x-sqrt(y2) end;
function f2(x,y1,y2: extended):extended;begin f2:=2*sqrt(y2) end;
function fun1(x:extended):extended;begin fun1:=x+sqr(x) end;
function fun2(x:extended):extended;begin fun2:=sqr(x+1) end;
begin tnach:=Gettime; Anach:=1; Bkonech:=10;y1:=2; y2:=4; h:=1.03e-2;
kk:=0; x:=Anach; k:=1;
```

```
while x < Bkonech do begin k11:= h*f1(x,y1,y2);
k12:= h*f2(x,y1,y2); k21:= h*f1(x+c2 *h, y1+ a21*k11, y2 +a21*k12);
k22:= h*f2(x+c2 *h, y1+ a21*k11, y2 +a21*k12);
k31:= h*f1(x+c3 *h, y1+ a31*k11+a32*k21, y2+ a31*k12+a32*k22);
k32:= h*f2(x+c3 *h, y1+ a31*k11+a32*k21, y2+ a31*k12+a32*k22);
y1_4:=y1+a41*k11+a42*k21+a43*k31; y2_4:=y2+a41*k12+a42*k22+a43*k32;
k41:= h*f1(x+c4 *h, y1_4, y2_4); k42:= h*f2(x+c4 *h, y1_4, y2_4);
y1_5:=y1+a51*k11+a52*k21+a53*k31+a54*k41;
y2_5:=y2+a51*k12+a52*k22+a53*k32+a54*k42;
k51 := h*f1(x+c5 *h, y1_5, y2_5); k52 := h*f2(x+c5 *h, y1_5, y2_5);
y1_6:= y1 + a61 *k11 +a62 *k21 +a63 *k31 +a64 *k41 +a65 *k51;
y2_6:= y2 + a61 *k12 +a62 *k22 +a63 *k32 +a64 *k42 +a65 *k52;
k61:= h*f1(x+c6 *h, y1_6, y2_6); k62:= h*f2(x+c6 *h, y1_6, y2_6);
y1_7:= y1 + a71 *k11 +a72 *k21 +a73 *k31 +a74 *k41 +a75 *k51 +a76 *k61;
y2_7:= y2 + a71 *k12 +a72 *k22 +a73 *k32 +a74 *k42 +a75 *k52 +a76 *k62;
k71  := h*f1(x+c7 *h, y1_7, y2_7); k72  := h*f2(x+c7 *h, y1_7, y2_7);
y1_8:=y1+a81*k11 +a82 *k21 +a83 *k31 +a84 *k41 +a85 *k51 +a86 *k61 +a87 *k71;
y2_8:=y2+a81 *k12 +a82 *k22 +a83 *k32 +a84 *k42 +a85 *k52 +a86 *k62 +a87 *k72;
k81:= h*f1(x+c8 *h, y1_8, y2_8); k82  := h*f2(x+c8 *h, y1_8, y2_8);
y1_9:=y1+a91*k11+a92*k21+a93*k31+a94*k41+a95*k51+a96*k61+a97*k71+a98*k81;
y2_9:=y2+a91*k12+a92*k22+a93*k32+a94*k42+a95*k52+a96*k62+a97*k72+a98*k82;
k91 := h*f1(x+c9 *h, y1_9, y2_9); k92 := h*f2(x+c9 *h, y1_9, y2_9);
y1_10:=y1+a101*k11+a102*k21+a103*k31+a104*k41+a105*k51+a106*k61+a107*k71
+a108*k81  +a109*k91;  y2_10:=  y2  +  a101*k12  +a102*k22  +a103*k32  +a104*k42
+a105*k52 +a106*k62 +a107*k72 +a108*k82 +a109*k92;
k101 := h*f1(x+c10*h, y1_10, y2_10);k102 := h*f2(x+c10*h, y1_10, y2_10);
y1_11:=  y1  +  a111*k11  +a112*k21  +a113*k31  +a114*k41  +a115*k51  +a116*k61
+a117*k71  +a118*k81  +a119*k91  +a1110*k101;  y2_11:=  y2  +  a111*k12  +a112*k22
+a113*k32   +a114*k42   +a115*k52   +a116*k62   +a117*k72   +a118*k82   +a119*k92
+a1110*k102; k111:=h*f1(x+c11*h,y1_11,y2_11);k112:=h*f2(x+c11*h,y1_11,y2_11);
y1_12:=  y1  +  a121*k11  +a122*k21  +a123*k31  +a124*k41  +a125*k51  +a126*k61
+a127*k71  +a128*k81  +a129*k91  +a1210*k101  +a1211*k111;  y2_12:=  y2  +  a121*k12
+a122*k22 +a123*k32 +a124*k42 +a125*k52 +a126*k62 +a127*k72 +a128*k82 +a129*k92
+a1210*k102 +a1211*k112; k121 := h*f1(x+c12*h, y1_12, y2_12);
k122 := h*f2(x+c12*h, y1_12, y2_12); y1_13:= y1 + a131*k11 +a132*k21 +a133*k31
+a134*k41   +a135*k51   +a136*k61   +a137*k71   +a138*k81   +a139*k91   +a1310*k101
+a1311*k111 +a1312*k121; y2_13:= y2 + a131*k12 +a132*k22 +a133*k32 +a134*k42
+a135*k52   +a136*k62   +a137*k72   +a138*k82   +a139*k92   +a1310*k102   +a1311*k112
+a1312*k122; k131:=h*f1(x+c13*h,y1_13,y2_13); k132:=h*f2(x+c13*h,y1_13,y2_13);
y1:=y1+b1*k11 +b2*k21 +b3*k31+ b4*k41+ b5*k51+ b6*k61+ b7*k71+ b8*k81+ b9*k91+
b10*k101+ b11*k111+ b12*k121+ b13*k131; y2:=y2+ b1*k12+ b2*k22+ b3*k32+ b4*k42+
b5*k52+ b6*k62+ b7*k72+ b8*k82+ b9*k92+b10*k102+b11*k112+b12*k122+b13*k132;
x:=Anach+k*h;inc(k);inc(kk);
if kk=100 then begin maxErr:= abs(y1-fun1(x));
if abs(y2-fun2(x))>maxErr then maxErr:= abs(y2-fun2(x)); writeln(x,' ',maxErr);
kk:=0; end; end; tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln;
writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec); readln; end.
```

Данные из табл. 2.2, 2.5, 2.8, 2.11 – 2.13, 3.1 – 3.4, 5.1, 5.2, 5.5 – 5.9 подтверждают повышение точности приближения при применении разностных методов высоких порядков: меньшие погрешности приближения получены по методам 6-го и 8-го порядков аппроксимации, однако их применение требует соответственной степени гладкости функций правой части системы. В том числе и поэтому в экспериментах с жесткими задачами для сравнения точности

и быстродействия предлагаемых в работе методов использованы также результаты, полученные на основе специальных решателей для жестких задач, реализованных в системах компьютерной математики и научных библиотеках (табл. 2.11 – 2.13, 3.4, 5.1, 5.2). Программы варьируемого разностно-полиномиального и кусочно-интерполяционного методов во всех приведенных в диссертации примерах дают меньшие границы погрешности в сравнении с результатами, полученными по представленным программным реализациям явных методов Рунге-Кутты и известным реализациям специализированных решателей для систем ОДУ.

ПРИЛОЖЕНИЕ К ГЛАВЕ 3

П3.1. Приложение к п. 3.6. С целью снижения трудоемкости, в частности, в случае приближенного решения жестких задач, в главах 3, 5 исследовано применение кусочно-интерполяционного метода с фиксированными значениями параметров n, k, ℓ, где n – степень интерполяционных полиномов, k – параметр, определяющий число подынтервалов $P = 2^k$ на текущем интервале приближения, ℓ – число итераций на каждом подынтервале. В главе 3 даны соответствующие оценки трудоемкости метода. Непосредственно ниже с подробными комментариями приводится программа, реализующая описанный метод для случая системы двух ОДУ и результаты численного эксперимента на примере приближенного решения жесткой задачи Коши $y_1' = -400(y_2 - 1) - 2y_1 x$, $y_2' = 5(1 - 0.003y_1)$, $y_1(0) = 0$, $y_2(0) = 1$ на отрезке $x \in [0, 128]$.

```
Program Fixed_Piecewise_Interpolation_S2; {$APPTYPE CONSOLE} uses SysUtils,
Math; var kiter,koutput,k_:integer;Npol:byte;tt:longint; tnach,tkonech:extended;
Anach,Bkonech,velint,ynach1,ynach2:extended; our,minut,sec,msec:word;fc:longint;
function f1(x,y1,y2:extended):extended;begin f1:=-400*(y2-1)-2*y1*x;inc(fc) end;
function f2(x,y1,y2:extended):extended;begin f2:=5*(1-0.003*y1) end;
function fun1(x:extended):extended; begin fun1:=-1000*x*x; end;
function fun2(x:extended):extended; begin fun2:=5*x*(1+x*x)+1 end;
procedure RD(y1_nach, y2_nach, A_nach, B_konech, vel_int:extended; n:byte;
k, k_iter, k_output:integer; tt_:longint); const nn=4;
type matr=array[0..nn,0..nn] of extended; vect=array[0..nn] of extended;
matrC=array[-5..nn+1] of extended; matrAll=array[0..66000] of matrC;
matrC_=^matrAll; var d:matr; CC1,CC2:matrC; xx:vect;
a0,b0,h,x,y01,y02,PogrFun1,PogrFun2,PogrFun:extended; tnach,tkonech:extended;
i,pod1,pod2:integer; kk,m:longint; hour,minut,sec,msec:word; Ck1,Ck2:matrC_;
//Восстановление коэффициентов полинома по его корням
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr;
begin e[1,1]:=1; e[1,0]:=0; for k:=2 to n do begin e[k,0]:=-e[k-1,0]*(k-1);
for i:=1 to k-1 do e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1]
end; for k:=1 to n do for i:=0 to k do d[i,k]:=e[k,i] end;
//вычисление конечных разностей
procedure Konech_Raznoct(fy1,fy2:vect; n:byte; var dy1,dy2:matr); var i,j:byte;
begin for j:=0 to n-1 do begin dy1[1,j]:=fy1[j+1]-fy1[j]; dy2[1,j]:=fy2[j+1]-
fy2[j]; end; for i:=2 to n do for j:=0 to n-i do begin dy1[i,j]:=dy1[i-1,j+1]-
dy1[i-1,j]; dy2[i,j]:=dy2[i-1,j+1]-dy2[i-1,j]; end; end;
//построение полинома Ньютона в алгебраической форме по узловым значениям
procedure Newton(U1,U2:Vect; n:byte; var Mcoef1,Mcoef2:matrC);
var dy1,dy2:matr; b1,b2:vect; p,s1,s2:extended; j,i:byte;
begin Konech_Raznoct(U1,U2,n,dy1,dy2); p:=1;
for j:=1 to n do begin p:=p*j; b1[j]:=dy1[j,0]/p; b2[j]:=dy2[j,0]/p; end;
Mcoef1[0]:=U1[0]; Mcoef2[0]:=U2[0];
```

```
for i:=1 to n do begin s1:=0; s2:=0; for j:=i to n do begin s1:=s1+d[i,j]*b1[j];
s2:=s2+d[i,j]*b2[j]; end; Mcoef1[i]:=s1; Mcoef2[i]:=s2; end end;
//вычисление значения полинома по схеме Горнера
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n]; for i:=n-1
downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
//Построение приближения на текущем отрезке
procedure Subinterval(k_,n,K_it:integer; a0,b0,Ynach1,Ynach2:extended; var
Ck1_,Ck2_:matrC_); var hpd,a00,b00,y01,y02,h:extended; m,pod:longint; x:vect;
j:byte; i,iter,r:integer; fy1,fy2,y1,y2,ytemp1,ytemp2:vect; C1,C2:matrC;
A1,A2:matrC; t,pp1,pp2,sum1,sum2,sum:extended;
begin hpd:=(b0-a0)/exp(k_*ln(2)); a00:=a0;b00:=a00+hpd; y01:=Ynach1;y02:=Ynach2;
x[0]:=a0;m:=0; pod:=0;
while a00<=b0-hpd/2 do begin h:=(b00-a00)/n; for j:=1 to n do begin inc(m);
x[j]:=a0+m*h end; for i:=0 to n do begin y1[i]:=y01; y2[i]:=y02; end;
fy1[0]:=f1(x[0],y01,y02); fy2[0]:=f2(x[0],y01,y02);
for iter:=1 to K_it do begin for i:=1 to n do begin ytemp1[i]:=y1[i];
ytemp2[i]:=y2[i]; end;
for i:=1 to n do begin fy1[i]:=f1(x[i],y1[i],y2[i]);
fy2[i]:=f2(x[i],y1[i],y2[i]); end; Newton(fy1,fy2,n,A1,A2);
for i:=1 to n do begin t:=(x[i]-x[0])/h; pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1);
for r:=n-1 downto 0 do begin pp1:=pp1*t+A1[r]/(r+1); pp2:=pp2*t+A2[r]/(r+1);
end; y1[i]:=pp1*h*t+y1[0]; y2[i]:=pp2*h*t+y2[0]; end; sum1:=0; for i:=1 to n do
sum1:=sum1+abs(y1[i]-ytemp1[i]); sum2:=0; for i:=1 to n do sum2:=sum2+abs(y2[i]-
ytemp2[i]); if sum1>sum2 then sum:=sum1 else sum:=sum2;
if (sum<1e-30) then break; end; C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-
3]:=n+1; C1[-4]:=k;C1[-5]:=n*h; C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h; C2[-
3]:=n+1; C2[-4]:=k;C2[-5]:=n*h;
for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i; C2[i]:=A2[i-1]*h/i; end;
Ck1_^[pod]:=C1; Ck2_^[pod]:=C2; y01:=y1[n]; y02:=y2[n]; x[0]:=x[n]; inc(pod);
a00:=a00+hpd; b00:=a00+hpd end; end;
begin Viet(nn,d); New(Ck1);New(Ck2); tnach:=Gettime; kk:=0; a0:=A_nach;
b0:=a0+vel_int; y01:=y1_nach; y02:=y2_nach;
while a0 <= B_konech-vel_int/2 do begin
Subinterval(k,n,k_iter,a0,b0,y01,y02,Ck1,Ck2); kk:=kk+1;if kk=tt_ then
begin //Вывод погрешности приближения в проверочных точках
for i:=1 to k_output-1 do begin x:= a0+i*Vel_int/k_output;
pod1:=trunc((x-a0)/Ck1[0,-5]);      PogrFun1:=abs(fun1(x)-Gorner(Ck1^[pod1],x));
PogrFun2:=abs(fun2(x)-Gorner(Ck2^[pod1],x));
writeln(x:6:2,'  ',Gorner(Ck1^[pod1],x),'  ',PogrFun1,'  ',Gorner(Ck2^[pod1],x),'
',PogrFun2); end; kk:=0; end; pod1:=trunc(exp(k*ln(2)))-1;
y01:=Gorner(Ck1[pod1],b0); y02:=Gorner(Ck2[pod1],b0); writeln(b0:6:2,'  ',y01,'
',abs(y01-fun1(b0)),' ',y02,' ',abs(y02-fun2(b0))); a0:=a0+vel_int;
b0:=a0+vel_int; end; Dispose(Ck1); Dispose(Ck2); tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln; writeln('Calculation
time  ',Hour,':',Minut,':', Sec,':',MSec); writeln('fc= ',fc); end;
BeginAnach:=0; Bkonech:=128;{границы отрезка интегрирования}
ynach1:=0;ynach2:=1; {начальное условие}
velint:=128; {величина интервала приближения}
Npol:=2; {степень интерполяционных полиномов Ньютона}
k_:=15; {параметр, определяющий число подынтервалов P=2^k_}
kiter:=28; {число итераций}
tt:=1; {частота вывода интервалов с результатами приближения}
koutput:=100;{число точек на интервале для вывода результатов приближения}
RD(ynach1,ynach2,Anach,Bkonech,velint,Npol,k_,kiter,koutput,tt); readln; end.
```

Результат работы программы (в 1-й колонке – аргумент, во 2-й – приближенное значение y_1, в 3-й – абсолютная погрешность приближения y_1, в 4-й – приближенное значение y_2, в 5-й – абсолютная погрешность приближения y_2):

```
1.28   -1.63840000000E+0003  0.00000000000E+0000  1.78857600000E+0001  0.00000000000E+0000
2.56   -6.55360000000E+0003  0.00000000000E+0000  9.76860800000E+0001  6.93889390390E-0018
3.84   -1.47456000000E+0004  0.00000000000E+0000  3.03315520000E+0002  2.77555756156E-0017
5.12   -2.62144000000E+0004  0.00000000000E+0000  6.97688640000E+0002  5.55111512313E-0017
6.40   -4.09600000000E+0004  0.00000000000E+0000  1.34372000000E+0003  0.00000000000E+0000
7.68   -5.89824000000E+0004  0.00000000000E+0000  2.30432416000E+0003  0.00000000000E+0000
8.96   -8.02816000000E+0004  0.00000000000E+0000  3.64241568000E+0003  0.00000000000E+0000
10.24  -1.04857600000E+0005  0.00000000000E+0000  5.42090912000E+0003  4.44089209850E-0016
11.52  -1.32710400000E+0005  0.00000000000E+0000  7.70271904000E+0003  4.44089209850E-0016
12.80  -1.63840000000E+0005  0.00000000000E+0000  1.05507600000E+0004  0.00000000000E+0000
.....................................................................................
60.16  -3.61922560000E+0006  0.00000000000E+0000  1.08896486048E+0006  0.00000000000E+0000
61.44  -3.77487360000E+0006  0.00000000000E+0000  1.15994936992E+0006  1.13686837722E-0013
62.72  -3.93379840000E+0006  0.00000000000E+0000  1.23395377824E+0006  0.00000000000E+0000
64.00  -4.09600000000E+0006  0.00000000000E+0000  1.31104100000E+0006  0.00000000000E+0000
65.28  -4.26147840000E+0006  4.54747350886E-0013  1.39127394976E+0006  1.13686837722E-0013
66.56  -4.43023360000E+0006  0.00000000000E+0000  1.47471554208E+0006  0.00000000000E+0000
.....................................................................................
116.48 -1.35675904000E+0007  0.00000000000E+0000  7.90234804896E+0006  0.00000000000E+0000
117.76 -1.38674176000E+0007  0.00000000000E+0000  8.16572528288E+0006  4.54747350886E-0013
119.04 -1.41705216000E+0007  0.00000000000E+0000  8.43489065632E+0006  0.00000000000E+0000
120.32 -1.44769024000E+0007  0.00000000000E+0000  8.70990708384E+0006  9.09494701773E-0013
121.60 -1.47865600000E+0007  0.00000000000E+0000  8.99083748000E+0006  0.00000000000E+0000
122.88 -1.50994944000E+0007  0.00000000000E+0000  9.27774475936E+0006  0.00000000000E+0000
124.16 -1.54157056000E+0007  9.09494701773E-0013  9.57069183648E+0006  0.00000000000E+0000
125.44 -1.57351936000E+0007  0.00000000000E+0000  9.86974162592E+0006  0.00000000000E+0000
126.72 -1.60579584000E+0007  0.00000000000E+0000  1.01749570422E+0007  9.09494701773E-0013
128.00 -1.63840000000E+0007  0.00000000000E+0000  1.04864010000E+0007  0.00000000000E+0000

fc= 1313078
```

Граница абсолютной погрешности кусочно-интерполяционного приближения не превышает порядка 10^{-13} на всем отрезке интегрирования (параметры метода $n=3$, $k=15$, $\ell=28$, величина интервала: `velint=128`). Погрешность приближения решения этой же задачи по известным разностным методам характеризуется границей порядка 10^{-10}. Сравнение нормы абсолютной погрешности кусочно-интерполяционного приближения с результатами приближения по разностным методам представлено в табл. 3.4, сравнение трудоемкости и времени работы программ в табл. 3.5. Предложенный метод характеризуется меньшей трудоемкостью при решении данной задачи чем разностные методы высокого порядка.

Кусочно-интерполяционный метод с итерационным уточнением в представленной выше программной реализации характеризуется малой трудоемкостью и в случае решения нежестких задач, в том числе и с неустойчивыми по Ляпунову решениями. Для приближенного решения задачи: $y_1' = x + 2y_1 / x - \sqrt{y_2}$, $y_2' = 2\sqrt{y_2}$, $y_1(1) = 2$, $y_2(1) = 4$, на отрезке $[1, 513]$ имеет место следующий результат работы программы:

```
6.12    4.35744000000E+0001  0.00000000000E+0000  5.06944000000E+0001 0.00000000000E+0000
11.24   1.37577600000E+0002  0.00000000000E+0000  1.49817600000E+0002 0.00000000000E+0000
16.36   2.84009600000E+0002  0.00000000000E+0000  3.01369600000E+0002 0.00000000000E+0000
21.48   4.82870400000E+0002  0.00000000000E+0000  5.05350400000E+0002 0.00000000000E+0000
26.60   7.34160000000E+0002  0.00000000000E+0000  7.61760000000E+0002 0.00000000000E+0000
31.72   1.03787840000E+0003  1.11022302463E-0016  1.07059840000E+0003 1.11022302463E-0016
36.84   1.39402560000E+0003  0.00000000000E+0000  1.43186560000E+0003 0.00000000000E+0000
41.96   1.80260160000E+0003  0.00000000000E+0000  1.84556160000E+0003 0.00000000000E+0000
47.08   2.26360640000E+0003  0.00000000000E+0000  2.31168640000E+0003 0.00000000000E+0000
52.20   2.77704000000E+0003  0.00000000000E+0000  2.83024000000E+0003 0.00000000000E+0000
.....................................................................................
241.64  5.86315296000E+0004  0.00000000000E+0000  5.88741696000E+0004 0.00000000000E+0000
246.76  6.11372576000E+0004  0.00000000000E+0000  6.13850176000E+0004 0.00000000000E+0000
251.88  6.36954144000E+0004  0.00000000000E+0000  6.39482944000E+0004 0.00000000000E+0000
257.00  6.63060000000E+0004  0.00000000000E+0000  6.65640000000E+0004 0.00000000000E+0000
262.12  6.89690144000E+0004  0.00000000000E+0000  6.92321344000E+0004 0.00000000000E+0000
267.24  7.16844576000E+0004  0.00000000000E+0000  7.19526976000E+0004 0.00000000000E+0000
.....................................................................................
466.92  2.18481206400E+0005  0.00000000000E+0000  2.18949126400E+0005 0.00000000000E+0000
472.04  2.23293801600E+0005  0.00000000000E+0000  2.23766841600E+0005 0.00000000000E+0000
477.16  2.28158825600E+0005  0.00000000000E+0000  2.28636985600E+0005 0.00000000000E+0000
482.28  2.33076278400E+0005  0.00000000000E+0000  2.33559558400E+0005 0.00000000000E+0000
487.40  2.38046160000E+0005  0.00000000000E+0000  2.38534560000E+0005 0.00000000000E+0000
492.52  2.43068470400E+0005  0.00000000000E+0000  2.43561990400E+0005 0.00000000000E+0000
497.64  2.48143209600E+0005  0.00000000000E+0000  2.48641849600E+0005 0.00000000000E+0000
502.76  2.53270377600E+0005  0.00000000000E+0000  2.53774137600E+0005 0.00000000000E+0000
507.88  2.58449974400E+0005  0.00000000000E+0000  2.58958854400E+0005 0.00000000000E+0000
513.00  2.63682000000E+0005  0.00000000000E+0000  2.64196000000E+0005 0.00000000000E+0000

fc=44066
```

Посторенное кусочно-интерполяционное приближение с параметрами: `velint=512`, $n=3$, $k=11$, $\ell=20$, характеризуется «нулевыми» значениями абсолютной погрешности в формате данных *extended*, тогда как граница погрешности приближения разностными методами, включая методы высокого порядка, на том же отрезке интегрирования характеризуется порядком 10^{-13}. Канонические нормы вектора абсолютных погрешностей компонентов системы представлены в главе 5 (табл. 5.6) наряду с количеством обращений к функции правой части. Использование предложенного метода, как и для предыдущей задачи, позволило повысить точность приближения при меньшей трудоемкости. Так, метод Дормана-Принса 8-го порядка при решении рассматриваемой задачи характеризуется следующими значениями: граница абсолютной погрешности 10^{-13}, количество обращений к функции правой части `fc=6656000`, время работы программы `882 ms`. Характеристики приближенного решения на основе предложенного метода: граница абсолютной погрешности 10^{-16}, количество обращений к функции правой части `fc=44066`, время работы программы `16 ms`. Таким образом, относительно высокая точность кусочно-

интерполяционного приближения решения, неустойчивого к возмущениям начальных условий, сохраняется и при исключении автоматического выбора параметров (см. табл. 3.3), что значительно снижает трудоемкость метода.

Как было отмечено в главе 3 отличительным качеством предложенного метода является его применимость к частному случаю двухточечной задачи Коши, где сохраняется высокая точность приближения и его свойство непрерывности и дифференцируемости.

П3.2. Приложение к п. 3.8. Кусочно-интерполяционный метод для приближенного решения двухточечной задачи Коши для системы ОДУ вида $Y' = F(x,Y)$, $Y(x_0)=Y_0$, $Y(x_{fin})=Y_{fin}$, где $Y=(y_1,y_2,\ldots,y_N)$, $Y_0=(y_{01},y_{02},\ldots,y_{0N})$, $Y_{fin}=(y_{fin\,1},y_{fin\,2},\ldots,y_{fin\,N})$, на отрезке $[x_0, x_{fin}]$ реализует программа, листинг которой представлен непосредственно ниже. Результат работы программы дан на примере решения задачи $y_1' = x+2y_1/x-\sqrt{y_2}$, $y_2' = 2\sqrt{y_2}$, $y_1(1)=2$, $y_2(1)=4$, $y_1(512)=262656$, $y_2(512)=263169$. Значения решения в конечной точке x_{fin} взяты из соответственных аналитических решений одноточечной задачи Коши.

```
program VPI_S2_twin; {$APPTYPE CONSOLE} uses SysUtils,  Math;
const nn=30; MaxExtended:extended = 1.18E4932; type
vect=array[0..nn] of extended; var Anach,Bkonech,h_s: extended;
 K_iter,K_iter_itog,output,ii,n_s:integer;  Nmin,Nmax,Kmin,Kmax:byte;
 Vel_int,Ynach1,Ynach2,otrezok_integr:extended;
Method_Iter,Nev:byte; sk:boolean; y1_s,y2_s,x_s:vect;
function f1(x,y1,y2: extended):extended; begin f1:=x+(2*y1)/x-sqrt(y2) end;
function f2(x,y1,y2: extended):extended; begin f2:=2*sqrt(y2) end;
function fun1(x:extended):extended; begin fun1:=x*(1+x) end;
function fun2(x:extended):extended; begin fun2:=(x+1)*(x+1) end;
procedure RD1(Ynach1, Ynach2, Vel_int:extended; K_iter, Nmin, Nmax, Kmin, Kmax,
output:integer;otrezok_integr:extended); type
 matr=array[0..nn,0..nn] of extended; matrC=array[-5..nn+1] of extended;
 matrAll=array[0..33000] of matrC; matrC_=^matrAll;
var  pod,pod1,pod2:longint; n,k,n_,k_,n_opt,k_opt:byte; i,j,int:integer;
 x,y01,y02,a0,b0,a00,b00,max,min,MinGL,s:extended;   Min0,   Xfinal,   Xfinal1,
Xfinal2, h,s1,s2:extended; Ck1,Ck11,Ck12,Ck1_,Ck1_opt,Ck11_,Ck11_opt:matrC_;
Ck2,Ck21,Ck22,Ck2_,Ck2_opt,Ck21_,Ck21_opt:matrC_; d:matr;
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr; begin
e[1,1]:=1; e[1,0]:=0; for k:=2 to n do begin e[k,0]:=-e[k-1,0]*(k-1); for i:=1
to k-1 do e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1] end;
for k:=1 to n do for i:=0 to k do d[i,k]:=e[k,i] end;
procedure Konech_Raznoct(fy:vect; n:byte; var dy:matr); var i,j:byte;
```

```
begin for j:=0 to n-1 do dy[1,j]:=fy[j+1]-fy[j]; for i:=2 to n do for j:=0 to n-
i do dy[i,j]:=dy[i-1,j+1]-dy[i-1,j] end;
procedure Newton(U:Vect; n:byte; var Mcoef:matrC); var dy:matr; b:vect;
p,s:extended; j,i:byte;
begin Konech_Raznoct(U,n,dy); p:=1; for j:=1 to n do begin p:=p*j;
b[j]:=dy[j,0]/p; end; Mcoef[0]:=U[0]; for i:=1 to n do begin s:=0; for j:=i to n
do s:=s+d[i,j]*b[j]; Mcoef[i]:=s; end end;
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
procedure Polynomial(x:vect; h,y01,y02:extended; n,k,K_iter:integer; var
C1,C2:matrC); var i,iter:integer; sum1,sum2,sum:extended; ytemp1, ytemp2, fy1,
y1, fy2, y2:vect; A1,A2:matrC;
begin for i:=0 to n do y1[i]:=y01; for i:=0 to n do y2[i]:=y02; for iter:=1 to
K_iter do begin for i:=1 to n do begin ytemp1[i]:=y1[i]; ytemp2[i]:=y2[i]; end;
if Method_Iter=1 then begin for i:=0 to n do begin fy1[i]:=f1(x[i],y1[i],y2[i]);
fy2[i]:=f2(x[i],y1[i],y2[i]) end; Newton(fy1,n,A1); Newton(fy2,n,A2);
C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;
C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h; C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h;
for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i; C2[i]:=A2[i-1]*h/i; end;
for i:=1 to n do begin y1[i]:=Gorner(C1,x[i]); y2[i]:=Gorner(C2,x[i]); end; end
else begin for i:=0 to n do fy1[i]:=f1(x[i],y1[i],y2[i]); Newton(fy1,n,A1);
C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;
for i:=1 to n+1 do C1[i]:=A1[i-1]*h/i; for i:=1 to n do y1[i]:=Gorner(C1,x[i]);
for i:=0 to n do fy2[i]:=f2(x[i],y1[i],y2[i]); Newton(fy2,n,A2); C2[0]:=y2[0];
C2[-1]:=x[0]; C2[-2]:=h; C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h; for i:=1 to n+1 do
C2[i]:=A2[i-1]*h/i; for i:=1 to n do y2[i]:=Gorner(C2,x[i]); end; sum1:=0; for
i:=1 to n do sum1:=sum1+abs(y1[i]-ytemp1[i]); sum2:=0; for i:=1 to n do
sum2:=sum2+abs(y2[i]-ytemp2[i]); if sum1>sum2 then sum:=sum1 else sum:=sum2;
if (sum<1e-30) or (sum>=1e50) then break; end end;
procedure Subinterval(k,n,K_iter:integer; a0,b0,Ynach1,Ynach2:extended; var Ck1,
Ck2: matrC_); var hpd,a00,b00,y01,y02,h:extended; m,pod:longint; x:vect; j:byte;
begin hpd:=(b0-a0)/exp(k*ln(2)); a00:=a0; b00:=a00+hpd; y01:=Ynach1;y02:=Ynach2;
x[0]:=a0; m:=0; pod:=0; while a00<=b0-hpd/2 do begin h:=(b00-a00)/n; for j:=1 to
n do begin inc(m); x[j]:=a0+m*h end;
Polynomial(x,h,y01,y02,n,k,K_iter,Ck1^[pod],Ck2^[pod]);
y01:=Gorner(Ck1^[pod],x[n]); y02:=Gorner(Ck2^[pod],x[n]); x[0]:=x[n]; inc(pod);
 a00:=a00+hpd; b00:=a00+hpd end end;
function Integral(a0,b0,a00,b00:extended; Ck:matrC_):extended;
var pod1,pod2,m:longint; x1,x2,h,ss:extended; begin
h:=(b00-a00)/5; m:=0;x1:=a00;x2:=a00+h;ss:=0; repeat pod1:=trunc((x1-a0)/Ck^[0,-
5]); if abs(x2-b0)<1e-15 then pod2:=trunc(exp(Ck^[0,-4]*ln(2)))-1 else pod2:=
trunc((x2-a0)/Ck^[0,-5]); ss:=ss+abs(Gorner(Ck^[pod2],x2)-Gorner(Ck^[pod1],x1));
inc(m); x1:=a00+m*h;x2:=x1+h; until x2>b00+h/2; Integral:=ss; end;
begin Viet(nn,d); New(Ck1); New(Ck11); New(Ck12); New(Ck1_); New(Ck1_opt);
New(Ck11_); New(Ck11_opt); New(Ck2); New(Ck21); New(Ck22); New(Ck2_);
New(Ck2_opt); New(Ck21_); New(Ck21_opt); a0:=Anach; b0:=Anach+Vel_int; while a0
< Bkonech do begin inc(int); MinGL:=MaxExtended; n:=Nmin; repeat k:=Kmin;
Min:=MaxExtended; repeat Subinterval(k,n,K_iter,a0,b0,Ynach1,Ynach2,Ck11,Ck21);
Subinterval(k+1,n,K_iter,a0,b0,Ynach1,Ynach2,Ck12,Ck22); max:=0; a00:=a0;
b00:=a00+otrezok_integr; i:=0; while b00<=b0 do begin if Nev=1 then begin
s1:=abs(Integral(a0,b0,a00,b00,Ck12)-Integral(a0,b0,a00,b00,Ck11));
s2:=abs(Integral(a0,b0,a00,b00,Ck22)-Integral(a0,b0,a00,b00,Ck21)); end else
begin s1:=abs(Integral(a0,b0,a00,b00,Ck12)-Integral(a0,b0,a00,b00,Ck11))/
(1e-7+abs(Integral(a0,b0,a00,b00,Ck12))); s2:=abs(Integral(a0,b0,a00,b00,Ck22)-
Integral(a0,b0,a00,b00,Ck21)) / (1e-7+abs(Integral(a0,b0,a00,b00,Ck22))); end;
if s1>s2 then s:=s1 else s:=s2; if s>max then max:=s; inc(i); a00:= a0+
i*otrezok_integr; b00:=a00+otrezok_integr; end; if max<Min then begin Min:=max;
Ck1_^:=Ck12^;Ck11_^:=Ck11^;Ck2_^:=Ck22^; Ck21_^:=Ck21^ end; inc(k);
 until k>Kmax-1; if Min<MinGL then begin MinGL:=Min; Ck1_opt^:=Ck1_^;
Ck11_opt^:=Ck11_^; Ck2_opt^:=Ck2_^;Ck21_opt^:=Ck21_^ end; inc(n); until n>Nmax;
```

```
Subinterval(trunc(Ck1_opt^[0,-4]),trunc(Ck1_opt^[0,-3]-1),K_iter_itog,a0,b0,
Ynach1, Ynach2,Ck1_opt,Ck2_opt); for i:=0 to output-1 do begin x:= a0+
i*Vel_int/ output; pod:=trunc((x-a0)/Ck1_opt^[0,-5]); if x<Bkonech then
// writeln(x:8:3,'     ',Gorner(Ck1_opt^[pod],x),'  ',Gorner(Ck2_opt^[pod],x));
writeln(x:8:3,'    ',abs(Gorner(Ck1_opt^[pod],x)-fun1(x)),'
',abs(Gorner(Ck2_opt^[pod],x)-fun2(x)));  end; n_s:=trunc(Ck1_opt^[0,-3])-1;
h_s:=Ck1_opt^[0,-5]/n_s;  for ii:=0 to n_s do x_s[ii]:=Bkonech+h_s*(ii-
trunc(n_s/2)); for i:=0 to trunc(n_s/2) do if x_s[i]>a0 then begin
if abs(x_s[i]-b0)<1e-15 then pod:=trunc(exp(Ck1_opt^[0,-4]*ln(2)))-1 else
pod:=trunc((x_s[i]-a0)/Ck1_opt^[0,-5]); y1_s[i]:=Gorner(Ck1_opt^[pod],x_s[i]);
y2_s[i]:=Gorner(Ck2_opt^[pod],x_s[i]); end;
pod:=trunc(exp(trunc(Ck1_opt^[0,-4])*ln(2)))-1;Ynach1:=Gorner(Ck1_opt^[pod],b0);
Ynach2:=Gorner(Ck2_opt^[pod],b0);       a0:=b0;       b0:=a0+Vel_int;       end;
Dispose(Ck1);Dispose(Ck11);Dispose(Ck12);Dispose(Ck1_);Dispose(Ck1_opt);
Dispose(Ck11_);Dispose(Ck11_opt);  Dispose(Ck2);  Dispose(Ck21);  Dispose(Ck22);
Dispose(Ck2_); Dispose(Ck2_opt);Dispose(Ck21_);Dispose(Ck21_opt); end;
procedure RD2(Ynach1,Ynach2,Vel_int:extended; K_iter, Nmin, Nmax, Kmin, Kmax,
output:integer;  otrezok_integr:extended);  type  matr=array[0..nn,0..nn]  of
extended; matrC=array[-5..nn+1] of extended; matrAll=array[0..33000] of matrC;
 matrC_=^matrAll; var pod,pod1,pod2:longint; n,k,n_,k_,n_opt,k_opt:byte; i,j,int
:integer; x, y01, y02, a0, b0, a00, b00, max, min, MinGL, s, Min0, Xfinal,
Xfinal1,   Xfinal2,h,s1,s2:extended;   Ck1,Ck11,Ck12,Ck1_,   Ck1_opt,   Ck11_,
Ck11_opt:matrC_; Ck2, Ck21, Ck22, Ck2_, Ck2_opt, Ck21_, Ck21_opt:matrC_; C1_s,
C2_s:matrC; d:matr;
procedure Konech_Raznoct(fy:vect; n:byte; var dy:matr); var i,j:byte; begin
for j:=0 to n-1 do dy[1,j]:=fy[j+1]-fy[j]; for i:=2 to n do for j:=0 to n-i do
dy[i,j]:=dy[i-1,j+1]-dy[i-1,j] end;
procedure  Newton(U:Vect;  n:byte;  var  Mcoef:matrC);  var  dy:matr;  b:vect;
p,s:extended; j,i:byte; begin Konech_Raznoct(U,n,dy); p:=1; for j:=1 to n do
begin p:=p*j; b[j]:=dy[j,0]/p; end; Mcoef[0]:=U[0]; for i:=1 to n do begin s:=0;
for j:=i to n do s:=s+d[i,j]*b[j]; Mcoef[i]:=s; end end;
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n]; for i:=n-1
downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
procedure Polynomial(x:vect; h, y01, y02:extended; n, k, K_iter:integer; var
C1,C2:matrC); var i,iter:integer; sum1,sum2,sum:extended; ytemp1, ytemp2, fy1,
y1, fy2, y2:vect; A1,A2:matrC; begin for i:=0 to n do y1[i]:=y01; for i:=0 to n
do y2[i]:=y02; if sk then begin for i:=0 to n do y1[i]:=y1_s[i]; for i:=0 to n
do y2[i]:=y2_s[i]; end; for iter:=1 to K_iter do begin for i:=1 to n do begin
ytemp1[i]:=y1[i]; ytemp2[i]:=y2[i]; end; if Method_Iter=1 then begin for i:=0 to
n  do  begin  fy1[i]:=f1(x[i],y1[i],y2[i]);  fy2[i]:=f2(x[i],y1[i],y2[i])  end;
Newton(fy1,n,A1); Newton(fy2,n,A2); C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-
3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;  C2[0]:=y2[0];  C2[-1]:=x[0];  C2[-2]:=h;  C2[-
3]:=n+1;C2[-4]:=k;C2[-5]:=n*h; for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i;
C2[i]:=A2[i-1]*h/i; end; for i:=1 to n do begin y1[i]:=Gorner(C1,x[i]);
y2[i]:=Gorner(C2,x[i]);  end;  end  else  begin  for  i:=0  to  n  do
fy1[i]:=f1(x[i],y1[i],y2[i]); Newton(fy1,n,A1); C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-
2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h; for i:=1 to n+1 do C1[i]:=A1[i-1]*h/i;
for  i:=1  to  n  do  y1[i]:=Gorner(C1,x[i]);  for  i:=0  to  n  do
fy2[i]:=f2(x[i],y1[i],y2[i]); Newton(fy2,n,A2); C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-
2]:=h; C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h; for i:=1 to n+1 do C2[i]:=A2[i-1]*h/i;
for i:=1 to n do y2[i]:=Gorner(C2,x[i]); end; sum1:=0; for i:=1 to n do
sum1:=sum1+abs(y1[i]-ytemp1[i]); sum2:=0; for i:=1 to n do sum2:=sum2+abs(y2[i]-
ytemp2[i]); if sum1>sum2 then sum:=sum1 else sum:=sum2; if (sum<1e-30) or
(sum>=1e50) then break; end end;
procedure Subinterval(k, n, K_iter:integer; a0, b0, Ynach1, Ynach2:extended; var
Ck1,  Ck2:matrC_);  var  hpd,a00,b00,y01,y02,h:extended;  m,pod:longint;  x:vect;
j:byte; begin hpd:=(b0-a0)/exp(k*ln(2)); a00:=a0; b00:=a00+hpd; y01:=Ynach1;
y02:=Ynach2; x[0]:=a0; m:=0; pod:=0; while a00>=b0-hpd/2 do begin h:=(b00-
a00)/n;  for  j:=1  to  n  do  begin  inc(m);  x[j]:=a0+m*h  end;
Polynomial(x,h,y01,y02,n,k,K_iter,Ck1^[pod],Ck2^[pod]);
```

```
y01:=Gorner(Ck1^[pod],x[n]); y02:=Gorner(Ck2^[pod],x[n]); x[0]:=x[n]; inc(pod);
a00:=a00+hpd; b00:=a00+hpd end end;
function Integral(a0, b0, a00, b00:extended; Ck:matrC_):extended; var pod1,
pod2, m:longint; x1,x2,h,ss:extended; begin h:=(b00-a00)/5; m:=0; x1:=a00;
x2:=a00+h; ss:=0; repeat pod1:=trunc((x1-a0)/Ck^[0,-5]); if abs(x2-b0)<1e-15
then pod2:=trunc(exp(Ck^[0,-4]*ln(2)))-1 else pod2:=trunc((x2-a0)/Ck^[0,-5]);
ss:=ss+abs(Gorner(Ck^[pod2],x2)-Gorner(Ck^[pod1],x1)); inc(m); x1:=a00+m*h;
x2:=x1+h; until x2<b00+h/2; Integral:=ss; end;
begin New(Ck1); New(Ck11); New(Ck12); New(Ck1_); New(Ck1_opt); New(Ck11_);
New(Ck11_opt); New(Ck2); New(Ck21); New(Ck22); New(Ck2_); New(Ck2_opt);
New(Ck21_); New(Ck21_opt); a0:=Anach; b0:=Anach+Vel_int; while a0 > Bkonech do
 begin inc(int);MinGL:=MaxExtended; n:=Nmin; repeat k:=Kmin; Min:=MaxExtended;
repeat Subinterval(k,n,K_iter,a0,b0,Ynach1,Ynach2,Ck11,Ck21);
 Subinterval(k+1,n,K_iter,a0,b0,Ynach1,Ynach2,Ck12,Ck22); max:=0; a00:=a0;
b00:=a00+otrezok_integr; i:=0; while b00>=b0 do begin if Nev=1 then begin
s1:=abs(Integral(a0,b0,a00,b00,Ck12)-Integral(a0,b0,a00,b00,Ck11));
s2:=abs(Integral(a0,b0,a00,b00,Ck22)-Integral(a0,b0,a00,b00,Ck21)); end else
begin s1:=abs(Integral(a0,b0,a00,b00,Ck12)-Integral(a0,b0,a00,b00,Ck11))/ (1e-
7+abs(Integral(a0,b0,a00,b00,Ck12))); s2:=abs(Integral(a0,b0,a00,b00,Ck22)-
Integral(a0,b0,a00,b00,Ck21))/(1e-7+abs(Integral(a0,b0,a00,b00,Ck22))); end;
if s1>s2 then s:=s1 else s:=s2; if s>max then max:=s; inc(i);
a00:=a0+i*otrezok_integr; b00:=a00+otrezok_integr; end; if max<Min then begin
Min:=max; Ck1_^:=Ck12^;Ck11_^:=Ck11^;Ck2_^:=Ck22^;Ck21_^:=Ck21^ end; inc(k);
until k>Kmax-1; if Min<MinGL then begin MinGL:=Min; Ck1_opt^:=Ck1_^;
Ck11_opt^:=Ck11_^; Ck2_opt^:=Ck2_^;Ck21_opt^:=Ck21_^ end; inc(n); until n>Nmax;
Subinterval(trunc(Ck1_opt^[0,-4]),trunc(Ck1_opt^[0,-3]-1), K_iter_itog, a0, b0,
Ynach1, Ynach2, Ck1_opt, Ck2_opt); for i:=trunc(n_s/2)+1 to n_s do if x_s[i]<a0
then begin if abs(x_s[i]-b0)<1e-15 then pod:=trunc(exp(Ck1_opt^[0,-4]*ln(2)))-1
else pod:=trunc((x_s[i]-a0)/Ck1_opt^[0,-5]);
y1_s[i]:=Gorner(Ck1_opt^[pod],x_s[i]); y2_s[i]:=Gorner(Ck2_opt^[pod],x_s[i]);
 end; sk:=true; Polynomial(x_s,h_s,y1_s[0],y2_s[0],n_s,5,1,C1_s,C2_s); for i:=0
to n_s do sk:=false; for i:=0 to output do begin x:= b0-i*Vel_int/output;
 if abs(x-b0)<1e-15 then pod:=trunc(exp(Ck1_opt^[0,-4]*ln(2)))-1 else
pod:=trunc((x-a0)/Ck1_opt^[0,-5]); if x>Bkonech then
//writeln(x:8:3,'     ',Gorner(Ck1_opt^[pod],x),'  ',Gorner(Ck2_opt^[pod],x));
  writeln(x:8:3,'     ',abs(Gorner(Ck1_opt^[pod],x)-fun1(x)),'
',abs(Gorner(Ck2_opt^[pod],x)-fun2(x))); end; pod:=trunc(exp(trunc(Ck1_opt^[0,-
4])*ln(2)))-1; Ynach1:=Gorner(Ck1_opt^[pod],b0);
Ynach2:=Gorner(Ck2_opt^[pod],b0); a0:=b0; b0:=a0+Vel_int; end; Dispose(Ck1);
Dispose(Ck11); Dispose(Ck12); Dispose(Ck1_); Dispose(Ck1_opt); Dispose(Ck11_);
Dispose(Ck11_opt); Dispose(Ck2); Dispose(Ck21); Dispose(Ck22); Dispose(Ck2_);
Dispose(Ck2_opt); Dispose(Ck21_);Dispose(Ck21_opt); end; begin sk:=false;
Anach:=1; Bkonech:=400; Ynach1:=fun1(Anach); Ynach2:=fun2(Anach); K_iter:=30;
K_iter_itog:=300; Vel_int:=512; output:=100; Nmin:=1; Nmax:=3; Kmin:=1;
Kmax:=12; Method_Iter:=1; Nev:=2; otrezok_integr:=sqrt(sqrt(abs(vel_int)));
RD1(Ynach1,Ynach2,Vel_int,K_iter,Nmin,Nmax,Kmin,Kmax,output,otrezok_integr);
 Anach:=512; Bkonech:=400; Ynach1:=fun1(Anach); Ynach2:=fun2(Anach);
K_iter:=30; K_iter_itog:=300; Vel_int:=-256; output:=100; Nmin:=3; Nmax:=3;
Kmin:=5; Kmax:=10; Method_Iter:=1; Nev:=2;
otrezok_integr:=-sqrt(sqrt(abs(vel_int)));
RD2(Ynach1,Ynach2,Vel_int,K_iter,Nmin,Nmax,Kmin,Kmax,output,otrezok_integr);
 readln;end.
```

Результат работы программы:

```
   1.000      0.00000000000000E+0000   0.00000000000000E+0000
   6.120      0.00000000000000E+0000   0.00000000000000E+0000
  11.240      0.00000000000000E+0000   0.00000000000000E+0000
  16.360      0.00000000000000E+0000   0.00000000000000E+0000
  21.480      0.00000000000000E+0000   0.00000000000000E+0000
  26.600      0.00000000000000E+0000   0.00000000000000E+0000
  31.720      1.11022302462516E-0016   1.11022302462516E-0016
  36.840      0.00000000000000E+0000   0.00000000000000E+0000
 .....................................................
```

```
313.320      0.00000000000000E+0000   0.00000000000000E+0000
318.440      0.00000000000000E+0000   0.00000000000000E+0000
323.560      0.00000000000000E+0000   0.00000000000000E+0000
328.680      0.00000000000000E+0000   0.00000000000000E+0000
333.800      0.00000000000000E+0000   0.00000000000000E+0000
338.920      0.00000000000000E+0000   0.00000000000000E+0000
.....................................................
494.080      0.00000000000000E+0000   0.00000000000000E+0000
496.640      0.00000000000000E+0000   0.00000000000000E+0000
499.200      0.00000000000000E+0000   0.00000000000000E+0000
501.760      0.00000000000000E+0000   0.00000000000000E+0000
504.320      0.00000000000000E+0000   0.00000000000000E+0000
506.880      0.00000000000000E+0000   0.00000000000000E+0000
509.440      0.00000000000000E+0000   0.00000000000000E+0000
512.000      0.00000000000000E+0000   0.00000000000000E+0000
```

Результат работы этой же программы для решения задачи $y_1' = y_2,\ y_2' = -y_1,$ $y_1(0) = 0,\ y_2(0) = 1,\ y_1(512) = \sin(512),\ y_2(512) = \cos(512)$:

```
  0.000      0.00000000000000E+0000   0.00000000000000E+0000
  2.560      0.00000000000000E+0000   5.42101086242752E-0020
  5.120      5.42101086242752E-0020   0.00000000000000E+0000
  7.680      0.00000000000000E+0000   1.49077798716757E-0019
 10.240      2.16840434497101E-0019   1.62630325872826E-0019
 12.800      3.11708124589583E-0019   1.62630325872826E-0019
 15.360      5.14996031930615E-0019   5.42101086242752E-0020
 17.920      4.87890977618477E-0019   2.71050543121376E-0019
.....................................................
302.080      9.48676900924816E-0019   1.40946282423115E-0018
304.640      1.45689666927739E-0018   7.58941520739853E-0019
307.200      1.68051336735253E-0018   1.62630325872826E-0019
309.760      1.19262238973405E-0018   1.02999206386122E-0018
312.320      4.33680868994202E-0019   1.51788304147970E-0018
314.880      4.33680868994202E-0019   1.46367293285543E-0018
317.440      1.16551733542191E-0018   9.21571846612679E-0019
320.000      1.43656787854329E-0018   1.62630325872826E-0019
322.560      1.24683249835833E-0018   6.50521303491303E-0019
325.120      7.04731412115578E-0019   1.26038502551439E-0018
.....................................................
494.080      1.08420217248550E-0019   2.16840434497101E-0019
496.640      0.00000000000000E+0000   2.16840434497101E-0019
499.200      0.00000000000000E+0000   1.08420217248550E-0019
501.760      5.42101086242752E-0020   5.42101086242752E-0020
504.320      1.62630325872826E-0019   4.74338450462408E-0020
506.880      1.62630325872826E-0019   1.08420217248550E-0019
509.440      1.08420217248550E-0019   0.00000000000000E+0000
512.000      0.00000000000000E+0000   0.00000000000000E+0000
```

Первый столбец данных соответствует значению независимой переменной, второй и третий – значениям абсолютной погрешности приближения соответственно первого и второго компонента решения. Точность приближенного решения сравнима с точностью приближения решения соответствующих задач Коши. Данные результаты подтверждают целесообразность применения кусочно-интерполяционного метода при решении рассмотренного частного случая двухточечной задачи Коши, при этом в отличие от конечно-разностных методов приближение решения

представляется в аналитическом виде, является непрерывным и дифференцируемым.

Таким образом, представлена программная реализация метода кусочно-интерполяционного решения задачи Коши для системы ОДУ с итерационным уточнением при фиксированных значениях параметров. Представлены результаты численного эксперимента по решению на ее основе жесткой и нежесткой задачи Коши на интервалах большой длины, подтверждающие существенное снижение трудоемкости в сравнении с реализацией предложенного метода с программным выбором значений параметров. При этом сохраняется высокая точность кусочно-интерполяционного приближения, которая превышает точность приближения, полученных на основе известных разностных методов на три и более десятичных порядка. Помимо того, в приложении дана программная реализация модификации предложенного метода для решения частного случая двухточечной задачи Коши, даны полученные на ее основе результаты высокоточного кусочно-интерполяционного приближения.

ПРИЛОЖЕНИЕ К ГЛАВЕ 4

П4.1. Приложение к п. 4.1. Вычисление действительной функции двух действительных переменных $u = u(x, t)$ выполняется в прямоугольной области $G = \{ (x, t) \mid x \in [a, b],\ t \in [c, d] \}$ с априори заданной границей абсолютной погрешности при условии минимизации временной сложности. В каждой прямоугольной подобласти, на которые разбивается область G, строится интерполяционный полином Ньютона от двух переменных, который программно переводится в форму алгебраического полинома с числовыми коэффициентами. Степень полинома n и число подобластей $2^{k_x+k_t}$, $k_x, k_t \in \{ 0, 1, \dots \}$, программно варьируются и фиксируются при достижении априори заданной границы абсолютной погрешности. Подробное описание схемы построения метода и его математическое обоснование представлены в главе 4. Ниже приводится его программная реализация. В качестве примера в программе задана функция $u(x, t) = \sin(x + t)$, для которой строится кусочно-интерполяционное приближение в области $G = \{ (x, t) \mid x \in [0, 1],\ t \in [0, 1] \}$ при границе абсолютной погрешности $\varepsilon = 10^{-18}$.

```
program FPI_Newton2;{$APPTYPE CONSOLE} uses SysUtils, Math;
const nn=20; eps=1e-18; a_Gl=0; b_Gl=1; c_Gl=0; d_Gl=1; //Размеры области
type Mass1     = array[0..nn] of extended;
 Mass2     = array[-2..nn,-2..nn] of extended;
 Mass2_int = array[0..nn,0..nn] of integer; Mass4_=^Mass4;
 Mass4     = array[0..nn,0..nn,0..nn,0..nn] of extended;
 Mass5     = array[0..nn,0..nn,0..nn,0..nn,0..nn] of extended; Mass5_=^Mass5;
 Mass7     = array[0..511,0..511] of Mass2;
var     hx,ht,hx_Gl,ht_Gl,xpr,tpr,hxpr,htpr: extended;
   N_ls,Nx,Nt,N_G,N_min,N_max,k_min,k_max,k: shortint; condition: boolean;
   x,t,factorial: Mass1; U,CGl: Mass2; DD_G: Mass4_; MassGl: Mass7;
//интерполируемая функция
function fun(x,t:extended):extended; begin fun:=sin(x+t)end;
//Вычисление целочисленных коэффициентов полиномов
procedure Viet2(var DD:Mass4_); var k,l,i,j,m_: shortint; d,e: Mass2_int;
begin e[1,1]:=1; e[1,0]:=0; for k:=2 to nn do begin e[k,0]:=-e[k-1,0]*(k-1);
for l:=1 to k-1 do e[k,k-l]:=e[k-1,k-l-1]-e[k-1,k-l]*(k-1); e[k,k]:=e[k-1,k-1]
end; for k:=1 to nn do for l:=0 to k do d[l,k]:=e[k,l];for m_:=1 to nn do for
k:=0 to m_ do for i:=0 to k do for j:=0 to m_-k do begin if (k=0) then
Dd[m_,k,i,j]:=d[j,m_-k] else if (m_-k=0) then Dd[m_,k,i,j]:=d[i,k] else
Dd[m_,k,i,j]:=d[i,k]*d[j,m_-k] end end;
//Вычисление конечных разностей
procedure Dual_finite_differences(UU:Mass2; N1:integer; var dz:mass5_);
var i,j,k,m:shortint; begin for i:=0 to N1-1 do for j:=0 to N1-1-i do
```

```
begin dz[1,1,0,i,j]:=UU[i+1,j]-UU[i,j]; dz[1,0,1,i,j]:=UU[i,j+1]-UU[i,j]; end;
for m:=2 to N1 do for k:=0 to m do for i:=0 to N1-m do for j:=0 to N1-i-m do
if m-k<>0 then dz[m,k,m-k,i,j]:=dz[m-1,k,m-k-1,i,j+1]-dz[m-1,k,m-k-1,i,j] else
dz[m,k,m-k,i,j]:=dz[m-1,k-1,m-k,i+1,j]-dz[m-1,k-1,m-k,i,j] end;
procedure preparation; var i:shortint; begin New(DD_G); Viet2(DD_G);
factorial[0]:=1; for i:=1 to nn do factorial[i]:=i*factorial[i-1]; end;
//Вычисление коэффициентов полинома Ньютона от двух переменных
procedure Two_dim_Newton(UU:mass2; N:integer; var C_:mass2);
var i,j,m_,k_:integer; Summa:extended; b_:mass2; dz_:Mass5_;
begin New(dz_); Dual_finite_differences(UU, N, dz_); for i:=0 to N do
for j:=0 to i do b_[i,j]:=dz_[i,j,i-j,0,0]/factorial[j]/factorial[i-j];
for i:=0 to N do for j:=0 to N-i do begin Summa:=0; for m_:=j to N do for k_:=i
to m_-j do Summa:= Summa + b_[m_,k_]*Dd_G[m_,k_,i,j]; if (i=0) and (j=0) then
C_[i,j]:=Summa+UU[0,0] else C_[i,j]:=Summa; end; Dispose(dz_); end;
//Вычисление значений интерполяционного полинома по коэффициентам и степени
function Polinom(xx,tt:extended; CC:mass2):extended;
var i,j,N:shortint; zp,wp,polin,polinom_z:extended;
begin zp:=(xx-CC[-1,0])/CC[-2,0]; wp:=(tt-CC[0,-1])/CC[0,-2];
N:=trunc(CC[-1,-1]); polin:=CC[0,N]; for i:=N-1 downto 0 do begin
polin:=polin*wp; polinom_z:=CC[N-i,i]; for j:=N-i-1 downto 0 do
polinom_z:=polinom_z*zp+CC[j,i]; polin:=polin+polinom_z; end; Result:=polin;end;
//Построение коэффицинтов интерполяционного полинома в подобласти
procedure solve(a0,c0,hx,ht:extended;N_G:integer; var C:mass2);
var i,j:shortint; hhx,hht:extended;
begin hhx:=hx/N_G; hht:=ht/N_G; for i:=0 to N_G do begin x[i]:=a0+i*hhx;
t[i]:=c0+i*hht; end;
for i:=0 to N_G do for j:=0 to N_G do u[i,j]:=fun(x[i],t[j]);
Two_dim_Newton(U, N_G, C); C[-1,0]:=x[0]; C[0,-1]:=t[0]; C[-2,0]:=hhx;
C[0,-2]:=hht; C[-1,-1]:=N_G; end;
///Проверка достижения априори заданного значения абсолютной погрешности
procedure check(K_check:integer;C:mass2);
var xfin, tfin, xpr, tpr, hxpr, htpr: extended;
begin xfin:=C[-1,0]+C[-2,0]*trunc(C[-1,-1]/2);
tfin:=C[0,-1]+C[0,-2]*trunc(C[-1,-1]/2); xpr:=C[-1,0]; tpr:=C[0,-1];
hxpr:=C[-2,0]/K_check; htpr:=C[0,-2]/K_check; repeat repeat
if abs(fun(xpr,tpr)-Polinom(xpr,tpr,C))>eps then
condition:=false; tpr:=tpr+htpr; until tpr>tfin; xpr:=xpr+hxpr;tpr:=C[0,-1];
until xpr>xfin end; //Кусочно-интерполяционное приближение
procedure vpi(Nmin,Nmax,kmin,kmax:integer); var a0,b0,c0,d0,a00,c00:extended;
begin a0:=a_Gl; b0:=a_Gl+hx_Gl; c0:=c_Gl; d0:=c_Gl+ht_Gl; Nx:=0; Nt:=0;
N_G:=Nmin; repeat k:=kmin; repeat Nx:=0;Nt:=0; condition:=true;
hx:=(b0-a0)/exp(k*ln(2)); ht:=(d0-c0)/exp(k*ln(2)); N_ls:=trunc(power(2,k))-1;
a00:=a0; c00:=c0; repeat repeat solve(a00,c00,hx,ht,N_G,MassGL[Nx,Nt]);
check(3,MassGL[Nx,Nt]); c00:=c00+ht/2; inc(Nt); until Nt > N_ls; a00:=a00+hx/2;
inc(Nx); c00:=c0; Nt:=0; until Nx > N_ls;
if condition then begin CGl:=MassGl[0,0]; writeln('n=',N_G:3,' k=',k:3);
hxpr:=(b_Gl-a_Gl)/7; htpr:=(d_Gl-c_Gl)/7; xpr:=a_Gl+hxpr; tpr:=c_Gl+htpr; repeat
repeat Nx:=trunc((xpr-a_Gl)/(CGl[-2,0]*CGl[-1,-1])*2);
Nt:=trunc((tpr-c_Gl)/(CGl[0,-2]* CGl[-1,-1])*2);
if abs(frac((xpr-a_Gl)/(CGl[-2,0]*CGl[-1,-1])))<1e-10 then dec(Nx);
 if abs(frac((tpr-c_Gl)/(CGl[0,-2]*CGl[-1,-1]))) < 1e-10 then dec(Nt);
writeln('x=',xpr:9:7,'   t=',tpr:9:7,'   ',fun(xpr,tpr),'   ',abs(fun(xpr,tpr)-
Polinom(xpr,tpr,MassGl[Nx,Nt]))); tpr:=tpr+htpr; until tpr>d_Gl+htpr/2;
xpr:=xpr+hxpr;tpr:=c_Gl+htpr; until xpr> b_Gl+hxpr/2; exit; end
else k:=k+1; until k>kmax; N_G:=N_G+1;until N_G>Nmax; end;
begin preparation; hx_Gl:=(b_Gl-a_Gl)*2; ht_Gl:=(d_Gl-c_Gl)*2;
N_min:=2; N_max:=13; //границы вариации степени интерполяционного полинома
//границы вариации параметра, определяющего количество подобластей 2^2k
k_min:=0; k_max:=6;
vpi(N_min,N_max,k_min,k_max); Dispose(DD_G); readln; end.
```

Результат работы программы (в 1-й и 2-й колонках значения аргументов x и t соответственно, в 3-й – значение функции, в 4-й – абсолютная погрешность кусочно-полиномиальной интерполяции):

```
0.1428571  0.1428571  2.81842852122210E-01   0.00000000000000E+00
0.1428571  0.2857143  4.15571854993052E-01   5.42101086242752E-20
0.1428571  0.4285714  5.40834213358832E-01   5.42101086242752E-20
0.1428571  0.5714286  6.55077897178519E-01   0.00000000000000E+00
0.1428571  0.7142857  7.55975365146732E-01   0.00000000000000E+00
0.1428571  0.8571429  8.41470984807897E-01   5.42101086242752E-20
0.1428571  1.0000000  9.09822912941124E-01   1.08420217248550E-19
0.2857143  0.1428571  4.15571854993052E-01   5.42101086242752E-20
0.2857143  0.2857143  5.40834213358832E-01   0.00000000000000E+00
0.2857143  0.4285714  6.55077897178519E-01   5.42101086242752E-20
0.2857143  0.5714286  7.55975365146732E-01   1.08420217248550E-19
0.2857143  0.7142857  8.41470984807897E-01   1.08420217248550E-19
0.2857143  0.8571429  9.09822912941124E-01   5.42101086242752E-20
0.2857143  1.0000000  9.59638582966685E-01   5.42101086242752E-20
0.4285714  0.1428571  5.40834213358832E-01   1.08420217248550E-19
0.4285714  0.2857143  6.55077897178519E-01   5.42101086242752E-20
0.4285714  0.4285714  7.55975365146732E-01   0.00000000000000E+00
0.4285714  0.5714286  8.41470984807897E-01   5.42101086242752E-20
0.4285714  0.7142857  9.09822912941124E-01   1.08420217248550E-19
0.4285714  0.8571429  9.59638582966685E-01   0.00000000000000E+00
0.4285714  1.0000000  9.89903076372124E-01   0.00000000000000E+00
0.5714286  0.1428571  6.55077897178519E-01   0.00000000000000E+00
0.5714286  0.2857143  7.55975365146732E-01   1.08420217248550E-19
0.5714286  0.4285714  8.41470984807897E-01   5.42101086242752E-20
0.5714286  0.5714286  9.09822912941124E-01   0.00000000000000E+00
0.5714286  0.7142857  9.59638582966685E-01   0.00000000000000E+00
0.5714286  0.8571429  9.89903076372124E-01   1.08420217248550E-19
0.5714286  1.0000000  9.99999800133368E-01   5.42101086242752E-20
0.7142857  0.1428571  7.55975365146732E-01   0.00000000000000E+00
0.7142857  0.2857143  8.41470984807897E-01   5.42101086242752E-20
0.7142857  0.4285714  9.09822912941124E-01   1.08420217248550E-19
0.7142857  0.5714286  9.59638582966685E-01   0.00000000000000E+00
0.7142857  0.7142857  9.89903076372124E-01   0.00000000000000E+00
0.7142857  0.8571429  9.99999800133368E-01   5.42101086242752E-20
0.7142857  1.0000000  9.89723048859821E-01   4.87890977618477E-19
0.8571429  0.1428571  8.41470984807897E-01   5.42101086242752E-20
0.8571429  0.2857143  9.09822912941124E-01   5.42101086242752E-20
0.8571429  0.4285714  9.59638582966685E-01   5.42101086242752E-20
0.8571429  0.5714286  9.89903076372124E-01   1.08420217248550E-19
0.8571429  0.7142857  9.99999800133368E-01   5.42101086242752E-20
0.8571429  0.8571429  9.89723048859821E-01   5.42101086242752E-20
0.8571429  1.0000000  9.59282195728840E-01   0.00000000000000E+00
1.0000000  0.1428571  9.09822912941124E-01   1.62630325872826E-19
1.0000000  0.2857143  9.59638582966685E-01   0.00000000000000E+00
1.0000000  0.4285714  9.89903076372124E-01   0.00000000000000E+00
1.0000000  0.5714286  9.99999800133368E-01   5.42101086242752E-20
1.0000000  0.7142857  9.89723048859821E-01   4.87890977618477E-19
1.0000000  0.8571429  9.59282195728840E-01   5.42101086242752E-20
1.0000000  1.0000000  9.09297426825682E-01   0.00000000000000E+00
k = 5 N = 7
```

С помощью варьируемого кусочно-интерполяционного метода функция $u(x, t) = \sin(x + t)$ вычислена с точностью порядка 10^{-19} полиномами Ньютона от двух переменных 7-й степени при количестве подобластей $P = 2^{10}$ (последняя строка вывода: `N = 7 k = 5`). При этом, если сохранить коэффициенты полиномов в памяти компьютера, то временная сложность вычисления рассмотренной функции при использовании аналога схемы

Горнера, представленного в главе 4, – $T = 35(t_c + t_y)$. В общем случае временная сложность вычисления действительной функции двух действительных переменных с помощью кусочно-интерполяционного метода при сохранении значений числовых коэффициентов аппроксимирующих полиномов, – $T(n) = (n^2 + 3n)/2 \times (t_c + t_y)$.

Следует отметить принципиальную особенность программной реализации. В силу треугольности расположения узлов интерполяции (см. глава 4, соотношение (4.4)) функция $u = u(x, t)$ интерполируется только к нижней треугольной части, а не во всей прямоугольной области $G = \{ (x, t) \mid x \in [a, b],\ t \in [c, d] \}$. С целью аппроксимации функции именно в G (аналогично, в каждой подобласти G_{ij}, $i = \overline{0, 2^{k_x}}$, $j = \overline{0, 2^{k_t}}$) в программе реализовано изменение подобия расположения узлов. Именно, интерполяционный полином Ньютона от двух переменных всегда строится в описанном прямоугольном треугольнике, катеты которого продолжают нижнюю и левую стороны прямоугольной области G, длина каждого катета вдвое больше продолжаемой стороны. Тогда гипотенуза треугольника, охватывающего все узлы, пройдет через правую вершину прямоугольной области G (рис. П4.1):

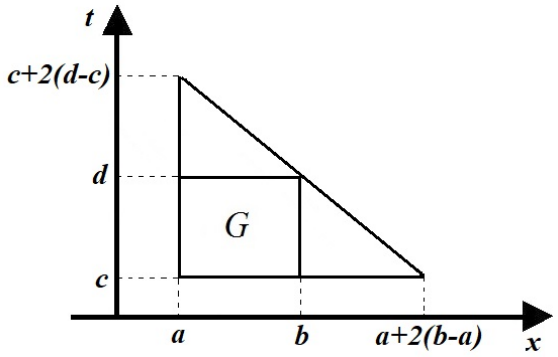

Рис. П4.1. Диаграмма расположения узлов интерполяции при построении интерполяционного полинома Ньютона в области G

Очевидно, узлы интерполяции в данном треугольнике заведомо охватят область G, и значение функции будет интерполироваться уже во всей области G. Аналогично, при построении интерполяционного полинома Ньютона для

прямоугольной подобласти G_{ij}, узлы интерполяции располагаются в описанном прямоугольном треугольнике, охватывающем G_{ij} так, как это показано на рис. П4.1, относительно области G. В частности, катеты описанного прямоугольного треугольника вдвое длинней сторон G_{ij}, по которым они проходят.

Ниже выполняется дополнительное исследование сходимости погрешности приближения функции от двух переменных на основе варьируемой кусочно-полиномиальной аппроксимации. Для наибольшей связности текста кратко повторяются основные предположения и базовые шаги алгоритма построения полинома с числовыми коэффициентами для варьируемой кусочно-полиномиальной аппроксимации. Аппроксимация действительной функции $u = u(x, t)$ от двух действительных переменных в замкнутой прямоугольной области $G = \{ (x, t) \mid x \in [a, b], t \in [c, d] \}$ из области определения данной функции строится следующим образом. Область G разбивается на подобласти G_{ij}, объединение которых покрывает G:

$$G = \bigcup_{j=0}^{P_t - 1} \bigcup_{i=0}^{P_x - 1} G_{ij}, \qquad \text{(П4.1)}$$

где $G_{ij} = \{ (x, t) \mid x \in [x_i, x_{i+1}],\ t \in [t_j, t_{j+1}] \}$, $P_x = 2^{k_x}$, $P_t = 2^{k_t}$, $k_x, k_t \in \{ 0, 1, \dots \}$.

Пусть априори задана граница $\varepsilon > 0$ абсолютной погрешности аппроксимации данной функции. В каждой подобласти G_{ij}, $i = \overline{0, P_x - 1}$, $j = \overline{0, P_t - 1}$, строится интерполяционный полином Ньютона $\Psi_n^{ij}(z, w)$ с равноотстоящими по каждой координате узлами: $h_x^{k_x} = \dfrac{x_{i+1} - x_i}{n}$, $h_t^{k_t} = \dfrac{t_{j+1} - t_j}{n}$ – в соответствии с числом подобластей $P_x \times P_t$, $P_x = 2^{k_x}$, $P_t = 2^{k_t}$, $k_x, k_t \in \{ 0, 1, \dots \}$. При этом $h_x^{k_x}$, $h_t^{k_t}$ – расстояния между узлами, $z = \dfrac{x - x_i}{h_x^{k_x}}$,

$w = \dfrac{t - t_j}{h_t^{k_t}}$. Степень полинома n выбирается одинаковой для всех подобластей и минимальной при условии:

$$| u(x, t) - \Psi_n^{ij}(z, w) | \le \varepsilon \ \ \forall \ (x, t) \in G_{ij}, \ i = \overline{0, P_x - 1}, \ j = \overline{0, P_t - 1}. \qquad (П4.2)$$

Интерполяционный полином Ньютона будет эквивалентно преобразован к виду $\Psi_n^{ij}(z, w) = \sum_{\ell=0}^{n} \sum_{m=0}^{n-\ell} a_{\ell m}^{ij} \, z^{\ell} \, w^{m}$ из известной первоначальной формы:

$$\Psi_n^{ij}(z, w) = u(x_{i0}, t_{j0}) + \sum_{k=1}^{n} \sum_{s=0}^{k} \frac{\Delta^k u_{x^s t^{k-s}}(x_{i0}, t_{j0})}{s!\,(k-s)!} \prod_{\ell=0}^{s} (z - \ell) \prod_{m=0}^{k-s} (w - m), \qquad (П4.3)$$

где $z = \dfrac{x - x_{i0}}{h_x^{k_x}}$, $w = \dfrac{t - t_{j0}}{h_t^{k_t}}$. Вводится обозначение $(x_{i\ell}, t_{jm})$ для узла интерполяции с координатами

$$x_{i\ell} = x_i + \ell h_x, \ t_{jm} = t_j + m h_t, \ \ell = \overline{0, n}, \ m = \overline{0, n - \ell}, \qquad (П4.4)$$

в (П4.3) $\Delta^k u_{x^s t^{k-s}}(x_{i0}, t_{j0})$ – конечные разности k-го порядка в точке (x_{i0}, t_{j0}), которые при $k = \overline{1, n}, \ s = \overline{0, k}$ вычисляются из соотношений:

$$\begin{cases} \Delta_{x^1 t^0} u(x_{i\ell}, t_{jm}) = u(x_{i\,\ell+1}, t_{jm}) - u(x_{i\,\ell}, t_{jm}), \\ \Delta_{x^0 t^1} u(x_{i\ell}, t_{jm}) = u(x_{i\,\ell}, t_{jm+1}) - u(x_{i\,\ell}, t_{jm}), \\ \Delta^k_{x^s t^{k-s}} u(x_{i\ell}, t_{jm}) = \Delta^{k-1}_{x^s t^{k-s-1}} u(x_{i\ell}, t_{jm+1}) - \Delta^{k-1}_{x^s t^{k-s-1}} u(x_{i\ell}, t_{jm}), \text{ при } k - s > 0, \\ \Delta^k_{x^s t^{k-s}} u(x_{i\ell}, t_{jm}) = \Delta^{k-1}_{x^{s-1} t^{k-s}} u(x_{i\ell+1}, t_{jm}) - \Delta^{k-1}_{x^{s-1} t^{k-s}} u(x_{i\ell}, t_{jm}), \text{ при } k - s = 0, \end{cases} \qquad (П4.5)$$

где $k = \overline{2, n}$, $s = \overline{0, k}$, $\ell = \overline{0, n-k}$, $m = \overline{0, n-k-\ell}$.

После вычисления факториалов из (П4.3) вводятся обозначения

$$b_{ks} = \frac{\Delta^k u_{x^s t^{k-s}}(x_{i0}, t_{j0})}{s!\,(k-s)!}.$$

В данных обозначениях полином примет вид:

$$\Psi_n^{ij}(z, w) = u(x_{i0}, t_{j0}) + \sum_{k=0}^{n} \sum_{s=0}^{k} b_{ks} \prod_{\ell=0}^{s} (z - \ell) \prod_{m=0}^{k-s} (w - m). \qquad (П4.6)$$

Коэффициенты полиномов $P_s(z)=\sum_{\ell=0}^{s} d_{s\ell}\, z^{\ell}$, $P_{k-s}(w)=\sum_{m=0}^{k-s} \widetilde{d}_{(k-s)m}\, w^{m}$, представленных в (П4.6) в виде разложения по корням, вычисляются по схеме (1.9). При уже вычисленных коэффициентах имеет место равенство

$$\prod_{\ell=0}^{s}(z-\ell)\prod_{m=0}^{k-s}(w-m)=\sum_{\ell=0}^{s}\sum_{m=0}^{k-s} D_{ks\ell m}\, z^{\ell} w^{m}, \qquad \text{(П4.7)}$$

где $D_{ks\ell m}=d_{s\ell}\,\widetilde{d}_{(k-s)\,m}$.

Далее полином (П4.7) переводится в форму полинома с явными значениями числовых коэффициентов. В результате

$$\Psi_n^{ij}(z,w)=\sum_{\ell=0}^{n}\sum_{m=0}^{n-\ell} a_{\ell m}^{ij}\, z^{\ell} w^{m}, \qquad \text{(П4.8)}$$

где

$$a_{00}^{ij}=u(x_{i0},t_{j0})+\sum_{k=0}^{n}\sum_{s=0}^{k} b_{ks}\, D_{ks00},\ a_{\ell m}^{ij}=\sum_{k=m}^{n}\sum_{s=\ell}^{k-m} b_{ks}\, D_{ks\ell m},\ \ \ell=\overline{1,n},\ \ m=\overline{1,n-\ell}.$$

При выполнении аппроксимации условие (П4.2) на каждой подобласти G_{ij} из (П4.1) проверяется в равноотстоящих по направлениям координат точках области с шагом меньшим шага интерполяции, значения полинома (П4.8) вычисляются по аналогу схемы Горнера

$$\begin{aligned}\Psi_n^{ij}(z,w)=(\ldots[\,(a_{0n}^{ij}w+a_{1\,n-1}^{ij}z+a_{0\,n-1}^{ij})\,w+\\ +(a_{2n-2}^{ij}z+a_{1\,n-2}^{ij})\,z+a_{0\,n-2}^{ij}\,]\,w+\ldots)\,w+\ldots+a_{00}^{ij}.\end{aligned} \qquad \text{(П4.9)}$$

Алгоритм аппроксимации заданной функции $u(x,t)$ на основе выбора полинома вида (П4.8) строится по двухмерной аналогии с одномерным случаем. Задаются равномерно расположенные проверочные точки в каждом прямоугольнике разбиения (П4.1). Последовательным обходом каждой из таких точек в каждом прямоугольнике проверяется выполнение неравенства (П4.2) для полинома (П4.8). Проверка начинается с значений $n=1,\ k_x=0,\ k_t=0$. Если во всех проверочных точках неравенство (П4.2) выполняется, то параметры $n=1,\ k_x=0,\ k_t=0$ сохраняются в качестве найденных для синтезируемого

алгоритма. Если хоть в одной проверочной точке неравенство (П4.2) нарушается, то к степени полинома добавляется единица: $n := n+1,\ k_x := 0,\ k_t := 0$, и процесс построения полинома (П4.8) данной степени с проверкой неравенства (П4.2) возобновляется во всех проверочных точках. Если в каждой из них неравенство (П4.2) оказалось верным, то данный набор n, k_x, k_t считается найденным, иначе степень n снова увеличивается на единицу при сохранении значений k_x, k_t. Так продолжается до достижения априори заданной границы степени $n \le N$. Если при ее достижении неравенство (П4.2) хотя бы в одной проверочной точке все же не выполняется, то полагается $n := 1,\ k_x := k_x + 1,\ k_t := k_t + 1,$ и все построения и проверки возобновляются для данного разбиения и данной степени. Если во всех проверочных точках (П4.2) верно, то параметры найдены. Иначе при сохранении k_x, k_t выполняется $n := n+1$, и так далее, до достижения границы степени $n \le N$, а затем, в продолжение процесса, априори заданных границ $k_x \le K_x,\ k_t \le K_t$. В качестве искомых фиксируются наименьшие n, k_x, k_t, для которых неравенство (П4.2) верно во всех проверочных точках.

Если функция часто встречается коэффициенты полинома (П4.8) целесообразно сохранить в памяти компьютера в соответствии всем элементам разбиения (П4.1). В таком случае алгорим целесообразно изменить с целью выбора наименьшей степени интерполяционного полинома (вначале увеличивать значения k_x, k_t, затем n).

В этом случае для приближенного вычисления полиномом (П4.8) функции $u(x,t),\ (x,t) \in G,$ выполняется дешифрация индексов подобласти $G_{ij}, \quad i = \mathrm{int}((x-a)/\rho_x),\ j = \mathrm{int}((t-c)/\rho_t), \quad \rho_x = x_{i+1} - x_i, \quad \rho_t = t_{j+1} - t_j,$ $i = \overline{0,\ P_x - 1},\ j = \overline{0,\ P_t - 1},\ x \in [x_i, x_{i+1}),\ t \in [t_j, t_{j+1})$.

Оценка сходимости варьируемой кусочно-полиномиальной аппроксимации. Всюду ниже предполагается непрерывность и однократная

непрерывная дифференцируемость функции $u(x,t)$ в области $G=\{(x,t)\,|\,x\in[a,b],t\in[c,d]\}$ с соответствующими односторонними частными производными на границах области. Пусть имеет место разбиение (П4.1), на элементах которого функция аппроксимируется согласно (П4.2). Расстояния между соседними узлами интерполяции в подобласти $G_{ij}=\{(x,t)\,|\,x\in[x_i,x_{i+1}],\ t\in[t_j,t_{j+1}]\}$ из (П4.1) обозначаются, как и выше, в соответствии с числом подобластей $P_x\times P_t$: $h_x^{k_x}=\dfrac{x_{i+1}-x_i}{n}$, $h_t^{k_t}=\dfrac{t_{j+1}-t_j}{n}$. Более подробно, при каждой паре k_x и k_t для полинома степени n, по построению,

$$h_x^{k_x}=x_{i\,(\ell+1)}-x_{i\ell}=\frac{x_{i+1}-x_i}{n}\qquad\forall\,\ell=\overline{0,\,n-1},\qquad h_t^{k_t}=t_{j(m+1)}-t_{jm}=\frac{t_{j+1}-t_j}{n}$$

$\forall\,m=\overline{0,\,n-\ell-1}$, где $x_{i\ell}$, t_{jm} – равноотстоящие по каждой координате узлы интерполяции на подобласти G_{ij} из (П4.1), пара значений h_x^0, h_t^0 соответствует отсутствию разбиения на начальной области $G=\{(x,t)\,|\,x\in[a,b],t\in[c,d]\}$.

В этих обозначениях интерполяционный полином Ньютона в подобласти сохраняет вид (П4.3), где $z=\dfrac{x-x_{i0}}{h_x^{k_x}}$, $w=\dfrac{t-t_{j0}}{h_t^{k_t}}$, $h_x^{k_x}=\dfrac{b-a}{2^{k_x}n}$, $h_t^{k_t}=\dfrac{d-c}{2^{k_t}n}$

$\forall\,i=\overline{0,\,2^{k_x}-1}$, $\forall\,j=\overline{0,\,2^{k_t}-1}$, или, в эквивалентном виде:

$$\begin{aligned}&\Psi_n^{ij}(x,t)=u(x_{i0},t_{j0})+\\&+\sum_{k=1}^{n}\sum_{s=0}^{k}\frac{\Delta^k u_{x^s t^{k-s}}(x_{i0},t_{j0})}{s!\,(k-s)!\,(h_x^{k_x})^{s+1}\,(h_t^{k_t})^{k-s+1}}\prod_{\ell=0}^{s}(x-x_{i\ell})\prod_{m=0}^{k-s}(t-t_{jm}),\end{aligned}\qquad\text{(П4.10)}$$

$h_x^{k_x}$ и $h_t^{k_t}$ – расстояния между соседними узлами.

Всюду в дальнейшем предполагается, что для всех полиномов (П4.3) вариация степени ограничена постоянными значениями: $2\le n\le N$, где $N=\text{const}$. Кроме того, формально снимается ограничение на количество

подобластей: 2^{k_x} и 2^{k_t} рассматриваются как переменные с неограниченным ростом показателей степени.

В данных предположениях и обозначениях требуется оценить разность

$$\delta = |u(x, t) - \Psi_n^{ij}(x, t)| \qquad \text{(П4.11)}$$

$$\forall (x, t) \mid x \in [x_i, x_{i+1}],\ t \in [t_j, t_{j+1}],\ \forall\, i = \overline{0,\, 2^{k_x} - 1},\ \forall\, j = \overline{0,\, 2^{k_t} - 1}.$$

Применение формулы конечных приращений к функции под знаком модуля в (П4.11) в замкнутой прямоугольной подобласти размером $h_x^{k_x} \times h_t^{k_t}$ между произвольно выбранными соседними узлами интерполяции в рассматриваемой подобласти влечет:

$$u(x, t) - \Psi_n^{ij}(x, t) - (u(x_{i\ell}, t_{jm}) - \Psi_n^{ij}(x_{i\ell}, t_{jm})) =$$
$$\left(u'_x(x_{i\ell} + \theta(x - x_{i\ell}), t_{jm} + \theta(t - t_{jm})) - \left(\Psi_n^{ij}(x_{i\ell} + \theta(x - x_{i\ell}), t_{jm} + \theta(t - t_{jm}))\right)'_x \right)(x - x_{i\ell}) +$$
$$\left(u'_t(x_{i\ell} + \theta(x - x_{i\ell}), t_{jm} + \theta(t - t_{jm})) - \left(\Psi_n^{ij}(x_{i\ell} + \theta(x - x_{i\ell}), t_{jm} + \theta(t - t_{jm}))\right)'_t \right)(t - t_{jm}),$$

где $0 < \theta < 1$, (ℓ, m) – номер узла интерполяции в подобласти G_{ij} из (П4.1), такой что $x_{i\,(\ell+1)} < x < x_{i\ell}$, $t_{j\,(m+1)} < t < t_{jm}$, $\ell = \overline{0,\, n-1}$, $m = \overline{0,\, n-\ell-1}$. Отсюда с учетом условия интерполяции, согласно которому $u(x_{i\ell}, t_{jm}) - \Psi_n^{ij}(x_{i\ell}, t_{jm}) = 0$, следует, что

$$\left| u(x, t) - \Psi_n^{ij}(x, t) \right| \le \left(\max_G |u'_x(x, t)| + \max_G |\left(\Psi_n^{ij}(x, t)\right)'_x| \right) h_x^{k_x} +$$
$$+ \left(\max_G |u'_t(x, t)| + \max_G |\left(\Psi_n^{ij}(x, t)\right)'_t| \right) h_t^{k_t}, \qquad \text{(П4.12)}$$

где $h_x^{k_x} = x_{i\,(\ell+1)} - x_{i\ell} = \dfrac{x_{i+1} - x_i}{n}$ $\forall\, \ell = \overline{0,\, n-1}$, $\forall\, i = \overline{0,\, 2^{k_x} - 1}$,

$h_t^{k_t} = t_{j(m+1)} - t_{jm} = \dfrac{t_{j+1} - t_j}{n}$ $\forall\, m = \overline{0,\, n-\ell-1}$, $\forall\, j = \overline{0,\, 2^{k_t} - 1}$.

По непрерывности частных производных $u'_x(x,t)$ и $u'_t(x,t)$ в замкнутой области G выполняются соотношения:

$$\max_G |u'_x(x,t)| = c_1,\ \max_G |u'_t(x,t)| = c_2, \qquad (П4.13)$$

где $c_1 = \text{const}, c_2 = \text{const}\ \ \forall (x,t) \in G$.

С целью подстановки в (П4.12) аналогичное (П4.13) соотношение требуется получить для $\left(\Psi_n^{ij}(x,t)\right)'_x$ и $\left(\Psi_n^{ij}(x,t)\right)'_t$ одновременно для каждой из подобластей G_{ij}, $\forall i = \overline{0, 2^{k_x}-1}$, $\forall j = \overline{0, 2^{k_t}-1}$ при всех n, $2 \le n \le N$, где $N = \text{const}$.

Согласно (П4.10) $z = \dfrac{x - x_{i0}}{h_x^{k_x}}$, $w = \dfrac{t - t_{j0}}{h_t^{k_t}}$, поэтому

$$\left(\Psi_n^{ij}(x,t)\right)'_x = \frac{1}{h_x^{k_x}} \left(\Psi_n^{ij}(z,w)\right)'_z,$$

$$\left(\Psi_n^{ij}(z,w)\right)'_z = \sum_{k=1}^{n} \sum_{s=0}^{k} \frac{\Delta^k u_{x^s t^{k-s}}(x_{i0}, t_{j0})}{s!\,(k-s)!} \left(\prod_{\ell=0}^{s}(z-\ell)\right)'_z \prod_{m=0}^{k-s}(w-m)$$

Из леммы П6.1 следует

$$\left(\prod_{\ell=0}^{s}(z-\ell)\right)'_z \le (s+1)\times(s+1)!\ \frac{n^{s+1}-1}{n-1},\ \prod_{m=0}^{k-s}(w-m) \le (k-s+1)!\ \frac{n^{k-s+2}-1}{n-1}.$$

Отсюда

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right| \le$$

$$\le \sum_{k=1}^{n} \sum_{s=0}^{k} \frac{\left|\Delta^k u_{x^s t^{k-s}}(x_{i0}, t_{j0})\right|}{s!\,(k-s)!}\ (s+1)\times(s+1)!\ \frac{n^{s+1}-1}{n-1}\ (k-s+1)!\ \frac{n^{k-s+2}-1}{n-1}$$

или

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right| \le$$
$$\le \sum_{k=1}^{n}\sum_{s=0}^{k}\left|\Delta^k u_{x^s t^{k-s}}(x_{i0},t_{j0})\right|(s+1)^2\frac{n^{s+1}-1}{n-1}(k-s+1)\frac{n^{k-s+2}-1}{n-1}. \qquad (П4.14)$$

Оценим $\Delta^k u_{x^s t^{k-s}}(x_{i0},t_{j0})$. Из (П4.5) следует, при $k-s>0$,

$$\left.\begin{aligned}
&\Delta^k_{x^s t^{k-s}} u(x_{i\ell},t_{jm}) = \Delta^{k-1}_{x^s t^{k-s-1}} u(x_{i\ell},t_{jm+1}) - \Delta^{k-1}_{x^s t^{k-s-1}} u(x_{i\ell},t_{jm}) = \\
&= \left(\Delta^{k-2}_{x^s t^{k-s-2}} u(x_{i\ell},t_{jm+2}) - \Delta^{k-2}_{x^s t^{k-s-2}} u(x_{i\ell},t_{jm+1})\right) - \\
&\qquad - \left(\Delta^{k-2}_{x^s t^{k-s-2}} u(x_{i\ell},t_{jm+1}) - \Delta^{k-2}_{x^s t^{k-s-2}} u(x_{i\ell},t_{jm})\right) = \\
&= \left(\Delta^{k-3}_{x^s t^{k-s-3}} u(x_{i\ell},t_{jm+3}) - \Delta^{k-3}_{x^s t^{k-s-3}} u(x_{i\ell},t_{jm+2})\right) - \\
&\qquad - \left(\Delta^{k-3}_{x^s t^{k-s-3}} u(x_{i\ell},t_{jm+2}) - \Delta^{k-3}_{x^s t^{k-s-3}} u(x_{i\ell},t_{jm+1})\right) - \\
&- \left(\Delta^{k-3}_{x^s t^{k-s-3}} u(x_{i\ell},t_{jm+2}) - \Delta^{k-3}_{x^s t^{k-s-3}} u(x_{i\ell},t_{jm+1})\right) + \\
&\qquad + \left(\Delta^{k-3}_{x^s t^{k-s-3}} u(x_{i\ell},t_{jm+1}) - \Delta^{k-3}_{x^s t^{k-s-3}} u(x_{i\ell},t_{jm})\right) = \ldots
\end{aligned}\right\} \qquad (П4.15)$$

где в дальнейшем полагается $l=0, m=0$. Разложение по данной схеме продолжается до выполнения равенства $k-s=0$, после чего оно продолжится по аналогичной схеме в соответствии с последней из формул (П4.5). Каждый шаг разложения удваивает число конечных разностей и одновременно снижает порядок разности на единицу. Поэтому в результате 2^k последовательных шагов получится выражение конечной разности k-го порядка через значения функции $u(x_{i\ell},t_{jm})$ в узлах интерполяции, число этих значений без приведения подобных на шагах разложения составит 2^k. Отсюда вытекает неравенство:

$$\left|\Delta^k u_{x^s t^{k-s}}(x_{i0},t_{j0})\right| \le 2^k \max_{\ell,m}\left|u(x_{i\ell},t_{jm})\right|.$$

Тем более,

$$\left|\Delta^k u_{x^s t^{k-s}}(x_{i0},t_{j0})\right| \le 2^k c_3\ ,$$
$$c_3 = \text{const}\ \forall k \le N\ ,\ \forall\ i = \overline{0,\ 2^{k_x}-1}\ ,\ \forall\ j = \overline{0,\ 2^{k_t}-1}\ \forall G_{ij} \in G, \qquad (П4.16)$$
$$c_3 = \max_G\left|u(x,t)\right|$$

$$\left|\Delta^k u_{x^s t^{k-s}}(x_{i0}, t_{j0})\right| \le 2^n c_3. \qquad (П4.17)$$

С учетом $n \le N$ приходим к оценке:

$$\left|\Delta^k u_{x^s t^{k-s}}(x_{i0}, t_{j0})\right| \le c_4 ,$$
$$c_4 = \text{const}\ \forall k \le N ,\ \forall\ i = \overline{0,\ 2^{k_x} - 1} ,\ \forall\ j = \overline{0,\ 2^{k_t} - 1}\ \forall G_{ij} \in G, \qquad (П4.18)$$
$$c_4 = 2^N \max_G \left|u(x, t)\right| .$$

Таким образом, имеет место

Лемма П4.1. Конечные разности высшего порядка из полинома (П4.10) ограничены во всей области G. При этом для них выполнены оценки (П4.16) – (П4.18).

Данного в лемме утверждения недостаточно для искомой оценки. Согласно (П4.12) требуется доказать ограниченность не только конечных разностей, но и их отношения к шагу, иначе он сократится с шагом за скобками вследствие выражений производных:

$$\left(\Psi_n^{ij}(x, t)\right)'_x = \frac{1}{h_x^{k_x}} \left(\Psi_n^{ij}(z, w)\right)'_z ,\ \left(\Psi_n^{ij}(x, t)\right)'_t = \frac{1}{h_t^{k_t}} \left(\Psi_n^{ij}(z, w)\right)'_w . \qquad (П4.19)$$

Для требуемой оценки можно заметить, что проделанное ниже (П4.14) выражение конечных разностей через значения функции в результате 2^k последовательных шагов дает выражение конечной разности k-го порядка через значения функции $u(x_{i\ell}, t_{jm})$ в узлах интерполяции, число этих значений без приведения подобных на шагах разложения составит 2^k, при этом они сдвоены как конечные разности на шаге:

$$u(x_{i\ell+1}, t_{jm}) - u(x_{i\ell}, t_{jm}), \qquad (П4.20)$$

или,

$$u(x_{i\ell}, t_{jm+1}) - u(x_{i\ell}, t_{jm}), \qquad (П4.21)$$

где m и ℓ – соответственные индексы узлов интерполяции. Далее, для (П4.20)

$$u(x_{i\,\ell+1}, t_{jm}) - u(x_{i\,\ell}, t_{jm}) = u(x_{i\,\ell} + h_x^{k_x}, t_{jm}) - u(x_{i\,\ell}, t_{jm}).$$

Очевидно,

$$\begin{aligned} &\left| u(x_{i\,\ell} + h_x^{k_x}, t_{jm}) - u(x_{i\,\ell}, t_{jm}) \right| \le \\ &\le \max_{0 \le \Delta t \le h_t^{k_t}} \left| u(x_{i\,\ell} + h_x^{k_x}, t_{jm} + \Delta t) - u(x_{i\,\ell}, t_{jm}) \right|. \end{aligned} \qquad (\text{П4.22})$$

Для выражения под знаком модуля правой части последнего неравенства по формуле конечных приращений получится:

$$\begin{aligned} &u(x_{i\,\ell} + h_x^{k_x}, t_{jm} + \Delta t) - u(x_{i\,\ell}, t_{jm}) = \\ &= u'_x\,(x_{i\ell} + \theta(x_{i\ell+1} - x_{i\ell}), t_{jm} + \theta\,\Delta t) \times (x_{i\ell+1} - x_{i\ell}) + \\ &\qquad + u'_t\,(x_{i\ell} + \theta(x_{i\ell+1} - x_{i\ell}), + \theta\,\Delta t) \times \Delta t, \end{aligned}$$

где $0 < \theta < 1$. Отсюда

$$\begin{aligned} &\left| u(x_{i\,\ell} + h_x^{k_x}, t_{jm} + \Delta t) - u(x_{i\,\ell}, t_{jm}) \right| \le \\ &\le \left| u'_x\,(x_{i\ell} + \theta\, h_x^{k_x}, t_{jm} + \theta\,\Delta t) \times h_x^{k_x} + u'_t\,(x_{i\ell} + \theta h_x^{k_x}, t_{jm} + \theta\Delta t) \times \Delta t \right|. \end{aligned}$$

Следовательно,

$$\begin{aligned} &\left| u(x_{i\,\ell} + h_x^{k_x}, t_{jm} + \Delta t) - u(x_{i\,\ell}, t_{jm}) \right| \le \\ &\le \max_{\theta_1, \theta_2} \left| u'_x\,(x_{i\ell} + \theta\, h_x^{k_x}, t_{jm} + \theta\Delta t) \right| \times h_x^{k_x} + \max_{\theta_1, \theta_2} \left| u'_t\,(x_{i\ell} + \theta h_x^{k_x}, t_{jm} + \theta\Delta t) \right| \times \Delta t. \end{aligned}$$

Очевидно,

$$\begin{aligned} &\left| u(x_{i\,\ell} + h_x^{k_x}, t_{jm} + \Delta t) - u(x_{i\,\ell}, t_{jm}) \right| \le \\ &\le \max_G \left| u'_x \right| \times h_x^{k_x} + \max_G \left| u'_t \right| \times h_t^{k_t}. \end{aligned}$$

Наконец,

$$\begin{aligned} &\left| u(x_{i\,\ell} + h_x^{k_x}, t_{jm} + \Delta t) - u(x_{i\,\ell}, t_{jm}) \right| \le \\ &\le \max\left(c_1, c_2\right) \times \max\left(h_x^{k_x}, h_t^{k_t}\right), \end{aligned}$$

где c_1, c_2 из (П4.13). Переход к максимуму по Δt в неравенстве влечет:

$$\max_{0\le\Delta t\le h_t^{k_t}}\left|u(x_{i\ell}+h_x^{k_x},t_{jm}+\Delta t)-u(x_{i\ell},t_{jm})\right|\le$$
$$\le\max\left(c_1,c_2\right)\times\max\left(h_x^{k_x},h_t^{k_t}\right).$$

Согласно (П4.22), по транзитивности,

$$\left|u(x_{i\ell+1},t_{jm})-u(x_{i\ell},t_{jm})\right|\le\max\left(c_1,c_2\right)\times\max\left(h_x^{k_x},h_t^{k_t}\right),$$

где c_1, c_2 из (П4.13). Аналогично, для (П4.21):

$$\left|u(x_{i\ell},t_{jm+1})-u(x_{i\ell},t_{jm})\right|\le\max\left(c_1,c_2\right)\times\max\left(h_x^{k_x},h_t^{k_t}\right),$$

Отсюда с учетом (П4.15) следует, что

$$\left|\Delta^k u_{x^s t^{k-s}}(x_{i0},t_{j0})\right|\le 2^{k-1}\widetilde{c}_3\ \max\left(h_x^{k_x},h_t^{k_t}\right),$$
$$\widetilde{c}_3=\text{const}\ \forall k\le N\ ,\ \forall\ i=\overline{0,\,2^{k_x}-1}\ ,\ \forall\ j=\overline{0,\,2^{k_t}-1}\ \forall G_{ij}\in G,\qquad(\text{П4.23})$$
$$\widetilde{c}_3=\max\left(c_1,c_2\right).$$

Подстановка (П4.23) в (П4.14) влечет:

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right|\le\widetilde{c}_3\sum_{k=1}^{n}\sum_{s=0}^{k}2^{k-1}\,(s+1)^2\,\frac{n^{s+1}-1}{n-1}\,(k-s+1)\,\frac{n^{k-s+2}-1}{n-1}\max\left(h_x^{k_x},h_t^{k_t}\right),$$

или,

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right|\le$$
$$\le\widetilde{c}_3\max\left(h_x^{k_x},h_t^{k_t}\right)\frac{1}{(n-1)^2}\sum_{k=1}^{n}\ 2^{k-1}\sum_{s=0}^{k}\ (s+1)^2\left(n^{s+1}-1\right)(k-s+1)\left(n^{k-s+2}-1\right).$$

Тем более,

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right|<\widetilde{c}_3\max\left(h_x^{k_x},h_t^{k_t}\right)\frac{1}{(n-1)^2}\sum_{k=1}^{n}\ 2^{k-1}\sum_{s=0}^{k}\ (s+1)^2\,n^{s+1}\,n^{k-s+2}\,(k-s+1)\ ,$$

или,

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right|<\widetilde{c}_3\max\left(h_x^{k_x},h_t^{k_t}\right)\frac{1}{(n-1)^2}\sum_{k=1}^{n}\ 2^{k-1}n^{k+3}\sum_{s=0}^{k}\ (s+1)^2\ (k-s+1)\ .$$

Отсюда

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right| < \widetilde{c}_3 \max\left(h_x^{k_x}, h_t^{k_t}\right)\frac{n^4}{(n-1)^2}\sum_{k=1}^{n} 2^{k-1}n^{k-1}\sum_{s=0}^{k}(s+1)^2\,(k-s+1)\ .$$

Заменяя $(s+1)(k-s+1)$ на максимум $\dfrac{(k+2)^2}{4}$, приходим к неравенству:

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right| < \widetilde{c}_3 \max\left(h_x^{k_x}, h_t^{k_t}\right)\frac{n^4}{(n-1)^2}\sum_{k=1}^{n}\frac{(k+2)^2}{4}(2n)^{k-1}\sum_{s=0}^{k}(s+1)\ ,$$

из которого следует:

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right| < \widetilde{c}_3 \max\left(h_x^{k_x}, h_t^{k_t}\right)\frac{n^4}{(n-1)^2}\frac{(n+2)^2}{4}\sum_{k=1}^{n}(2n)^{k-1}\frac{(k+2)(k+1)}{2}\ ,$$

что влечет:

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right| < \widetilde{c}_3 \max\left(h_x^{k_x}, h_t^{k_t}\right)\frac{n^4}{(n-1)^2}\times\frac{(n+2)^3(n+1)}{8}\times\frac{(2n)^n-1}{2n-1}\ .$$

Отсюда

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right| < \widetilde{c}_3\left((2n)^n-1\right)\frac{(n+2)^3n^2}{2}\max\left(h_x^{k_x}, h_t^{k_t}\right),$$

Тем более

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_z\right| < \frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\max\left(h_x^{k_x}, h_t^{k_t}\right).$$

Аналогично,

$$\left|\left(\Psi_n^{ij}(z,w)\right)'_w\right| < \frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\max\left(h_x^{k_x}, h_t^{k_t}\right).$$

Таким образом, имеет место

Лемма П4.2. Частные производные (П4.19) от полинома (П4.10) ограничены во всей области G из неравенств

$$\left|\left(\Psi_n^{ij}(x,t)\right)'_x\right| < \frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\frac{1}{h_x^{k_x}}\max\left(h_x^{k_x}, h_t^{k_t}\right),$$

$$\left|\left(\Psi_n^{ij}(x,t)\right)'_t\right| < \frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\frac{1}{h_t^{k_t}}\max\left(h_x^{k_x}, h_t^{k_t}\right).$$

Подставляя два последних неравенства в (П4.12), получим

$$\left|u(x,t)-\Psi_n^{ij}(x,t)\right| \le$$

$$\le\left(c_3+\frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\frac{1}{h_x^{k_x}}\max\left(h_x^{k_x},h_t^{k_t}\right)\right)h_x^{k_x}+$$

$$+\left(c_3+\frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\frac{1}{h_t^{k_t}}\max\left(h_x^{k_x},h_t^{k_t}\right)\right)h_t^{k_t}.$$

Отсюда

$$\left|u(x,t)-\Psi_n^{ij}(x,t)\right| \le$$

$$\le\left(\frac{c_3h_x^{k_x}+\frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\ \max\left(h_x^{k_x},h_t^{k_t}\right)}{h_x^{k_x}}\right)h_x^{k_x}+$$

$$+\left(\frac{c_3h_t^{k_t}+\frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\ \max\left(h_x^{k_x},h_t^{k_t}\right)}{h_t^{k_t}}\right)h_t^{k_t},$$

или,

$$\left|u(x,t)-\Psi_n^{ij}(x,t)\right| \le$$

$$\le c_3h_x^{k_x}+\frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\ \max\left(h_x^{k_x},h_t^{k_t}\right)+$$

$$+c_3h_t^{k_t}+\frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\ \max\left(h_x^{k_x},h_t^{k_t}\right).$$

Следовательно,

$$\left|u(x,t)-\Psi_n^{ij}(x,t)\right| \le$$

$$\le c_3\max\left(h_x^{k_x},h_t^{k_t}\right)+\frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\ \max\left(h_x^{k_x},h_t^{k_t}\right)+$$

$$+c_3\max\left(h_x^{k_x},h_t^{k_t}\right)+\frac{1}{2}\widetilde{c}_3\left((2N)^N-1\right)(N+2)^3N^2\ \max\left(h_x^{k_x},h_t^{k_t}\right).$$

Очевидно,

$$\left| u(x,t) - \Psi_n^{ij}(x,t) \right| \le \widetilde{C} \max\left(h_x^{k_x}, h_t^{k_t} \right),$$

где

$$\widetilde{C} = 2\left(c_3 + \frac{1}{2}\widetilde{c}_3 \left((2N)^N - 1 \right)(N+2)^3 N^2 \right).$$

Окончательно,

$$\left| u(x,t) - \Psi_n^{ij}(x,t) \right| \le \widetilde{C} \max\left(\frac{b-a}{2^{k_x} n}, \frac{d-c}{2^{k_t} n} \right). \qquad \text{(П4.24)}$$

Таким образом, доказана

Теорема П4.1. Пусть функция $u(x,t)$ определена и непрерывна, в области $G = \{ (x,t) \mid x \in [a,b], t \in [c,d] \}$ имеет непрерывные частные производные по обеим переменным. На элементах разбиения (П4.1) функция аппроксимируется согласно (П4.2). Тогда $\forall n,\ 2 \le n \le N$, последовательность полиномов $\Psi_n^{ij}(x,t)$ равномерно сходится к функции $u(x,t)$ во всей области G при $k_x \to \infty$ и $k_t \to \infty$, скорость сходимости оценивается из неравенства (П4.24).

Поскольку узлы интерполяции на смежных границах подобластей общие, то имеет место

Следствие П4.1. Утверждение теоремы выполняется с тем дополнением, что множество полиномов $\Psi_n^{ij}(x,t)$ непрерывно на каждом разбиении (П4.1).

П4.2. Приложение к п. 4.8. Аналогично случаю кусочно-интерполяционного приближения решения задачи Коши для системы ОДУ ниже с целью снижения трудоемкости приводится и исследуется программная реализация кусочно-интерполяционного метода приближенного решения задачи Коши для уравнения переноса с фиксированными значениями параметров n, k, ℓ, где n – степень интерполяционных полиномов Ньютона от двух переменных, k – параметр, определяющий число подобластей $P = 2^{2k}$

разбиения исходной области приближения, ℓ – число итераций в каждой подобласти. Непосредственно ниже с подробными комментариями приводится программа, реализующая описанный метод для случая приближенного решения задачи Коши для линейного уравнения переноса вида $u_t + a(x,t)u_x = f(x,t)$, $u(x,0) = \varphi(x)$, в прямоугольной области $G = \{(x,t) \mid x \in [a,b],\ t \in [0,T]\}$. Результаты работы программы приводятся на примере кусочно-интерполяционного решения задачи $u'_t + u'_x = 0$, $u(x,0) = \cos(x)$ в области $G = \{(x,t) \mid x \in [0,1], t \in [0,1]\}$.

```
program FPI_IVP_FOR_PDE; // Ut+a(x,t)*Ux = f(x,t), u(x,0)=IC(x)
{$APPTYPE CONSOLE} uses SysUtils; const nn=13; CoefA=1;
a_Gl=0; b_Gl=1; c_Gl=0; d_Gl=1; //Размеры области приближения
kk=0;//число, определяющее количество подобластей разбиения 2^2k
N_G=13;  // степень интерполяционных полиномов Ньютона
K_iter=40; {число итераций} outputGl_x=10; outputGl_t=10;{кол-во точек вывода}
type Mass1 = array[-nn..nn] of extended;
Mass2=array[-nn..nn,-nn..nn] of extended;
Mass2_int=array[0..nn,0..nn] of integer; Mass4_=^Mass4;
Mass4 = array[0..nn,0..nn,0..nn,0..nn] of extended; Mass5_=^Mass5;
Mass5 = array[0..nn,0..nn,0..nn,0..nn,0..nn] of extended;
Mass7=array[0..500,1..3] of Mass2; Mass8_=^Mass8; Mass8=array[-16..16] of Mass7;
var oprpol,sdvigr,sdvigl,kl,kr,Nsloya: shortint; podT,podtGl,Ksl: longint;
hx,ht,hhx,hht,c00,d00,x1,x2,a0,c0,x1_t,x2_t,line:extended;  hour,minut,sec,msec:
word; tnach,tkonech: extended; x,xx,t,factorial: Mass1; U,C,CX,Fi,C1,CT: Mass2;
C_last_C,C_last_R,C_last_L: Mass2; MasPolT:Mass7; MasGl:Mass8_;
U0,U0L,U0R: Mass1;  DD_G : Mass4_;
function f(x, t:extended):extended; begin f:=0 end;
function a(x, t:extended):extended; begin a:=1 end;
function IC(x:extended):extended; begin IC:=cos(x)end;
function Solution(x,t:extended):extended; begin Solution:=cos(x-t)end;
//Вычисление целочисленных коэффициентов полиномов
Procedure Viet2(var DD:Mass4_); var k,l,i,j,m_: Shortint; d,e: Mass2_int;
begin e[1,1]:=1; e[1,0]:=0; for k:=2 to nn do begin e[k,0]:=-e[k-1,0]*(k-1);
for l:=1 to k-1 do e[k,k-l]:=e[k-1,k-l-1]-e[k-1,k-l]*(k-1); e[k,k]:=e[k-1,k-1]
end; for k:=1 to nn do for l:=0 to k do d[l,k]:=e[k,l];
for m_:=1 to nn do for k:=0  to  m_ do for i:=0 to k do for j:=0  to m_-k do
begin if (k=0) then Dd[m_,k,i,j]:=d[j,m_-k] else if (m_-k=0) then
Dd[m_,k,i,j]:=d[i,k] else Dd[m_,k,i,j]:=d[i,k]*d[j,m_-k] end end;
//Вычисление конечных разностей
Procedure Dual_finite_differences(UU:Mass2; N1:integer; var dz:mass5_);
var i,j,k,m:shortint;
begin for i:=0 to N1-1 do for j:=0 to N1-1-i do begin
dz[1,1,0,i,j]:=UU[i+1,j]-UU[i,j]; dz[1,0,1,i,j]:=UU[i,j+1]-UU[i,j]; end;
for m:=2 to N1 do for k:=0 to m do for i:=0 to N1-m do for j:=0 to N1-i-m do
if m-k<>0 then dz[m,k,m-k,i,j]:=dz[m-1,k,m-k-1,i,j+1]-dz[m-1,k,m-k-1,i,j] else
dz[m,k,m-k,i,j]:=dz[m-1,k-1,m-k,i+1,j]-dz[m-1,k-1,m-k,i,j] end;
procedure preparation; var i:shortint;
begin New(DD_G);New(MasGl); Viet2(DD_G); factorial[0]:=1;
for   i:=1   to   nn   do   factorial[i]:=i*factorial[i-1];   kl:=trunc(N_G/2);
kr:=trunc(N_G/2); sdvigl:=trunc(N_G/4); sdvigr:=trunc(N_G/4); end;
//Вычисление коэффициентов полинома Ньютона от двух переменных
procedure Two_dim_Newton(UU:mass2; N:integer; var C_:mass2);
var i,j,m_,k_:integer; Summa:extended; b_:mass2; dz_:Mass5_;
```

```
begin New(dz_); Dual_finite_differences(UU, N, dz_);
for i:=0 to N do for j:=0 to i do
b_[i,j]:=dz_[i,j,i-j,0,0]/factorial[j]/factorial[i-j];
for i:=0 to N do for j:=0 to N-i do begin Summa:=0;
for m_:=j to N do for k_:=i to m_-j do
Summa:= Summa + b_[m_,k_]*Dd_G[m_,k_,i,j]; if (i=0) and (j=0) then
C_[i,j]:=Summa+UU[0,0] else C_[i,j]:=Summa; end; C_[-1,0]:=x[0]; C_[0,-1]:=t[0];
C_[-2,0]:=hhx; C_[0,-2]:=hht; C_[-1,-1]:=N; Dispose(dz_); end;
//Вычисление значений полинома по коэффициентам и степени (аналог схемы Горнера)
function Polinom(xx,tt:extended; CC:mass2):extended;
var i,j,N:shortint; zp,wp,polin,polinom_z:extended;
begin zp:=(xx-CC[-1,0])/CC[-2,0]; wp:=(tt-CC[0,-1])/CC[0,-2];
N:=trunc(CC[-1,-1]); polin:=CC[0,N]; for i:=N-1 downto 0 do begin
polin:=polin*wp; polinom_z:=CC[N-i,i]; for j:=N-i-1 downto 0 do
polinom_z:=polinom_z*zp+CC[j,i]; polin:=polin+polinom_z; end; Result:=polin;end;
//Построение интерполяционных полиномов для реализации итерационного уточнения
procedure Utochnenie; var i,j:word;
begin Two_dim_Newton(U, N_G, C);
//Вычисление массива коэффициентов для аппроксимации UX
for i:=0 to N_G-1 do for j:=0 to N_G-1 do CX[i,j]:=C[i+1,j]*(i+1)/hhx;
CX[-1,0]:=x[0]; CX[0,-1]:=t[0]; CX[-2,0]:=hhx; CX[0,-2]:=hht; CX[-1,-1]:=N_G-1;
//Вычисление условий интерполяции функции правой части
for i:=0 to N_G-1 do for j:=0 to N_G-1 do
Fi[i,j]:=f(x[i],t[j])-a(x[i],t[j])*Polinom(x[i],t[j],CX);
Two_dim_Newton(Fi, N_G-1, CT); for i:=0 to N_G-1 do for j:=0 to N_G-1 do
C1[i,j]:=CT[i,j]/(j+1); C1[-1,0]:=x[0]; C1[0,-1]:=t[0]; C1[-2,0]:=hhx;
C1[0,-2]:=hht; C1[-1,-1]:=N_G-1; end;
//Определение начальных условий в подобласти
Procedure Nachusl; vari,j:shortint;
begin case OprPol of
0: begin if podT=0 then for j:=0 to N_G do for i:=0 to N_G-j do u[i,j]:=IC(x[i])
else begin for i:=0 to N_G do if x[i]<x1 then U0[i]:=Polinom(x[i],t[0],C_last_L)
else if x[i]<x2 then U0[i]:=Polinom(x[i],t[0],C_last_C) else
U0[i]:=Polinom(x[i],t[0],C_last_R); for i:=0 to N_G do u[i,0]:=U0[i]; for j:= 1
to N_G do for i:= 0 to N_G-j do U[i,j]:= Polinom(x[i],t[j],C_last_C) end; end;
-1:begin if podT=0 then for j:=0 to N_G do for i:=0 to N_G-j do u[i,j]:=IC(x[i])
else begin for i:=0 to N_G do if x[i]<x1 then begin
u[i,0]:=Polinom(x[i],t[0],C_last_l); U0L[i]:=u[i,0] end else begin
u[i,0]:=Polinom(x[i],t[0],C_last_C); U0L[i]:=u[i,0] end; for j:=1 to N_G do
for i:=0 to N_G-j do u[i,j]:=u[i,0]; end; end;
1:begin if podT=0 then begin for j:=0 to N_G do for i:=0 to N_G-j do
u[i,j]:=IC(x[i]); end else begin for i:=0 to N_G do for j:=0 to N_G-i do
u[i,j]:=Polinom(x[i],t[j],C_last_R); for i:=0 to N_G do if x[i]<x2 then begin
u[i,0]:= Polinom(x[i],t[0],C_last_C);U0R[i]:=u[i,0] end else begin
u[i,0]:= Polinom(x[i],t[0],C_last_R);U0R[i]:=u[i,0] end; end; end; end; end;
//Реализация итерационного уточнения
procedure IterUt(U00:Mass1;var C_last:Mass2); var i,j:shortint;iter:word;
begin for iter:= 1 to K_iter do begin Utochnenie; for i:=0 to N_G do for j:=0 to
N_G do if podT=0 then U[i,j]:= Polinom(x[i],t[j],C1)*(t[j]-t[0])+IC(x[i]) else
U[i,j]:= Polinom(x[i],t[j],C1)*(t[j]-t[0])+U00[i]; end; C_last:=C;
C_last[-1,0]:=x[0]; C_last[0,-1]:=t[0]; C_last[-2,0]:=hhx;  C_last[0,-2]:=hht;
C_last[-1,-1]:=N_G; end;
{Определение начальных условий для интерполяции в области, полученной сдвигом
текущей подобласти приближения влево}
procedure NachuslNewLevo; var i,j:shortint;
begin dec(Nsloya); if podT=0 then for j:=0 to N_G do for i:=0 to N_G-j do
u[i,j]:=IC(x[i]) else begin for i:=kl to N_G do for j:=1 to N_G-i do
u[i,j]:=Polinom(x[i],t[j],C_last_C); for i:=0 to kl-1 do for j:=0 to N_G-i do
begin podtGl:=trunc(t[j]/line); if (t[j]<>0) and (frac(t[j]/line)=0) then
podtGl:=podtGl-1; x1_t:=MasGl[Nsloya,podTGl,1][-3,0];
x2_t:=MasGl[Nsloya,podTGl,1][-4,0]; if x[i]<x1_t then
u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,1]) else if x[i]<x2_t then
```

```
u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,2]) else
u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,3]) end; for i:=0 to N_G do
if x[i]<x1 then begin podtGl:=trunc(t[0]/line);
if (t[0]<>0) and (frac(t[0]/line)=0) then podtGl:=podtGl-1;
x1_t:=MasGl[Nsloya,podTGl,1][-3,0]; x2_t:=MasGl[Nsloya,podTGl,1][-4,0];
if  x[i]<x1_t  then  u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,1])  else  if
x[i]<x2_t then u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,2]) else
u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,3]); U0L[i]:=u[i,0] end else begin
u[i,0]:=Polinom(x[i],t[0],C_last_C); U0L[i]:=u[i,0] end; end; inc(Nsloya); end;
{Построение полиномиальных приближений в подобласти с выполнением сдвига текущей
подобласти влево и вправо}
procedure solve(Nsl:integer); var i,j:shortint;
begin hhx:=hx/N_G; hht:=(d00-c00)/N_G; for i:=-N_G to N_G do x[i]:=a0 + i*hhx;
for j:=1 to N_G do t[j]:= t[0] + j*hht;{Центральный полином} OprPol:=0; Nachusl;
 IterUt(U0,C_last_C);{Сдвиг влево} for i:=kl to N_G do xx[i]:= x[i-kl]; for i:=1
to kl do xx[kl-i]:=x[0]-i*hhx; for i:=0 to N_G do x[i]:= xx[i]; for i:=-1 downto
-N_G  do  x[i]:=  x[0]-abs(i)*hhx;  if  (Nsl=-Ksl  div  2)  then  begin  OprPol:=-1;
Nachusl end else NachuslNewLevo; IterUt(U0L,C_last_L); for i:=0 to N_G do
x[i]:= a0 + i*hhx;{Сдвиг вправо} for i:=0 to N_G-kr do xx[i]:= x[i+kr];
for i:=0 to kr-1 do xx[N_G-i]:= x[N_G]+(kr-i)*hhx; for i:=0 to N_G do
 x[i]:= xx[i]; OprPol:=1; Nachusl; IterUt(U0R,C_last_R);
for i:=0 to N_G do x[i]:= a0 + i*hhx;
end;
//Построение кусочно-интерполяционного приближения по вертикальным слоям
procedure RD; var Dteck:extended;
begin x[0]:=a0; t[0]:=c0; podT:=0; c00:=c0; d00:=c00+ht;
Dteck:=d_Gl+ht*(Ksl-Nsloya); while (t[0]+line-Dteck) <=1e-15 do begin
solve(Nsloya); x1:=x[sdvigl]; x2:=x[N_G-sdvigr];
{Сохранение массивов для следующих слоев}
C_last_L[-3,0]:=x1; C_last_L[-4,0]:=x2; MasPolT[podT,1]:=C_last_L;
MasPolT[podT,2]:=C_last_C; MasPolT[podT,3]:=C_last_R;
c00:=c00+hht; d00:=c00+ht; t[0]:=t[0]+line; inc(podT); end;
MasGl[Nsloya]:=MasPolT; end;
procedure outputGl; {Вывод результата}
var Nsl,j,i:shortint; err,xpr,tpr,P:extended;
begin writeln;writeln('Output of results'); for j:=1 to outputGl_t do begin
tpr:=c_Gl+j*(d_Gl-c_Gl)/outputGl_t; podtGl:=trunc((tpr-c_Gl)/line);
if (tpr<>c_Gl) and (abs(frac(tpr/line))<1e-18)  then podtGl:=podtGl-1;
for i:=0 to outputGl_x do  begin xpr:=a_Gl+i*(b_Gl-a_Gl)/outputGl_x;
Nsl:=trunc((xpr-a_Gl)/hx)+1; if  (xpr<>a_Gl)  and  (abs(frac(xpr/hx))<1e-18)  then
Nsl:=Nsl-1;  x1:=MasGl[Nsl,podTGl,1][-3,0];  x2:=MasGl[Nsl,podTGl,1][-4,0];  if
xpr<x1 then begin P:=Polinom(xpr,tpr,MasGl[Nsl,podTGl,1]);
err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,1])-solution(xpr,tpr)) end else
if xpr<x2 then begin P:=Polinom(xpr,tpr,MasGl[Nsl,podTGl,2]);
err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,2])-solution(xpr,tpr)) end else begin
P:=Polinom(xpr,tpr,MasGl[Nsl,podTGl,3]);
err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,3])-solution(xpr,tpr)); end;
writeln(' t=',tpr:7:4,' x=',xpr:7:4,' ',P,' ',err) end; readln; end; end;
//Построение кусочно-интерполяционного приближения по горизонатальным слоям
procedure RD_GL;
begin Ksl:=trunc(exp(kk*ln(2))); a0:=a_Gl; c0:=c_Gl; Nsloya:=-Ksl div 2;
hx:=(b_Gl-a_Gl)/Ksl; ht:=hx/2; line:=ht/10; a0:=a_Gl-(abs(Nsloya)+1)*hx;
repeat RD; inc(Nsloya); a0:=a0+hx; until Nsloya>Ksl; end; begin preparation;
writeln('The calculation process is in progress.'); writeln('Please wait...');
tnach:=Gettime; RD_GL; tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln;
writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec); outputGl;
Dispose(DD_G);Dispose(MasGl); end.
```

Результат работы программы:

```
t= 0.1000 x= 0.0000  9.95004165278026E-0001  2.16840434497101E-0019
```

```
t= 0.1000 x= 0.1000  1.00000000000000E+0000  2.16840434497101E-0019
t= 0.1000 x= 0.2000  9.95004165278026E-0001  2.71050543121376E-0019
t= 0.1000 x= 0.3000  9.80066577841242E-0001  1.62630325872826E-0019
t= 0.1000 x= 0.4000  9.55336489125606E-0001  2.71050543121376E-0019
t= 0.1000 x= 0.5000  9.21060994002885E-0001  0.00000000000000E+0000
t= 0.1000 x= 0.6000  8.77582561890373E-0001  2.16840434497101E-0019
t= 0.1000 x= 0.7000  8.25335614909678E-0001  1.08420217248550E-0019
t= 0.1000 x= 0.8000  7.64842187284488E-0001  1.08420217248550E-0019
t= 0.1000 x= 0.9000  6.96706709347165E-0001  1.08420217248550E-0019
t= 0.1000 x= 1.0000  6.21609968270664E-0001  5.42101086242752E-0020

t= 0.2000 x= 0.0000  9.80066577841242E-0001  0.00000000000000E+0000
t= 0.2000 x= 0.1000  9.95004165278026E-0001  2.16840434497101E-0019
t= 0.2000 x= 0.2000  1.00000000000000E+0000  3.25260651745651E-0019
t= 0.2000 x= 0.3000  9.95004165278026E-0001  2.16840434497101E-0019
t= 0.2000 x= 0.4000  9.80066577841242E-0001  5.42101086242752E-0020
t= 0.2000 x= 0.5000  9.55336489125606E-0001  2.71050543121376E-0019
t= 0.2000 x= 0.6000  9.21060994002885E-0001  5.42101086242752E-0020
t= 0.2000 x= 0.7000  8.77582561890373E-0001  3.25260651745651E-0019
t= 0.2000 x= 0.8000  8.25335614909678E-0001  2.16840434497101E-0019
t= 0.2000 x= 0.9000  7.64842187284488E-0001  2.71050543121376E-0019
t= 0.2000 x= 1.0000  6.96706709347165E-0001  5.42101086242752E-0020

t= 0.3000 x= 0.0000  9.55336489125606E-0001  1.62630325872826E-0019
t= 0.3000 x= 0.1000  9.80066577841242E-0001  2.16840434497101E-0019
t= 0.3000 x= 0.2000  9.95004165278026E-0001  0.00000000000000E+0000
t= 0.3000 x= 0.3000  1.00000000000000E+0000  2.71050543121376E-0019
t= 0.3000 x= 0.4000  9.95004165278026E-0001  4.33680868994202E-0019
t= 0.3000 x= 0.5000  9.80066577841242E-0001  0.00000000000000E+0000
t= 0.3000 x= 0.6000  9.55336489125606E-0001  3.79470760369927E-0019
t= 0.3000 x= 0.7000  9.21060994002885E-0001  1.08420217248550E-0019
t= 0.3000 x= 0.8000  8.77582561890373E-0001  3.79470760369927E-0019
t= 0.3000 x= 0.9000  8.25335614909678E-0001  1.62630325872826E-0019
t= 0.3000 x= 1.0000  7.64842187284488E-0001  4.87890977618477E-0019

t= 0.4000 x= 0.0000  9.21060994002885E-0001  5.42101086242752E-0020
t= 0.4000 x= 0.1000  9.55336489125606E-0001  5.96311194867027E-0019
t= 0.4000 x= 0.2000  9.80066577841242E-0001  1.08420217248550E-0019
t= 0.4000 x= 0.3000  9.95004165278026E-0001  5.42101086242752E-0020
t= 0.4000 x= 0.4000  1.00000000000000E+0000  2.71050543121376E-0019
t= 0.4000 x= 0.5000  9.95004165278026E-0001  3.25260651745651E-0019
t= 0.4000 x= 0.6000  9.80066577841242E-0001  1.62630325872826E-0019
t= 0.4000 x= 0.7000  9.55336489125606E-0001  5.42101086242752E-0019
t= 0.4000 x= 0.8000  9.21060994002885E-0001  5.42101086242752E-0020
t= 0.4000 x= 0.9000  8.77582561890373E-0001  4.87890977618477E-0019
t= 0.4000 x= 1.0000  8.25335614909678E-0001  0.00000000000000E+0000

t= 0.5000 x= 0.0000  8.77582561890373E-0001  2.16840434497101E-0019
t= 0.5000 x= 0.1000  9.21060994002885E-0001  1.62630325872826E-0019
t= 0.5000 x= 0.2000  9.55336489125606E-0001  1.08420217248550E-0019
t= 0.5000 x= 0.3000  9.80066577841242E-0001  1.62630325872826E-0019
t= 0.5000 x= 0.4000  9.95004165278026E-0001  3.79470760369927E-0019
t= 0.5000 x= 0.5000  1.00000000000000E+0000  1.08420217248550E-0019
t= 0.5000 x= 0.6000  9.95004165278026E-0001  4.33680868994202E-0019
t= 0.5000 x= 0.7000  9.80066577841242E-0001  0.00000000000000E+0000
t= 0.5000 x= 0.8000  9.55336489125606E-0001  5.42101086242752E-0019
t= 0.5000 x= 0.9000  9.21060994002885E-0001  2.16840434497101E-0019
t= 0.5000 x= 1.0000  8.77582561890373E-0001  4.87890977618477E-0019

t= 0.6000 x= 0.0000  8.25335614909678E-0001  5.96311194867027E-0019
t= 0.6000 x= 0.1000  8.77582561890373E-0001  6.50521303491303E-0019
t= 0.6000 x= 0.2000  9.21060994002885E-0001  1.62630325872826E-0019
t= 0.6000 x= 0.3000  9.55336489125606E-0001  2.16840434497101E-0019
t= 0.6000 x= 0.4000  9.80066577841242E-0001  2.71050543121376E-0019
t= 0.6000 x= 0.5000  9.95004165278026E-0001  3.79470760369927E-0019
t= 0.6000 x= 0.6000  1.00000000000000E+0000  0.00000000000000E+0000
t= 0.6000 x= 0.7000  9.95004165278026E-0001  3.79470760369927E-0019
t= 0.6000 x= 0.8000  9.80066577841242E-0001  5.42101086242752E-0020
t= 0.6000 x= 0.9000  9.55336489125606E-0001  7.04731412115578E-0019
t= 0.6000 x= 1.0000  9.21060994002885E-0001  4.87890977618477E-0019

t= 0.7000 x= 0.0000  7.64842187284488E-0001  1.08420217248550E-0019
t= 0.7000 x= 0.1000  8.25335614909678E-0001  5.96311194867027E-0019
```

```
t= 0.7000 x= 0.2000  8.77582561890373E-0001  7.58941520739853E-0019
t= 0.7000 x= 0.3000  9.21060994002885E-0001  5.42101086242752E-0020
t= 0.7000 x= 0.4000  9.55336489125606E-0001  3.25260651745651E-0019
t= 0.7000 x= 0.5000  9.80066577841242E-0001  3.25260651745651E-0019
t= 0.7000 x= 0.6000  9.95004165278026E-0001  2.71050543121376E-0019
t= 0.7000 x= 0.7000  1.00000000000000E+0000  1.08420217248550E-0019
t= 0.7000 x= 0.8000  9.95004165278026E-0001  1.62630325872826E-0019
t= 0.7000 x= 0.9000  9.80066577841242E-0001  0.00000000000000E+0000
t= 0.7000 x= 1.0000  9.55336489125606E-0001  4.87890977618477E-0019

t= 0.8000 x= 0.0000  6.96706709347165E-0001  1.08420217248550E-0019
t= 0.8000 x= 0.1000  7.64842187284488E-0001  2.16840434497101E-0019
t= 0.8000 x= 0.2000  8.25335614909678E-0001  3.79470760369927E-0019
t= 0.8000 x= 0.3000  8.77582561890373E-0001  5.42101086242752E-0019
t= 0.8000 x= 0.4000  9.21060994002885E-0001  2.16840434497101E-0019
t= 0.8000 x= 0.5000  9.55336489125606E-0001  4.33680868994202E-0019
t= 0.8000 x= 0.6000  9.80066577841242E-0001  4.33680868994202E-0019
t= 0.8000 x= 0.7000  9.95004165278026E-0001  1.62630325872826E-0019
t= 0.8000 x= 0.8000  1.00000000000000E+0000  0.00000000000000E+0000
t= 0.8000 x= 0.9000  9.95004165278026E-0001  2.16840434497101E-0019
t= 0.8000 x= 1.0000  9.80066577841242E-0001  1.08420217248550E-0019

t= 0.9000 x= 0.0000  6.21609968270664E-0001  1.62630325872826E-0019
t= 0.9000 x= 0.1000  6.96706709347165E-0001  1.08420217248550E-0019
t= 0.9000 x= 0.2000  7.64842187284488E-0001  5.42101086242752E-0020
t= 0.9000 x= 0.3000  8.25335614909678E-0001  5.42101086242752E-0019
t= 0.9000 x= 0.4000  8.77582561890373E-0001  5.96311194867027E-0019
t= 0.9000 x= 0.5000  9.21060994002885E-0001  1.62630325872826E-0019
t= 0.9000 x= 0.6000  9.55336489125606E-0001  4.33680868994202E-0019
t= 0.9000 x= 0.7000  9.80066577841242E-0001  3.25260651745651E-0019
t= 0.9000 x= 0.8000  9.95004165278026E-0001  4.87890977618477E-0019
t= 0.9000 x= 0.9000  1.00000000000000E+0000  5.42101086242752E-0020
t= 0.9000 x= 1.0000  9.95004165278026E-0001  3.25260651745651E-0019

t= 1.0000 x= 0.0000  5.40302305868140E-0001  7.04731412115578E-0019
t= 1.0000 x= 0.1000  6.21609968270664E-0001  3.79470760369927E-0019
t= 1.0000 x= 0.2000  6.96706709347165E-0001  2.16840434497101E-0019
t= 1.0000 x= 0.3000  7.64842187284488E-0001  5.42101086242752E-0020
t= 1.0000 x= 0.4000  8.25335614909678E-0001  4.87890977618477E-0019
t= 1.0000 x= 0.5000  8.77582561890373E-0001  6.50521303491303E-0019
t= 1.0000 x= 0.6000  9.21060994002885E-0001  0.00000000000000E+0000
t= 1.0000 x= 0.7000  9.55336489125606E-0001  4.87890977618477E-0019
t= 1.0000 x= 0.8000  9.80066577841242E-0001  3.25260651745651E-0019
t= 1.0000 x= 0.9000  9.95004165278026E-0001  4.33680868994202E-0019
t= 1.0000 x= 1.0000  1.00000000000000E+0000  1.08420217248550E-0019

Calculation time 0:0:5:911
```

Таким образом, построено кусочно-непрерывное приближение рассматриваемой задачи с абсолютной погрешностью, не превышающей порядка 10^{-19} в прямоугольной области единичной длины.При этом время работы программы составило $5\,s\,911\,ms$, параметры метода: $n=13$, $k=0$, $\ell=40$.

В случае решения начально-краевой задачи для линейного уравнения переноса вида $u_t + a(x,t)u_x = f(x,t)$, $u(x,0)=\varphi(x)$, $u(0,t)=\psi(x)$, в приведенный выше листинг программы вносятся незначительные изменения с целью учета значений функции граничного условия при построении кусочно-интерполяционных приближений. Непосредственно ниже представленна

программа кусочно-интерполяционного приближения с итерационным уточнением начально-краевой задачи для уравнения переноса при фиксированных значениях параметров метода. Результат работы программы приводится на примере решения задачи $u_t + u_x = 2e^{x+t}$, $u(x,0) = e^x$, $u(0,t) = e^t$, которая соответствует задаче (4.84), представленной в главе 4, при значениях параметров уравнения $k = 1$, $A_3 = 0$ и имеет точное аналитическое решение $u(x,t) = e^{x+t}$, использованное для вывода абсолютной погрешности приближения.

```
program example_4_1;   // Ut+a(x,t)*Ux = f(x,t), u(x,0)=IC(x), u(0,t)=BC(t)
{$APPTYPE CONSOLE} uses SysUtils; const nn=13; CoefA=1; a_Gl=0; b_Gl=1; c_Gl=0;
d_Gl=1; kk=0; N_G=13; K_iter=20; outputGl_x=7; outputGl_t=7;
type Mass1 = array[-nn..nn] of extended; Mass2 = array[-nn..nn,-nn..nn] of
extended; Mass2_int = array[0..nn,0..nn] of integer; Mass4_=^Mass4;
Mass4 = array[0..nn,0..nn,0..nn,0..nn] of extended; Mass5_=^Mass5;
Mass5 = array[0..nn,0..nn,0..nn,0..nn,0..nn] of extended;
Mass7=array[0..500,1..3] of Mass2; Mass8_=^Mass8; Mass8=array[-16..16] of Mass7;
var oprpol,sdvigr,sdvigl,kl,kr,Nsloya: shortint; podT,podtGl,Ksl: longint;
hx,ht,hhx,hht,c00,d00,x1,x2,a0,c0,x1_t,x2_t,line: extended; hour,minut,sec,msec:
word; tnach,tkonech: extended; x,xx,t,factorial: Mass1; U,C,CX,Fi,C1,CT: Mass2;
 C_last_C,C_last_R,C_last_L: Mass2; MasPolT:Mass7; MasGl:Mass8_; U0,U0L,U0R:
Mass1; DD_G : Mass4_;
function f(x, t:extended):extended; begin f:=2*exp(t+x) end;
function a(x, t:extended):extended; begin a:=1 end;
function IC(x:extended):extended; begin IC:=exp(x) end;
function BC(t:extended):extended;begin BC:=exp(t) end;
function Solution(x,t:extended):extended; begin Solution:=exp(x+t) end;
Procedure Viet2(var DD:Mass4_); var k,l,i,j,m_: Shortint; d,e: Mass2_int;
begin e[1,1]:=1; e[1,0]:=0; for k:=2 to nn do begin e[k,0]:=-e[k-1,0]*(k-1);
for l:=1 to k-1 do e[k,k-l]:=e[k-1,k-l-1]-e[k-1,k-l]*(k-1); e[k,k]:=e[k-1,k-1]
end; for k:=1 to nn do for l:=0 to k do d[l,k]:=e[k,l]; for m_:=1 to nn do for
k:=0 to m_ do for i:=0  to k do for j:=0  to m_-k do begin if (k=0) then
Dd[m_,k,i,j]:=d[j,m_-k]  else if (m_-k=0)  then Dd[m_,k,i,j]:=d[i,k]  else
Dd[m_,k,i,j]:=d[i,k]*d[j,m_-k] end end;
Procedure Dual_finite_differences(UU:Mass2; N1:integer; var dz:mass5_);
var i,j,k,m:shortint; begin for i:=0 to N1-1 do for j:=0 to N1-1-i do begin
dz[1,1,0,i,j]:=UU[i+1,j]-UU[i,j]; dz[1,0,1,i,j]:=UU[i,j+1]-UU[i,j]; end;
for m:=2 to N1 do for k:=0 to m do for i:=0 to N1-m do for j:=0 to N1-i-m do
if m-k<>0 then dz[m,k,m-k,i,j]:=dz[m-1,k,m-k-1,i,j+1]-dz[m-1,k,m-k-1,i,j] else
dz[m,k,m-k,i,j]:=dz[m-1,k-1,m-k,i+1,j]-dz[m-1,k-1,m-k,i,j] end;
procedure preparation; var i:shortint; begin New(DD_G);New(MasGl); Viet2(DD_G);
factorial[0]:=1; for i:=1 to nn do factorial[i]:=i*factorial[i-1];
kl:= trunc(N_G/2);kr:=trunc(N_G/2); sdvigl:=trunc(N_G/4);
sdvigr:=trunc(N_G/4);end;
procedure Two_dim_Newton(UU:mass2; N:integer; var C_:mass2);
var i,j,m_,k_:integer; Summa:extended; b_:mass2; dz_:Mass5_; begin New(dz_);
 Dual_finite_differences(UU, N, dz_); for i:=0 to N do for j:=0 to i do
b_[i,j]:=dz_[i,j,i-j,0,0]/factorial[j]/factorial[i-j]; for i:=0 to N do
for j:=0 to N-i do begin Summa:=0; for m_:=j to N do for k_:=i to m_-j do
Summa:= Summa + b_[m_,k_]*Dd_G[m_,k_,i,j]; if (i=0) and (j=0) then
C_[i,j]:=Summa+UU[0,0] else C_[i,j]:=Summa; end; C_[-1,0]:=x[0]; C_[0,-1]:=t[0];
C_[-2,0]:=hhx; C_[0,-2]:=hht; C_[-1,-1]:=N; Dispose(dz_); end;
function Polinom(xx,tt:extended; CC:mass2):extended; var i,j,N:shortint;
```

```
zp,wp,polin,polinom_z:extended; begin zp:=(xx-CC[-1,0])/CC[-2,0];
wp:=(tt-CC[0,-1])/CC[0,-2]; N:=trunc(CC[-1,-1]); polin:=CC[0,N];
for i:=N-1 downto 0 do begin polin:=polin*wp; polinom_z:=CC[N-i,i]; for j:=N-i-1
downto 0 do polinom_z:=polinom_z*zp+CC[j,i]; polin:=polin+polinom_z; end;
 Result:=polin; end;
procedure Utochnenie; var i,j:word; begin Two_dim_Newton(U, N_G, C);
for i:=0 to N_G-1 do for j:=0 to N_G-1 do CX[i,j]:=C[i+1,j]*(i+1)/hhx;
CX[-1,0]:=x[0]; CX[0,-1]:=t[0]; CX[-2,0]:=hhx; CX[0,-2]:=hht; CX[-1,-1]:=N_G-1;
for i:=0 to N_G-1 do for j:=0 to N_G-1 do
Fi[i,j]:=f(x[i],t[j])-a(x[i],t[j])*Polinom(x[i],t[j],CX);
 Two_dim_Newton(Fi, N_G-1, CT); for i:=0 to N_G-1 do
for j:=0 to N_G-1 do C1[i,j]:=CT[i,j]/(j+1); C1[-1,0]:=x[0]; C1[0,-1]:=t[0];
C1[-2,0]:=hhx; C1[0,-2]:=hht; C1[-1,-1]:=N_G-1; end;
procedure Nachusl; var i,j:shortint; begin case OprPol of 0:begin if podT=0 then
 for j:=0 to N_G do for i:=0 to N_G-j do u[i,j]:=IC(x[i]) else
begin for i:=0 to N_G do if x[i]<x1 then U0[i]:=Polinom(x[i],t[0],C_last_L) else
if x[i]<x2 then U0[i]:=Polinom(x[i],t[0],C_last_C) else
U0[i]:=Polinom(x[i],t[0],C_last_R); for i:=0 to N_G do u[i,0]:=U0[i]; for j:= 1
to N_G do for i:= 0 to N_G-j do U[i,j]:= Polinom(x[i],t[j],C_last_C) end;
if Nsloya=0 then begin U0[N_G]:=BC(t[0]);u[N_G,0]:=U0[N_G];end;
if Nsloya=1 then begin U0[0]:=BC(t[0]);u[0,0]:=U0[0];end; end;
-1:begin if podT=0 then for j:=0 to N_G do for i:=0 to N_G-j do u[i,j]:=IC(x[i])
else begin for i:=0 to N_G do if x[i]<x1 then begin
u[i,0]:=Polinom(x[i],t[0],C_last_l);U0L[i]:=u[i,0] end else begin
u[i,0]:=Polinom(x[i],t[0],C_last_C);U0L[i]:=u[i,0] end; for j:=1 to N_G do
for i:=0 to N_G-j do u[i,j]:=u[i,0]; end;
if Nsloya=1 then begin U0L[kl]:=BC(t[0]);u[kl,0]:=U0L[kl];end; end;
1:begin if podT=0 then begin for j:=0 to N_G do for i:=0 to N_G-j do
u[i,j]:=IC(x[i]); end else begin for i:=0 to N_G do for j:=0 to N_G-i do
u[i,j]:=Polinom(x[i],t[j],C_last_R); for i:=0 to N_G do if x[i]<x2 then begin
u[i,0]:= Polinom(x[i],t[0],C_last_C);U0R[i]:=u[i,0] end else begin
u[i,0]:= Polinom(x[i],t[0],C_last_R);U0R[i]:=u[i,0] end; end;
if Nsloya=0 then for i:=0 to N_G do if abs(x[i]-a_Gl)<1e-17 then begin
u[i,0]:= BC(t[0]); U0R[i]:=u[i,0] end; end; end; end;
procedure IterUt(U00:Mass1;var C_last:Mass2); var i,j:shortint;iter:word;
begin for iter:= 1 to K_iter do begin Utochnenie; for i:=0 to N_G do for j:=0 to
N_G do if podT=0 then U[i,j]:= Polinom(x[i],t[j],C1)*(t[j]-t[0])+IC(x[i]) else
U[i,j]:= Polinom(x[i],t[j],C1)*(t[j]-t[0])+U00[i]; end; C_last:=C;
C_last[-1,0]:=x[0]; C_last[0,-1]:=t[0]; C_last[-2,0]:=hhx;  C_last[0,-2]:=hht;
 C_last[-1,-1]:=N_G; end;
procedure NachuslNewLevo; var i,j:shortint; begin dec(Nsloya); if podT=0 then
for j:=0 to N_G do for i:=0 to N_G-j do u[i,j]:=IC(x[i]) else begin
for i:=kl to N_G do for j:=1 to N_G-i do u[i,j]:=Polinom(x[i],t[j],C_last_C);
for i:=0 to kl-1 do for j:=0 to N_G-i do begin podtGl:=trunc(t[j]/line);
if (t[j]<>0) and (frac(t[j]/line)=0) then podtGl:=podtGl-1;
x1_t:=MasGl[Nsloya,podTGl,1][-3,0]; x2_t:=MasGl[Nsloya,podTGl,1][-4,0];
if x[i]<x1_t then u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,1]) else if
x[i]<x2_t then u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,2]) else
u[i,j]:=Polinom(x[i],t[j],MasGl[Nsloya,podTGl,3]) end; for i:=0 to N_G do
 if   x[i]<x1   then   begin   podtGl:=trunc(t[0]/line);   if   (t[0]<>0)   and
(frac(t[0]/line)=0)  then  podtGl:=podtGl-1;  x1_t:=MasGl[Nsloya,podTGl,1][-3,0];
x2_t:=MasGl[Nsloya,podTGl,1][-4,0]; if x[i]<x1_t then
u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,1]) else if x[i]<x2_t then
u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,2]) else
u[i,0]:=Polinom(x[i],t[0],MasGl[Nsloya,podTGl,3]); U0L[i]:=u[i,0] end else begin
u[i,0]:=Polinom(x[i],t[0],C_last_C); U0L[i]:=u[i,0] end;
if Nsloya=1 then begin u[kl,0]:=BC(t[0]);U0L[kl]:=u[kl,0]end; end; inc(Nsloya);
end;
procedure solve(Nsl:integer); var i,j:shortint; begin hhx:=hx/N_G;
hht:=(d00-c00)/N_G; for i:=-N_G to N_G do x[i]:=a0 + i*hhx; for j:=1 to N_G do
t[j]:= t[0] + j*hht; OprPol:=0; Nachusl; IterUt(U0,C_last_C);
for i:=kl to N_G do xx[i]:= x[i-kl];
```

```
for i:=1  to kl do xx[kl-i]:=x[0]-i*hhx; for i:=0 to N_G do
x[i]:= xx[i]; for i:=-1 downto -N_G do  x[i]:= x[0]-abs(i)*hhx;
if (Nsl=-Ksl div 2) then begin OprPol:=-1; Nachusl end else NachuslNewLevo;
IterUt(U0L,C_last_L); for i:=0 to N_G do x[i]:= a0 + i*hhx;
for i:=0 to N_G-kr do xx[i]:= x[i+kr];
for i:=0 to kr-1 do xx[N_G-i]:= x[N_G]+(kr-i)*hhx; for i:=0 to N_G do
x[i]:= xx[i]; OprPol:=1; Nachusl; IterUt(U0R,C_last_R); for i:=0 to N_G do
x[i]:= a0 + i*hhx; end;
procedure RD; var Dteck:extended;
begin x[0]:=a0; t[0]:=c0; podT:=0; c00:=c0; d00:=c00+ht; Dteck:=d_Gl+ht*(Ksl-
Nsloya); while (t[0]+line-Dteck) <=1e-15 do begin solve(Nsloya); x1:=x[sdvigl];
x2:=x[N_G-sdvigr]; C_last_L[-3,0]:=x1; C_last_L[-4,0]:=x2;
MasPolT[podT,1]:=C_last_L; MasPolT[podT,2]:=C_last_C; MasPolT[podT,3]:=C_last_R;
c00:=c00+hht; d00:=c00+ht; t[0]:=t[0]+line; inc(podT); end;
MasGl[Nsloya]:=MasPolT; end;
procedure outputGl; var Nsl,j,i:shortint; err,xpr,tpr,P:extended;
begin for j:=1 to outputGl_t do begin tpr:=c_Gl+j*(d_Gl-c_Gl)/outputGl_t;
podtGl:=trunc((tpr-c_Gl)/line); if (tpr<>c_Gl) and (abs(frac(tpr/line))<1e-18)
then podtGl:=podtGl-1; for i:=0 to outputGl_x do begin
 xpr:=a_Gl+i*(b_Gl-a_Gl)/outputGl_x; Nsl:=trunc((xpr-a_Gl)/hx)+1; if (xpr<>a_Gl)
and (abs(frac(xpr/hx))<1e-18) then Nsl:=Nsl-1; x1:=MasGl[Nsl,podTGl,1][-3,0];
x2:=MasGl[Nsl,podTGl,1][-4,0]; if xpr<x1 then begin
P:=Polinom(xpr,tpr,MasGl[Nsl,podTGl,1]);
err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,1])-solution(xpr,tpr)) end else
 if xpr<x2 then begin P:=Polinom(xpr,tpr,MasGl[Nsl,podTGl,2]);
err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,2])-solution(xpr,tpr)) end else begin
P:=Polinom(xpr,tpr,MasGl[Nsl,podTGl,3]);
err:=abs(Polinom(xpr,tpr,MasGl[Nsl,podTGl,3])-solution(xpr,tpr)); end;
writeln(' t=',tpr:7:4,' x=',xpr:7:4,' ',P,' ',err) end; readln; end;end;
procedure RD_GL; begin Ksl:=trunc(exp(kk*ln(2))); a0:=a_Gl; c0:=c_Gl;
Nsloya:=-Ksl div 2; hx:=(b_Gl-a_Gl)/Ksl; ht:=hx/2; line:=ht/10;
a0:=a_Gl-(abs(Nsloya)+1)*hx; repeat RD; inc(Nsloya); a0:=a0+hx;
until Nsloya>Ksl; end; begin preparation;
writeln('The calculation process is in progress.'); writeln('Please wait...');
tnach:=Gettime; RD_GL; tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln;
writeln('Calculation   time   ',Hour,':',Minut,':',   Sec,':',MSec);   outputGl;
Dispose(DD_G);Dispose(MasGl); end.
```

Результат работы программы:

```
t= 0.1429 x= 0.0000  1.15356499489511E+0000  0.00000000000000E+0000
t= 0.1429 x= 0.1429  1.33071219744735E+0000  0.00000000000000E+0000
t= 0.1429 x= 0.2857  1.53506300925521E+0000  2.16840434497101E-0019
t= 0.1429 x= 0.4286  1.77079495243515E+0000  4.33680868994202E-0019
t= 0.1429 x= 0.5714  2.04272707026614E+0000  0.00000000000000E+0000
t= 0.1429 x= 0.7143  2.35641844238366E+0000  4.33680868994202E-0019
t= 0.1429 x= 0.8571  2.71828182845905E+0000  6.50521303491303E-0019
t= 0.1429 x= 1.0000  3.13571476356982E+0000  2.16840434497101E-0019

t= 0.2857 x= 0.0000  1.33071219744735E+0000  0.00000000000000E+0000
t= 0.2857 x= 0.1429  1.53506300925521E+0000  5.42101086242752E-0019
t= 0.2857 x= 0.2857  1.77079495243515E+0000  1.30104260698260E-0018
t= 0.2857 x= 0.4286  2.04272707026614E+0000  2.16840434497101E-0019
t= 0.2857 x= 0.5714  2.35641844238366E+0000  8.67361737988404E-0019
t= 0.2857 x= 0.7143  2.71828182845905E+0000  1.95156391047391E-0018
t= 0.2857 x= 0.8571  3.13571476356982E+0000  2.16840434497101E-0019
t= 0.2857 x= 1.0000  3.61725078522994E+0000  0.00000000000000E+0000

t= 0.4286 x= 0.0000  1.53506300925521E+0000  0.00000000000000E+0000
t= 0.4286 x= 0.1429  1.77079495243515E+0000  3.25260651745651E-0019
t= 0.4286 x= 0.2857  2.04272707026614E+0000  2.16840434497101E-0019
t= 0.4286 x= 0.4286  2.35641844238366E+0000  3.46944695195361E-0018
t= 0.4286 x= 0.5714  2.71828182845905E+0000  3.46944695195361E-0018
t= 0.4286 x= 0.7143  3.13571476356982E+0000  1.73472347597681E-0018
t= 0.4286 x= 0.8571  3.61725078522994E+0000  4.11996825544492E-0018
```

```
t= 0.4286 x= 1.0000  4.17273388359810E+0000  2.60208521396521E-0018

t= 0.5714 x= 0.0000  1.77079495243515E+0000  1.40946282423115E-0018
t= 0.5714 x= 0.1429  2.04272707026614E+0000  1.51788304147970E-0018
t= 0.5714 x= 0.2857  2.35641844238366E+0000  0.00000000000000E+0000
t= 0.5714 x= 0.4286  2.71828182845905E+0000  6.50521303491303E-0019
t= 0.5714 x= 0.5714  3.13571476356982E+0000  3.03576608295941E-0018
t= 0.5714 x= 0.7143  3.61725078522994E+0000  6.50521303491303E-0019
t= 0.5714 x= 0.8571  4.17273388359810E+0000  2.60208521396521E-0018
t= 0.5714 x= 1.0000  4.81351974113148E+0000  3.90312782094782E-0018

t= 0.7143 x= 0.0000  2.04272707026614E+0000  1.08420217248550E-0018
t= 0.7143 x= 0.1429  2.35641844238366E+0000  2.16840434497101E-0019
t= 0.7143 x= 0.2857  2.71828182845905E+0000  4.55364912443912E-0018
t= 0.7143 x= 0.4286  3.13571476356982E+0000  2.16840434497101E-0019
t= 0.7143 x= 0.5714  3.61725078522994E+0000  1.95156391047391E-0018
t= 0.7143 x= 0.7143  4.17273388359810E+0000  1.30104260698260E-0018
t= 0.7143 x= 0.8571  4.81351974113148E+0000  1.73472347597681E-0018
t= 0.7143 x= 1.0000  5.55270787560584E+0000  5.63785129692462E-0018

t= 0.8571 x= 0.0000  2.35641844238366E+0000  2.16840434497101E-0018
t= 0.8571 x= 0.1429  2.71828182845905E+0000  4.33680868994202E-0019
t= 0.8571 x= 0.2857  3.13571476356982E+0000  7.37257477290143E-0018
t= 0.8571 x= 0.4286  3.61725078522994E+0000  4.77048955893622E-0018
t= 0.8571 x= 0.5714  4.17273388359810E+0000  4.77048955893622E-0018
t= 0.8571 x= 0.7143  4.81351974113148E+0000  3.46944695195361E-0018
t= 0.8571 x= 0.8571  5.55270787560584E+0000  1.73472347597681E-0018
t= 0.8571 x= 1.0000  6.40540943217727E+0000  3.03576608295941E-0018

t= 1.0000 x= 0.0000  2.71828182845905E+0000  6.50521303491303E-0019
t= 1.0000 x= 0.1429  3.13571476356982E+0000  0.00000000000000E+0000
t= 1.0000 x= 0.2857  3.61725078522994E+0000  2.16840434497101E-0019
t= 1.0000 x= 0.4286  4.17273388359810E+0000  3.90312782094782E-0018
t= 1.0000 x= 0.5714  4.81351974113148E+0000  3.03576608295941E-0018
t= 1.0000 x= 0.7143  5.55270787560584E+0000  0.00000000000000E+0000
t= 1.0000 x= 0.8571  6.40540943217727E+0000  2.16840434497101E-0018
t= 1.0000 x= 1.0000  7.38905609893065E+0000  1.30104260698260E-0018

Calculation time 0:0:2:963
```

Граница абсолютной погрешности кусочно-интерполяционного приближения не превышает порядка 10^{-18} во всей области приближения $G = \{ (x,t) \mid x \in [0,1], t \in [0,1] \}$, время работы программы $2\,s\,963\,ms$, параметры метода: $n = 13$, $k = 0$, $\ell = 20$. Значения абсолютных погрешностей в проверочных точках и время работы программы в аспекте сравнения с известными разностными методами приведены в главе 4 (табл. 4.4).

При значениях параметров $k = 1$, $A_3 = -1$ задача (4.84) имеет вид $u_t + u_x = -u + 3e^{x+t}$, $u(x,0) = e^x$, $u(0,t) = e^t$, отличающийся от рассмотренного выше вида линейного уравнения переноса наличием функции $u(x,t)$ в правой части уравнения. Для построения кусочно-интерполяционного приближения решения этой задачи в процедуре `Utochnenie`, представленной непосредственно выше программы необходимо заменить фрагмент кода:

```
for i:=0 to N_G-1 do for j:=0 to N_G-1 do
```

```
Fi[i,j]:=f(x[i],t[j])-a(x[i],t[j])*Polinom(x[i],t[j],CX);
```

на следующий фрагмент:

```
for i:=0 to N_G-1 do for j:=0 to N_G-1 do
Fi[i,j]:=f(x[i],t[j])-a(x[i],t[j])*Polinom(x[i],t[j],CX)-Polinom(x[i],t[j],C);
```

и задать соответственную задаче функцию правой части `f:=3*exp(x+t)`.

После внесения указанных изменений при значениях параметров $n=13$, $k=0$, $\ell=30$ имеет место следующий результат работы программы:

```
t= 0.1429 x= 0.0000  1.15356499489511E+0000  3.25260651745651E-0019
t= 0.1429 x= 0.1429  1.33071219744735E+0000  4.33680868994202E-0019
t= 0.1429 x= 0.2857  1.53506300925521E+0000  3.25260651745651E-0019
t= 0.1429 x= 0.4286  1.77079495243515E+0000  7.58941520739853E-0019
t= 0.1429 x= 0.5714  2.04272707026614E+0000  4.33680868994202E-0019
t= 0.1429 x= 0.7143  2.35641844238366E+0000  4.33680868994202E-0019
t= 0.1429 x= 0.8571  2.71828182845905E+0000  8.67361737988404E-0019
t= 0.1429 x= 1.0000  3.13571476356982E+0000  1.95156391047391E-0018

t= 0.2857 x= 0.0000  1.33071219744735E+0000  1.08420217248550E-0018
t= 0.2857 x= 0.1429  1.53506300925521E+0000  0.00000000000000E+0000
t= 0.2857 x= 0.2857  1.77079495243515E+0000  4.33680868994202E-0019
t= 0.2857 x= 0.4286  2.04272707026614E+0000  4.33680868994202E-0019
t= 0.2857 x= 0.5714  2.35641844238366E+0000  1.08420217248550E-0018
t= 0.2857 x= 0.7143  2.71828182845905E+0000  1.30104260698260E-0018
t= 0.2857 x= 0.8571  3.13571476356982E+0000  4.33680868994202E-0019
t= 0.2857 x= 1.0000  3.61725078522994E+0000  1.73472347597681E-0018

t= 0.4286 x= 0.0000  1.53506300925521E+0000  1.40946282423115E-0018
t= 0.4286 x= 0.1429  1.77079495243515E+0000  9.75781955236954E-0019
t= 0.4286 x= 0.2857  2.04272707026614E+0000  2.16840434497101E-0018
t= 0.4286 x= 0.4286  2.35641844238366E+0000  6.50521303491303E-0019
t= 0.4286 x= 0.5714  2.71828182845905E+0000  4.33680868994202E-0019
t= 0.4286 x= 0.7143  3.13571476356982E+0000  3.90312782094782E-0018
t= 0.4286 x= 0.8571  3.61725078522994E+0000  2.60208521396521E-0018
t= 0.4286 x= 1.0000  4.17273388359810E+0000  0.00000000000000E+0000

t= 0.5714 x= 0.0000  1.77079495243515E+0000  1.40946282423115E-0018
t= 0.5714 x= 0.1429  2.04272707026614E+0000  1.30104260698260E-0018
t= 0.5714 x= 0.2857  2.35641844238366E+0000  4.33680868994202E-0019
t= 0.5714 x= 0.4286  2.71828182845905E+0000  1.95156391047391E-0018
t= 0.5714 x= 0.5714  3.13571476356982E+0000  2.60208521396521E-0018
t= 0.5714 x= 0.7143  3.61725078522994E+0000  1.08420217248550E-0018
t= 0.5714 x= 0.8571  4.17273388359810E+0000  0.00000000000000E+0000
t= 0.5714 x= 1.0000  4.81351974113148E+0000  4.33680868994202E-0019

t= 0.7143 x= 0.0000  2.04272707026614E+0000  2.16840434497101E-0019
t= 0.7143 x= 0.1429  2.35641844238366E+0000  6.50521303491303E-0019
t= 0.7143 x= 0.2857  2.71828182845905E+0000  2.38524477946811E-0018
t= 0.7143 x= 0.4286  3.13571476356982E+0000  2.38524477946811E-0018
t= 0.7143 x= 0.5714  3.61725078522994E+0000  1.30104260698260E-0018
t= 0.7143 x= 0.7143  4.17273388359810E+0000  5.20417042793042E-0018
t= 0.7143 x= 0.8571  4.81351974113148E+0000  8.23993651088983E-0018
t= 0.7143 x= 1.0000  5.55270787560584E+0000  4.77048955893622E-0018

t= 0.8571 x= 0.0000  2.35641844238366E+0000  0.00000000000000E+0000
t= 0.8571 x= 0.1429  2.71828182845905E+0000  4.11996825544492E-0018
t= 0.8571 x= 0.2857  3.13571476356982E+0000  1.73472347597681E-0018
t= 0.8571 x= 0.4286  3.61725078522994E+0000  2.60208521396521E-0018
t= 0.8571 x= 0.5714  4.17273388359810E+0000  6.93889390390723E-0018
t= 0.8571 x= 0.7143  4.81351974113148E+0000  2.60208521396521E-0018
t= 0.8571 x= 0.8571  5.55270787560584E+0000  4.33680868994202E-0018
t= 0.8571 x= 1.0000  6.40540943217727E+0000  2.16840434497101E-0018

t= 1.0000 x= 0.0000  2.71828182845905E+0000  2.16840434497101E-0019
t= 1.0000 x= 0.1429  3.13571476356982E+0000  1.30104260698260E-0018
t= 1.0000 x= 0.2857  3.61725078522994E+0000  1.51788304147970E-0018
```

```
t= 1.0000 x= 0.4286  4.17273388359810E+0000  1.73472347597681E-0018
t= 1.0000 x= 0.5714  4.81351974113148E+0000  8.67361737988404E-0019
t= 1.0000 x= 0.7143  5.55270787560584E+0000  0.00000000000000E+0000
t= 1.0000 x= 0.8571  6.40540943217727E+0000  8.67361737988404E-0019
t= 1.0000 x= 1.0000  7.38905609893065E+0000  4.33680868994202E-0019

Calculation time  0:0:7:20
```

Граница абсолютной погрешности кусочно-интерполяционного приближения, аналогично предыдущему примеру, не превышает порядка 10^{-18}, время работы программы $7\,s\,20\,ms$. Сравнение полученных результатов с результатами известных методов дано в табл. 4.4. Подстановка в (4.84) значений параметров $k=2$, $A_3=-1$ приводит к квазилинейной начально-краевой задаче вида $u_t+u_x=-u^2+2e^{x+t}+e^{2(x+t)}$, $u(x,0)=e^x$, $u(0,t)=e^t$. Для ее кусочно-интерполяционного решения в процедуре `Utochnenie` листинга программы, реализующей предложенный метод, необходимо фрагмент кода:

```
for i:=0 to N_G-1 do for j:=0 to N_G-1 do
Fi[i,j]:=f(x[i],t[j])-CoefA*Polinom(x[i],t[j],CX);
```

заменить на следующий:

```
for i:=0 to N_G-1 do for j:=0 to N_G-1 do
Fi[i,j]:=f(x[i],t[j])-CoefA*Polinom(x[i],t[j],CX)-sqr(Polinom(x[i],t[j],C));
```

и определить соответственную задаче функцию правой части `f:=2*exp(t+x)+exp(2*t+2*x)`. Результат работы программы при фиксированных значениях параметров метода $n=13$, $k=0$, $\ell=35$:

```
t= 0.1429 x= 0.0000  1.15356499489511E+0000  2.16840434497101E-0019
t= 0.1429 x= 0.1429  1.33071219744735E+0000  0.00000000000000E+0000
t= 0.1429 x= 0.2857  1.53506300925521E+0000  2.16840434497101E-0019
t= 0.1429 x= 0.4286  1.77079495243515E+0000  4.33680868994202E-0019
t= 0.1429 x= 0.5714  2.04272707026614E+0000  0.00000000000000E+0000
t= 0.1429 x= 0.7143  2.35641844238366E+0000  0.00000000000000E+0000
t= 0.1429 x= 0.8571  2.71828182845905E+0000  2.60208521396521E-0018
t= 0.1429 x= 1.0000  3.13571476356982E+0000  8.67361737988404E-0019

t= 0.2857 x= 0.0000  1.33071219744735E+0000  4.33680868994202E-0019
t= 0.2857 x= 0.1429  1.53506300925521E+0000  3.25260651745651E-0019
t= 0.2857 x= 0.2857  1.77079495243515E+0000  7.58941520739853E-0019
t= 0.2857 x= 0.4286  2.04272707026614E+0000  1.30104260698260E-0018
t= 0.2857 x= 0.5714  2.35641844238366E+0000  8.67361737988404E-0019
t= 0.2857 x= 0.7143  2.71828182845905E+0000  6.50521303491303E-0019
t= 0.2857 x= 0.8571  3.13571476356982E+0000  2.16840434497101E-0019
t= 0.2857 x= 1.0000  3.61725078522994E+0000  1.30104260698260E-0018

t= 0.4286 x= 0.0000  1.53506300925521E+0000  8.67361737988404E-0019
t= 0.4286 x= 0.1429  1.77079495243515E+0000  1.08420217248550E-0018
t= 0.4286 x= 0.2857  2.04272707026614E+0000  1.30104260698260E-0018
t= 0.4286 x= 0.4286  2.35641844238366E+0000  0.00000000000000E+0000
t= 0.4286 x= 0.5714  2.71828182845905E+0000  8.67361737988404E-0019
t= 0.4286 x= 0.7143  3.13571476356982E+0000  1.95156391047391E-0018
t= 0.4286 x= 0.8571  3.61725078522994E+0000  2.38524477946811E-0018
t= 0.4286 x= 1.0000  4.17273388359810E+0000  1.30104260698260E-0018
```

```
t= 0.5714 x= 0.0000  1.77079495243515E+0000  1.30104260698260E-0018
t= 0.5714 x= 0.1429  2.04272707026614E+0000  8.67361737988404E-0019
t= 0.5714 x= 0.2857  2.35641844238366E+0000  3.03576608295941E-0018
t= 0.5714 x= 0.4286  2.71828182845905E+0000  2.38524477946811E-0018
t= 0.5714 x= 0.5714  3.13571476356982E+0000  1.30104260698260E-0018
t= 0.5714 x= 0.7143  3.61725078522994E+0000  2.16840434497101E-0019
t= 0.5714 x= 0.8571  4.17273388359810E+0000  4.33680868994202E-0019
t= 0.5714 x= 1.0000  4.81351974113148E+0000  1.73472347597681E-0018

t= 0.7143 x= 0.0000  2.04272707026614E+0000  1.30104260698260E-0018
t= 0.7143 x= 0.1429  2.35641844238366E+0000  4.33680868994202E-0019
t= 0.7143 x= 0.2857  2.71828182845905E+0000  2.38524477946811E-0018
t= 0.7143 x= 0.4286  3.13571476356982E+0000  5.20417042793042E-0018
t= 0.7143 x= 0.5714  3.61725078522994E+0000  4.77048955893622E-0018
t= 0.7143 x= 0.7143  4.17273388359810E+0000  3.03576608295941E-0018
t= 0.7143 x= 0.8571  4.81351974113148E+0000  1.30104260698260E-0018
t= 0.7143 x= 1.0000  5.55270787560584E+0000  4.33680868994202E-0019

t= 0.8571 x= 0.0000  2.35641844238366E+0000  3.46944695195361E-0018
t= 0.8571 x= 0.1429  2.71828182845905E+0000  4.98732999343332E-0018
t= 0.8571 x= 0.2857  3.13571476356982E+0000  3.90312782094782E-0018
t= 0.8571 x= 0.4286  3.61725078522994E+0000  4.77048955893622E-0018
t= 0.8571 x= 0.5714  4.17273388359810E+0000  7.80625564189563E-0018
t= 0.8571 x= 0.7143  4.81351974113148E+0000  2.16840434497101E-0018
t= 0.8571 x= 0.8571  5.55270787560584E+0000  2.16840434497101E-0018
t= 0.8571 x= 1.0000  6.40540943217727E+0000  4.33680868994202E-0019

t= 1.0000 x= 0.0000  2.71828182845905E+0000  8.67361737988404E-0019
t= 1.0000 x= 0.1429  3.13571476356982E+0000  1.73472347597681E-0018
t= 1.0000 x= 0.2857  3.61725078522994E+0000  2.16840434497101E-0019
t= 1.0000 x= 0.4286  4.17273388359810E+0000  0.00000000000000E+0000
t= 1.0000 x= 0.5714  4.81351974113148E+0000  8.67361737988404E-0019
t= 1.0000 x= 0.7143  5.55270787560584E+0000  4.33680868994202E-0019
t= 1.0000 x= 0.8571  6.40540943217727E+0000  3.03576608295941E-0018
t= 1.0000 x= 1.0000  7.38905609893065E+0000  2.60208521396521E-0018

Calculation time 0:0:8:3
```

Граница абсолютной погрешности кусочно-интерполяционного приближения не превышает порядка 10^{-18} время работы программы $8\,s\,3\,ms$. Сравнение полученных результатов с результатами известных методов также дано в табл. 4.4. Время работы программы, реализующей предложенный метод с фиксированными параметрами, более чем на порядок меньше времени приближения этой же задачи на основе известных разностных методов, реализованных в системах компьютерной математики, при этом точность кусочно-интерполяционного приближения в рассматриваемой области $G = \{ (x, t) \mid x \in [0, 1], t \in [0, 1] \}$ выше на 9 и более десятичных порядков.

В главе 4 анализируются результаты эксперимента по моделированию процесса переноса волны на основе кусочно-интерполяционного приближения решения задачи Коши с финитными начальными условиями. Рассмотрена задача Коши

$$u_t + u_x = 0,\ u(x,0) = \begin{cases} 0, & x \in (-\infty,\ \ell_1) \cup (\ell_2,\ +\infty), \\ u_0(x), & x \in [\ell_1,\ \ell_2], \end{cases}$$

где $u_0(x) = \frac{1}{2} - \frac{1}{2}\cos\left(\frac{2\pi}{\ell_2 - \ell_1}(x - \ell_1)\right)$, при $\ell_1 = 10$, $\ell_2 = 30$. При решении данной задачи в области $G = \{(x,t) \mid x \in [0,40], t \in [0,1]\}$ программа `FPI_IVP_FOR_PDE`, приведенная в начале п. П4.2, с параметрами $n = 13$, $k = 5$, $\ell = 20$ дает следующий результат:

```
t= 0.5000 x= 0.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 1.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 2.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 3.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 4.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 5.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 6.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 7.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 8.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 0.5000 x= 9.0000  9.79382692782297E-0007  9.79382692782297E-0007
t= 0.5000 x=10.0000 -1.75832898476824E-0006  1.75832898476824E-0006
t= 0.5000 x=11.0000  6.15592889403491E-0003  9.91916037697988E-0008
t= 0.5000 x=12.0000  5.44967379537783E-0002  4.79622261265850E-0011
t= 0.5000 x=13.0000  1.46446609406724E-0001  2.29640796396008E-0015
t= 0.5000 x=14.0000  2.73004750130227E-0001  3.55076211489003E-0017
t= 0.5000 x=15.0000  4.21782767479885E-0001  1.62630325872826E-0019
t= 0.5000 x=16.0000  5.78217232520115E-0001  7.58941520739853E-0019
t= 0.5000 x=17.0000  7.26995249869773E-0001  5.96311194867027E-0019
t= 0.5000 x=18.0000  8.53553390593274E-0001  2.16840434497101E-0019
t= 0.5000 x=19.0000  9.45503262094184E-0001  9.21571846612679E-0019
t= 0.5000 x=20.0000  9.93844170297569E-0001  1.62630325872826E-0019
t= 0.5000 x=21.0000  9.93844170297569E-0001  3.25260651745651E-0019
t= 0.5000 x=22.0000  9.45503262094184E-0001  3.25260651745651E-0019
t= 0.5000 x=23.0000  8.53553390593274E-0001  8.13151629364128E-0019
t= 0.5000 x=24.0000  7.26995249869773E-0001  2.00577401909818E-0018
t= 0.5000 x=25.0000  5.78217232520115E-0001  2.27682456221956E-0018
t= 0.5000 x=26.0000  4.21782767479885E-0001  1.21972744404619E-0018
t= 0.5000 x=27.0000  2.73004750130227E-0001  1.19262238973405E-0018
t= 0.5000 x=28.0000  1.46446609406726E-0001  5.01443504774546E-0019
t= 0.5000 x=29.0000  5.44957585231233E-0002  9.79382692792229E-0007
t= 0.5000 x=30.0000  6.15758803141594E-0003  1.75832898479983E-0006
t= 0.5000 x=31.0000 -9.91916037586355E-0008  9.91916037586355E-0008
t= 0.5000 x=32.0000 -4.79622261190744E-0011  4.79622261190744E-0011
t= 0.5000 x=33.0000  2.29708010398264E-0015  2.29708010398264E-0015
t= 0.5000 x=34.0000 -3.52037893329370E-0017  3.52037893329370E-0017
t= 0.5000 x=35.0000  4.46620647855917E-0022  4.46620647855917E-0022
t= 0.5000 x=36.0000  3.31329112007869E-0027  3.31329112007869E-0027
t= 0.5000 x=37.0000 -8.22797675230973E-0031  8.22797675230973E-0031
t= 0.5000 x=38.0000 -5.50619313271536E-0038  5.50619313271536E-0038
t= 0.5000 x=39.0000 -2.46472810548493E-0039  2.46472810548493E-0039
t= 0.5000 x=40.0000  1.17989045560667E-0055  1.17989045560667E-0055

t= 1.0000 x= 0.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 1.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 2.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 3.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 4.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 5.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 6.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 7.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 8.0000  0.00000000000000E+0000  0.00000000000000E+0000
t= 1.0000 x= 9.0000 -1.91721836201956E-0008  1.91721836201956E-0008
t= 1.0000 x=10.0000  2.27284744822978E-0006  2.27284744822978E-0006
t= 1.0000 x=11.0000 -6.48039389676009E-0006  6.48039389676009E-0006
t= 1.0000 x=12.0000  2.44717421163858E-0002  2.63962581855892E-0010
```

```
 t= 1.0000 x=13.0000  9.54915028130287E-0002  5.02430094870190E-0013
 t= 1.0000 x=14.0000  2.06107373853770E-0001  6.41136178435725E-0015
 t= 1.0000 x=15.0000  3.45491502812526E-0001  7.31836466427715E-0019
 t= 1.0000 x=16.0000  5.00000000000000E-0001  7.58941520739853E-0019
 t= 1.0000 x=17.0000  6.54508497187474E-0001  7.58941520739853E-0019
 t= 1.0000 x=18.0000  7.93892626146237E-0001  3.52365706057789E-0018
 t= 1.0000 x=19.0000  9.04508497187474E-0001  2.22261445359528E-0018
 t= 1.0000 x=20.0000  9.75528258147577E-0001  1.30104260698260E-0018
 t= 1.0000 x=21.0000  1.00000000000000E+0000  0.00000000000000E+0000
 t= 1.0000 x=22.0000  9.75528258147577E-0001  1.40946282423115E-0018
 t= 1.0000 x=23.0000  9.04508497187474E-0001  3.68628738645072E-0018
 t= 1.0000 x=24.0000  7.93892626146237E-0001  2.38524477946811E-0018
 t= 1.0000 x=25.0000  6.54508497187474E-0001  1.46367293285543E-0018
 t= 1.0000 x=26.0000  5.00000000000000E-0001  2.16840434497101E-0018
 t= 1.0000 x=27.0000  3.45491502812526E-0001  2.60208521396521E-0018
 t= 1.0000 x=28.0000  2.06107373853763E-0001  3.02221355580334E-0018
 t= 1.0000 x=29.0000  9.54915219847099E-0002  1.91721836226871E-0008
 t= 1.0000 x=30.0000  2.44694690049750E-0002  2.27284744817540E-0006
 t= 1.0000 x=31.0000  6.48039389658913E-0006  6.48039389658913E-0006
 t= 1.0000 x=32.0000 -2.63962581300866E-0010  2.63962581300866E-0010
 t= 1.0000 x=33.0000 -5.02429845333396E-0013  5.02429845333396E-0013
 t= 1.0000 x=34.0000 -6.41103537631020E-0015  6.41103537631020E-0015
 t= 1.0000 x=35.0000  5.22454374616397E-0019  5.22454374616397E-0019
 t= 1.0000 x=36.0000 -1.06821251584021E-0022  1.06821251584021E-0022
 t= 1.0000 x=37.0000  2.76975665101907E-0026  2.76975665101907E-0026
 t= 1.0000 x=38.0000 -8.39783296589511E-0031  8.39783296589511E-0031
 t= 1.0000 x=39.0000  6.55659233251887E-0033  6.55659233251887E-0033
t= 1.0000 x=40.0000 -2.81541471199366E-0038  2.81541471199366E-0038
```

Для сравнения в окрестности точки с абсциссой $x = 21$, соответствующей пику волны при $t = 1$, абсолютная погрешность разностной схемы Кранка-Николсон (*Maple*) имеет порядок – 10^{-14}, граница кусочно-интерполяционного приближения в данной окрестности характеризуется порядком 10^{-18}. Помимо того, кусочная непрерывность приближения позволяет сравнительно адекватно отразить характер изменения решения в окрестности амплитуды (см. глава 4, рис. 4.2). В начале и в конце волны погрешность можно снизить, окружив эти точки узкой областью с большим количеством подобластей. Результат кусочно-интерполяционного приближения в окрестности начала волны при $t = 1$:

```
t= 1.0000 x=11.0000 -1.54910487354894E-0007  1.54910487354894E-0007
t= 1.0000 x=11.0500  6.16854627055899E-0005  1.70353588927038E-0009
t= 1.0000 x=11.1000  2.46719878237331E-0004  6.11031099421430E-0011
t= 1.0000 x=11.1500  5.55062521730547E-0004  2.71553377020146E-0012
t= 1.0000 x=11.2000  9.86635785798283E-0004  6.59358798453074E-0014
t= 1.0000 x=11.2500  1.54133313342381E-0003  1.22020964931515E-0014
t= 1.0000 x=11.3000  2.21901769845929E-0003  7.01018019674815E-0016
t= 1.0000 x=11.3500  3.01952227241010E-0003  5.59175152877031E-0017
t= 1.0000 x=11.4000  3.94264934276108E-0003  4.86620428197596E-0018
t= 1.0000 x=11.4500  4.98817114172123E-0003  1.58462923772335E-0017
t= 1.0000 x=11.5000  6.15582970243113E-0003  9.15812022571350E-0018
t= 1.0000 x=11.5500  7.44533692261302E-0003  2.34645067048386E-0017
t= 1.0000 x=11.6000  8.85637463565567E-0003  1.49518255849329E-0017
t= 1.0000 x=11.6500  1.03885946891171E-0002  3.30258146134452E-0018
t= 1.0000 x=11.7000  1.20416190306263E-0002  8.29584068540862E-0018
t= 1.0000 x=11.7500  1.38150398011617E-0002  5.98767590414065E-0018
t= 1.0000 x=11.8000  1.57084194356844E-0002  3.22177451817646E-0017
t= 1.0000 x=11.8500  1.77212907711010E-0002  2.37846851589008E-0018
t= 1.0000 x=11.9000  1.98531571615285E-0002  1.55159495278043E-0017
t= 1.0000 x=11.9500  2.21034926008349E-0002  5.68359107607636E-0018
t= 1.0000 x=12.0000  2.44717418524232E-0002  2.70711729942474E-0018
```

Так, в точке $x=11.2$ абсолютная погрешность кусочно-интерполяционного приближения составляет 6.59×10^{-14}, погрешность приближения в этой же точке по разностной схеме Кранка-Николсон – 8.29×10^{-10}. Параметры метода: $n=13$, $k=4$, $\ell=40$, $G=\{(x,t)\,|\,x\in[10,12],\ t\in[0,1]\}$.

Таким образом, представлены программные реализации кусочно-интерполяционного метода с итерационным уточнением с фиксированными значениями параметров метода, на их основе выполнен численный эксперимент по кусочно-интерполяционному приближению задачи Коши и начально-краевой задачи для линейных и квазилинейных уравнений переноса. Результаты эксперимента потверждают высокую точность (граница абсолютной погрешности порядка 10^{-18}) и кусочно-непрерывный характер полученных приближений в прямоугольной области единичной длины, а также малую временную сложность предложенного метода. Указанные качества метода позволяют, в частности, уточнить результаты численного моделирования переноса волны. Кроме того, дана программная реализация метода варьируемого кусочно-интерполяционного приближения действительных функций двух действительных переменных. Представлен пример вычисления стандартной функции с точностью порядка 10^{-19} на основе кусочной интерполяции полиномами Ньютона от двух переменных седьмого порядка, преобразованными к виду алгебраических полиномов с числовыми коэффициентами.

ПРИЛОЖЕНИЕ К ГЛАВЕ 5

П5.1. Приложение к п. 5.1.2. Результаты кусочно-интерполяционного решения системы (5.2), предложенной Дж. Хиггинсом для выражения математической модели гликолиза, обсуждаемые в главе 5, были получены по следующей программе:

```
Program FPI_GL; {$APPTYPE CONSOLE} uses SysUtils, Math;
var kiter,koutput,k_:integer;Npol:byte;tt:longint; s_rez:text;
Anach,Bkonech,velint,ynach1,ynach2:extended;
function f1(x,y1,y2:extended):extended; begin f1:=1-y1*y2 end;
function f2(x,y1,y2:extended):extended; const alfa=100;betta=10;
begin f2:=alfa*y2*(y1-(1+betta)/(y2+betta)) end;
procedure RD(y1_nach, y2_nach, A_nach, B_konech, vel_int: extended; n:byte; k,
k_iter, k_output:integer;tt_:longint); const nn=10;
type matr=array[0..nn,0..nn] of extended; vect=array[0..nn] of extended;
  matrC=array[-5..nn+1] of extended; matrAll=array[0..1200] of matrC;
var d:matr; CC1,CC2:matrC; xx:vect; kk,m:longint; hour,minut,sec,msec:word;
a0,b0,h,x,y01,y02,PogrFun1,PogrFun2,PogrFun:extended; i,pod1,pod2:integer;
Ck1,Ck2,Ck11,Ck21:matrAll; x_max2,max_y2,x_min2,min_y2,y1,y2:extended;
x_max1,max_y1,x_min1,min_y1:extended; tnach,tkonech:extended;
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr;
begin e[1,1]:=1; e[1,0]:=0; for k:=2 to n do begin e[k,0]:=-e[k-1,0]*(k-1);
for i:=1 to k-1 do e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1]
end; for k:=1 to n do for i:=0 to k do d[i,k]:=e[k,i] end;
procedure Konech_Raznoct(fy1,fy2:vect; n:byte; var dy1,dy2:matr); var i,j:byte;
begin for j:=0 to n-1 do begin dy1[1,j]:=fy1[j+1]-fy1[j];
 dy2[1,j]:=fy2[j+1]-fy2[j];end; for i:=2 to n do for j:=0 to n-i do begin
dy1[i,j]:=dy1[i-1,j+1]-dy1[i-1,j]; dy2[i,j]:=dy2[i-1,j+1]-dy2[i-1,j]; end; end;
procedure Newton(U1,U2:Vect; n:byte; var Mcoef1,Mcoef2:matrC); var dy1,dy2:matr;
b1,b2:vect; p,s1,s2:extended; j,i:byte; begin Konech_Raznoct(U1,U2,n,dy1,dy2);
p:=1; for j:=1 to n do begin p:=p*j; b1[j]:=dy1[j,0]/p; b2[j]:=dy2[j,0]/p; end;
Mcoef1[0]:=U1[0]; Mcoef2[0]:=U2[0]; for i:=1 to n do begin s1:=0;s2:=0; for j:=i
to n do begin s1:=s1+d[i,j]*b1[j]; s2:=s2+d[i,j]*b2[j]; end; Mcoef1[i]:=s1;
Mcoef2[i]:=s2; end end;
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
procedure Subinterval(k_, n, K_it: integer; a0, b0, Ynach1, Ynach2: extended;
var Ck1_,Ck2_:matrAll); var hpd, a00, b00, y01, y02, h:extended; m,pod:longint;
x:vect; j:byte; i,iter,r:integer; fy1,fy2,y1,y2:vect; C1,C2:matrC; A1,A2:matrC;
t,pp1,pp2:extended;
begin hpd:=(b0-a0)/exp(k_*ln(2)); a00:=a0;b00:=a00+hpd; y01:=Ynach1;
y02:=Ynach2; x[0]:=a0;m:=0; pod:=0; while a00<=b0-hpd/2 do begin h:=(b00-a00)/n;
for j:=1 to n do begin inc(m); x[j]:=a0+m*h end; for i:=0 to n do begin
y1[i]:=y01; y2[i]:=y02; end; fy1[0]:=f1(x[0],y01,y02);
fy2[0]:=f2(x[0],y01,y02); for iter:=1 to K_it do begin for i:=1 to n do begin
fy1[i]:=f1(x[i],y1[i],y2[i]); fy2[i]:=f2(x[i],y1[i],y2[i]); end;
 Newton(fy1,fy2,n,A1,A2);  for i:=1 to n do begin t:=(x[i]-x[0])/h;
pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1); for r:=n-1 downto 0 do begin
pp1:=pp1*t+A1[r]/(r+1);  pp2:=pp2*t+A2[r]/(r+1);  end;  y1[i]:=pp1*h*t+y1[0];
y2[i]:=pp2*h*t+y2[0]; end; end; C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h;
C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h; C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h;
C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h; for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i;
C2[i]:=A2[i-1]*h/i; end; Ck1_[pod]:=C1; Ck2_[pod]:=C2; y01:=y1[n]; y02:=y2[n];
x[0]:=x[n]; inc(pod); a00:=a00+hpd; b00:=a00+hpd end end;
begin Viet(nn,d); assign(s_rez,'result_Gl_FPI.dat'); rewrite(s_rez);
```

```
tnach:=Gettime;kk:=0; a0:=A_nach; b0:=a0+vel_int; max_y1:=0; max_y2:=0;
min_y1:=1e+10; min_y2:=1e+10; y01:=y1_nach; y02:=y2_nach;
while a0 <= B_konech-vel_int/2 do begin
Subinterval(k,n,k_iter,a0,b0,y01,y02,Ck1,Ck2);
//Вывод погрешности приближения в проверочных точках
Subinterval(k+1,n,k_iter,a0,b0,y01,y02,Ck11,Ck21); kk:=kk+1;if kk=tt_ then begin
x:= b0; pod1:=trunc(exp(k*ln(2)))-1; pod2:=trunc(exp((k+1)*ln(2)))-1;
PogrFun1:=abs(Gorner(Ck11[pod2],x)-Gorner(Ck1[pod1],x));
PogrFun2:=abs(Gorner(Ck21[pod2],x)-Gorner(Ck2[pod1],x)); writeln(x:4:1,'
',Gorner(Ck1[pod1],x),' ',PogrFun1,' ',Gorner(Ck2[pod1],x),' ',PogrFun2);
//Сохранение результатов приближения в файл для визуализации
{ for i:=0 to k_output-1 do begin x:= a0+i*Vel_int/k_output;
pod1:=trunc((x-a0)/Ck1[0,-5]); writeln(s_rez, x,' ',Gorner(Ck1[pod1],x),' ',
Gorner(Ck2[pod1],x)); end;}
kk:=0; end; pod1:=trunc(exp(k*ln(2)))-1; y01:=Gorner(Ck1[pod1],b0);
 y02:=Gorner(Ck2[pod1],b0); a0:=a0+vel_int; b0:=a0+vel_int; end;
tkonech:=Gettime; DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec);
writeln; writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec);
close(s_rez);  end;  Begin  Anach:=0;  Bkonech:=20;  ynach1:=1;  ynach2:=0.001;
velint:=0.01; Npol:=4; k_:=8; kiter:=7; tt:=20; koutput:=1000000;
RD(ynach1,ynach2,Anach,Bkonech,velint,Npol,k_,kiter,koutput,tt); readln; end.
```

Результат работы программы (в 1-й колонке – аргумент, во 2-й – приближенное значение y_1, в 3-й – абсолютная погрешность приближения y_1, в 4-й – приближенное значение y_2, в 5-й – абсолютная погрешность приближения y_2):

```
0.2  1.19984051886977E+0000 1.08420217248550E-0018 1.00005499695898E-0003 2.11758236813575E-0021
0.4  1.39701695076564E+0000 1.08420217248550E-0019 5.50547246391618E-0002 1.35525271560688E-0020
0.6  2.05515213212580E-0002 1.69406589450860E-0021 7.26022579746926E+0001 2.08166817117217E-0017
0.8  1.47580466051557E-0001 1.35525271560688E-0020 2.06727246137795E-0004 1.05879118406788E-0022
1.0  3.47580118473389E-0001 5.69206140554890E-0019 8.15423764261614E-0012 3.94430452610506E-0030
1.2  5.47580118473350E-0001 0.00000000000000E+0000 1.75604756628930E-0017 1.20370621524202E-0035
.....................................................................................
9.8  2.07104387948544E+0000 2.60208521396521E-0018 1.20199441073359E-0003 6.35274710440725E-0022
10.0 8.87788076178559E-0003 4.23516473627150E-0021 1.04989065225308E+0002 6.93889390390723E-0018
10.2 1.12123316782485E-0001 8.80914265144472E-0020 6.23473196126084E-0003 5.08219768352580E-0021
10.4 3.12115503363511E-0001 1.62630325872826E-0019 1.21076415582596E-0010 1.26217744835362E-0028
.....................................................................................
16.8 1.99420716340088E+0000 2.16840434497101E-0019 9.26016547575093E-0007 3.61891517991950E-0025
17.0 4.06887613508499E-0002 1.01643953670516E-0020 1.72937028386977E+0002 9.71445146547012E-0017
17.2 4.01791378098853E-0002 3.38813178901720E-0021 7.46004315964979E+0000 3.03576608295941E-0018
17.4 2.35253024820399E-0001 8.13151629364128E-0020 6.94032822661529E-0008 0.00000000000000E+0000
.....................................................................................
19.0 1.83525302461922E+0000 6.50521303491303E-0019 2.19937535073627E-0012 5.91645678915759E-0031
19.2 2.03525215830125E+0000 8.67361737988404E-0019 3.95563835996004E-0005 2.64697796016969E-0023
19.4 6.99281200167872E-0003 9.74087889342445E-0021 1.37980671679483E+0002 2.08166817117217E-0016
19.6 7.64916214510665E-0002 2.71050543121376E-0020 2.23358567511315E-0001 1.62630325872826E-0019
19.8 2.76298865045757E-0001 8.13151629364128E-0020 2.17003996746666E-0009 1.00974195868289E-0027
20.0 4.76298865038030E-0001 9.75781955236954E-0019 1.12325806005676E-0015 3.65926689433575E-0033
```

Значения параметров кусочно-интерполяционного метода в программной реализации заданы в соответствии с высокой степенью жесткости задачи: `velint:=0.01`, $n=4$, $k=8$, $\ell=7$. Программа с фиксированными значениями параметров выбрана с целью снижения времени приближенного решения. Время работы программы представлено в табл. 5.2 при исключении операторов вычисления значений абсолютной погрешности и вывода данных в консоль. Комментированный фрагмент кода программы `FPI_GL` позволяет сохранить

результаты приближения в файл и выполнить на этой основе визуализацию значений компонентов системы, представленную на рис. 5.5, 5.6, а также построить фазовый портрет системы (рис. П5.1).

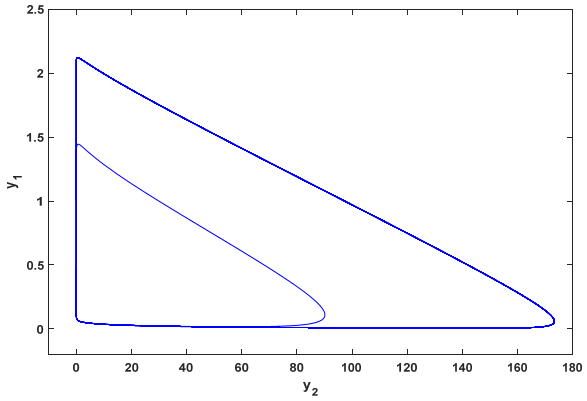

Рис. П5.1. Фазовый портретсистемы (5.3), построенный на основе кусочно-интерполяционного приближения на интервале $t \in [0, 20]$
при $\beta = 10$, $\alpha = 100$, $y_1(0) = 1$, $y_2(0) = 0.001$

Вследствие высокой точности и непрерывности построенных приближений представленная программа в отличие от известных разностных методов позволила также провести детальный анализ кинетики изменения концентраций фруктозо-6-фосфата (y_1) и фруктозодифосфата (y_2) в окрестностях точек их локальных максимумов (рис. 5.6), а также рассчитать значения амплитуд колебанийреагентов химической реакциис границейабсолютной погрешности не превышающей порядка 10^{-15}.

П5.2. Приложение к п. 5.1.3. Система Чумакова-Слинько (5.6) при значениях параметров (5.7) моделирует автоколебания, возникающие в реакции окисления молекулярного водорода на поверхности никелевого или платинового катализатора. Результаты численного моделирования на основе кусочно-интерполяционного приближения данной дифференциальной модели, обсуждаемые в главе 5, получены с использованием следующей программы:

```
Program FPI_ChumSlin; {$APPTYPE CONSOLE} uses SysUtils, Math;
var kiter,koutput,k_:integer;Npol:byte; tt:longint; s_rez:text;
Anach, Bkonech, velint, ynach1, ynach2, ynach3:extended;
const alpha=7.88;eps=0.0024;k1=1.5;k_1=0.008;k2=20;k_2=0.02;k30=100;k40=20;
mu3=2; mu4=7; mu5=-12.2732865366359;
function f1(x,y1,y2,y3:extended):extended;
```

```
begin f1:=k1*sqr(1-y1-y2)-k_1*y1*y1-2*k30*exp(-mu3*y2)*y1*y1*y2; end;
function f2(x,y1,y2,y3:extended):extended;
begin   f2:=k2*sqr(1-y1-y2)-k_2*y2*y2-k30*exp(-mu3*y2)*y1*y1*y2-k40*exp(-mu4*y2-
mu5*y3)*y2 end;
function f3(x,y1,y2,y3:extended):extended;
begin f3:=eps*(y2*(1-y3)-alpha*y3*(1-y1-y2)) end;
procedure  RD(y1_nach,y2_nach,y3_nach,A_nach,B_konech,vel_int:extended;  n:byte;
k, k_iter,k_output:integer;tt_:longint); const nn=12;
type matr=array[0..nn,0..nn] of extended; vect=array[0..nn] of extended;
matrC=array[-5..nn+1] of extended; matrAll=array[0..600] of matrC;
var d:matr; CC1,CC2,CC3:matrC; xx:vect; Ck1,Ck2,Ck3,Ck11,Ck21,Ck31:matrAll;
a0, b0, h, x, y01, y02, y03, PogrFun1, PogrFun2, PogrFun3, PogrFun:extended;
i, pod1, pod2:integer; kk, m:longint; hour,minut,sec,msec:word;
tnach, tkonech: extended; y1, y2, y3, ymax2, ymax3, ymin2, ymin3, xmax2, xmax3,
xmin2, xmin3:extended;
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr;
begin e[1,1]:=1; e[1,0]:=0; for k:=2 to n do begin e[k,0]:=-e[k-1,0]*(k-1); for
i:=1 to k-1 do e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1] end;
for k:=1 to n do for i:=0 to k do d[i,k]:=e[k,i] end;
procedure Konech_Raznoct(fy1,fy2,fy3:vect; n:byte; var dy1,dy2,dy3:matr);
var i,j:byte; begin for j:=0 to n-1 do begin dy1[1,j]:=fy1[j+1]-fy1[j];
dy2[1,j]:=fy2[j+1]-fy2[j]; dy3[1,j]:=fy3[j+1]-fy3[j]; end; for i:=2 to n do
for j:=0 to n-i do begin dy1[i,j]:=dy1[i-1,j+1]-dy1[i-1,j];
dy2[i,j]:=dy2[i-1,j+1]-dy2[i-1,j]; dy3[i,j]:=dy3[i-1,j+1]-dy3[i-1,j]; end; end;
procedure Newton(U1,U2,U3:Vect; n:byte; var Mcoef1,Mcoef2,Mcoef3:matrC);
var dy1,dy2,dy3:matr; b1,b2,b3:vect; p,s1,s2,s3:extended; j,i:byte;
begin Konech_Raznoct(U1,U2,U3,n,dy1,dy2,dy3); p:=1; for j:=1 to n do begin
p:=p*j; b1[j]:=dy1[j,0]/p; b2[j]:=dy2[j,0]/p; b3[j]:=dy3[j,0]/p;  end;
Mcoef1[0]:=U1[0]; Mcoef2[0]:=U2[0]; Mcoef3[0]:=U3[0]; for i:=1 to n do
begin s1:=0;s2:=0;s3:=0; for j:=i to n do begin s1:=s1+d[i,j]*b1[j];
s2:=s2+d[i,j]*b2[j]; s3:=s3+d[i,j]*b3[j]; end; Mcoef1[i]:=s1; Mcoef2[i]:=s2;
Mcoef3[i]:=s3; end end;
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
procedure  Subinterval(k_,  n,  K_it:  integer;  a0,  b0,  Ynach1,  Ynach2,  Ynach3:
extended; var Ck1_,Ck2_,Ck3_:matrAll);
var hpd,a00,b00,y01,y02,y03,h:extended; m,pod:longint; x:vect; j:byte;
i,iter,r:integer;  fy1,fy2,fy3,y1,y2,y3:vect;  C1,C2,C3:matrC;  A1,A2,A3:matrC;
t,pp1,pp2,pp3:extended;
begin hpd:=(b0-a0)/exp(k_*ln(2)); a00:=a0;b00:=a00+hpd; y01:=Ynach1;
y02:=Ynach2; y03:=Ynach3;x[0]:=a0;m:=0; pod:=0; while a00<=b0-hpd/2 do begin
h:=(b00-a00)/n; for j:=1 to n do begin inc(m); x[j]:=a0+m*h end; for i:=0 to n
do begin y1[i]:=y01; y2[i]:=y02; y3[i]:=y03; end; for iter:=1 to K_it do begin
for i:=0 to n do begin fy1[i]:=f1(x[i],y1[i],y2[i],y3[i]);
fy2[i]:=f2(x[i],y1[i],y2[i],y3[i]);fy3[i]:=f3(x[i],y1[i],y2[i],y3[i]); end;
Newton(fy1,fy2,fy3,n,A1,A2,A3);  for  i:=1  to  n  do  begin  t:=(x[i]-x[0])/h;
pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1); pp3:=a3[n]/(n+1); for r:=n-1 downto 0 do
begin pp1:=pp1*t+A1[r]/(r+1);pp2:=pp2*t+A2[r]/(r+1);pp3:=pp3*t+A3[r]/(r+1); end;
y1[i]:=pp1*h*t+y1[0]; y2[i]:=pp2*h*t+y2[0]; y3[i]:=pp3*h*t+y3[0]; end; end;
C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;
C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h; C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h;
C3[0]:=y3[0]; C3[-1]:=x[0]; C3[-2]:=h; C3[-3]:=n+1;C3[-4]:=k;C3[-5]:=n*h;
for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i; C2[i]:=A2[i-1]*h/i;
C3[i]:=A3[i-1]*h/i; end; Ck1_[pod]:=C1; Ck2_[pod]:=C2; Ck3_[pod]:=C3;
y01:=y1[n];  y02:=y2[n];  y03:=y3[n];  x[0]:=x[n];  inc(pod);  a00:=a00+hpd;
b00:=a00+hpd  end  end;  begin  Viet(nn,d);  assign(s_rez,'result_ChumGlin3.dat');
rewrite(s_rez);
 tnach:=Gettime; kk:=0; a0:=A_nach; b0:=a0+vel_int; y01:=y1_nach; y02:=y2_nach;
y03:=y3_nach; ymax2:=0; ymin2:=10; ymax3:=0; ymin3:=10; while a0 <= B_konech-
vel_int/2 do begin Subinterval(k,n,k_iter,a0,b0,y01,y02,y03,Ck1,Ck2,Ck3);
//Расчет и вывод погрешности приближения в проверочных точках
```

```
Subinterval(k+1,n,k_iter,a0,b0,y01,y02,y03,Ck11,Ck21,Ck31); kk:=kk+1; if kk=tt_
then begin x:= b0; pod1:=trunc(exp(k*ln(2)))-1; pod2:=trunc(exp((k+1)*ln(2)))-1;
PogrFun1:=abs(Gorner(Ck11[pod2],x)-Gorner(Ck1[pod1],x));
PogrFun2:=abs(Gorner(Ck21[pod2],x)-Gorner(Ck2[pod1],x));
PogrFun3:=abs(Gorner(Ck31[pod2],x)-Gorner(Ck3[pod1],x));
writeln(x:6:1,Format(' %1.7e',[Gorner(Ck1[pod1],x)]),Format(' %1.7e',[PogrFun1])
,Format(' %1.7e',[Gorner(Ck2[pod1],x)]),Format(' %1.7e',[PogrFun2])
,Format(' %1.7e',[Gorner(Ck3[pod1],x)]),Format(' %1.7e',[PogrFun3]));kk:=0; end;
//Сохранение результатов приближения в файл для визуализации
{for i:=0 to k_output-1 do begin x:= a0+i*Vel_int/k_output;
pod1:=trunc((x-a0)/Ck1[0,-5]); writeln(s_rez, x,' ',Gorner(Ck1[pod1],x),' ',
Gorner(Ck2[pod1],x),' ', Gorner(Ck3[pod1],x)); end;}
pod1:=trunc(exp(k*ln(2)))-1; y01:=Gorner(Ck1[pod1],b0);
y02:=Gorner(Ck2[pod1],b0); y03:=Gorner(Ck3[pod1],b0); a0:=a0+vel_int;
b0:=a0+vel_int; end; tkonech:=Gettime; DecodeTime(tnach-tkonech, Hour, Minut,
Sec, MSec); writeln; writeln('Calculation time ',Hour,':',Minut,':',
Sec,':',MSec); close(s_rez); end;
Begin Anach:=0; Bkonech:=1000; ynach1:=0.1; ynach2:=0.5; ynach3:=0; velint:=0.5;
Npol:=7; k_:=8; kiter:=10; tt:=100; koutput:=1000; RD(ynach1, ynach2, ynach3,
Anach, Bkonech, velint, Npol, k_,kiter, koutput,tt); readln; end.
```

Результат работы программы (в 1-й колонке – аргумент, во 2-й, 4-й и 6-й – приближенные значения y_1, y_2 и y_3 соответственно, в 3-й, 5-й и 7-й – соответственные значения абсолютной погрешности их приближения):

```
  50.0 1.820991E-002 4.065758E-020 8.997532E-001 2.168404E-019 1.012875E-001 2.710505E-020
 100.0 3.445373E-002 6.776264E-020 8.050176E-001 7.589415E-019 1.739308E-001 1.219727E-019
 150.0 6.522083E-002 8.131516E-020 6.145379E-001 5.421011E-019 2.014238E-001 2.710505E-020
 200.0 3.926232E-002 3.726945E-020 7.759980E-001 2.710505E-019 1.838446E-001 2.710505E-020
 250.0 7.450823E-002 4.065758E-020 5.566645E-001 4.878910E-019 1.995836E-001 2.710505E-020
 300.0 4.471182E-002 2.032879E-020 7.426516E-001 4.878910E-019 1.917323E-001 1.490778E-019
 350.0 8.582137E-002 1.355253E-020 4.881251E-001 1.084202E-019 1.947831E-001 0.000000E+000
 400.0 5.088662E-002 2.371692E-020 7.043845E-001 2.168404E-019 1.974088E-001 1.084202E-019
 450.0 1.890360E-001 4.336809E-019 1.279080E-001 8.131516E-019 1.751204E-001 1.897354E-019
 500.0 5.790719E-002 7.115077E-020 6.604558E-001 6.505213E-019 2.006866E-001 4.065758E-020
 550.0 3.485858E-002 1.016440E-020 8.026292E-001 1.626303E-019 1.748605E-001 8.131516E-020
 600.0 6.596257E-002 1.151965E-019 6.098877E-001 1.084202E-018 2.013766E-001 8.131516E-020
 650.0 3.970552E-002 6.776264E-021 7.733032E-001 2.710505E-019 1.846065E-001 5.421011E-020
 700.0 7.538774E-002 5.421011E-020 5.512415E-001 4.336809E-019 1.992896E-001 1.355253E-020
 750.0 4.521352E-002 6.776264E-021 7.395599E-001 2.168404E-019 1.923107E-001 1.355253E-019
 800.0 8.731913E-002 2.371692E-019 4.784681E-001 2.059984E-018 1.942183E-001 8.131516E-020
 850.0 5.145612E-002 1.694066E-020 7.008344E-001 1.626303E-019 1.977884E-001 2.710505E-020
 900.0 1.813650E-001 2.439455E-019 1.406818E-001 4.743385E-019 1.733862E-001 9.486769E-020
 950.0 5.855722E-002 6.776264E-021 6.563755E-001 4.878910E-019 2.008507E-001 1.761829E-019
1000.0 3.524420E-002 3.726945E-020 8.002881E-001 4.878910E-019 1.757756E-001 2.710505E-020
```

В программной реализации заданы следующие фиксированные значения параметров: `velint:=0.5`, $n=7$, $k=8$, $\ell=10$. Комментированный фрагмент кода программы `FPI_ChumSlin` позволяет сохранить результаты приближения в файл с целью визуализации значений компонентов системы и построения фазового портрета системы, которые представлены на рис. 5.7, 5.8. Из представленного результата работы программы видно, что абсолютная погрешность приближения не превышает порядок 10^{-18}, а большинстве проверочных точек харакеризуется порядками 10^{-20} – 10^{-19}. На этой основе

уточнены экстремальные значения концентраций реагентов автоколебательной реакции и, как следствие, амплитуды и период колебаний, уточненные значения в аспекте сравнения со аналогичными значениями, полученными на основе известных разностных методов, представлены в главе 5.

П5.3. Приложение к п. 5.2.1. Модель электрического равновесия автогенератора с внутренней обратной связью при использовании кубической аппроксимации вольт-амперной характеристики туннельного диода представлена в главе 5 в виде нелинейной системы (5.8). Результаты ее уточненного численного моделирования, обсуждаемые в главе, получены на интервале $t \in [0, 100]$ при значениях параметров (5.9), соответствующих релаксационному характеру автоколебаний, на основе следующей программы:

```
Program FPI_GL; {$APPTYPE CONSOLE} uses SysUtils, Math;
var kiter,k_:integer;Npol:byte;tt:longint;koutput:int64;h_int:extended;
 Anach,Bkonech,velint,ynach1,ynach2:extended;s_rez:text;
function f1(x,y1,y2:extended):extended; const G=4;
begin f1:=G*(y1-y1*y1*y1/3)-y2 end;
function f2(x,y1,y2:extended):extended; begin f2:=y1 end;
procedure RD(y1_nach, y2_nach, A_nach, B_konech, vel_int: extended; n:byte; k,
k_iter: integer; tt_:longint); const nn=10;
type matr=array[0..nn,0..nn] of extended; vect=array[0..nn] of extended;
matrC=array[-5..nn+1] of extended; matrAll=array[0..1200] of matrC;
var d:matr; CC1, CC2:matrC; xx:vect; a0, b0, h, x, y01, y02, PogrFun1, PogrFun2,
PogrFun: extended; i,pod1,pod2:integer; kk,m:longint; hour,minut,sec,msec:word;
tnach,tkonech:extended;  Ck1,Ck2,Ck11,Ck21:matrAll;  x_max1,  max1_y,  x_min1,
min1_y: extended; x_max2,max2_y,x_min2,min2_y,y1,y2:extended;
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr; begin e[1,1]:=1;
e[1,0]:=0; for k:=2 to n do begin e[k,0]:=-e[k-1,0]*(k-1); for i:=1 to k-1 do
e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1] end; for k:=1 to n
do for i:=0 to k do d[i,k]:=e[k,i] end;
procedure Konech_Raznoct(fy1,fy2:vect; n:byte; var dy1,dy2:matr); var i,j:byte;
begin for j:=0 to n-1 do begin dy1[1,j]:=fy1[j+1]-fy1[j]; dy2[1,j]:=fy2[j+1]-
fy2[j];end; for i:=2 to n do for j:=0 to n-i do begin dy1[i,j]:=dy1[i-1,j+1]-
dy1[i-1,j]; dy2[i,j]:=dy2[i-1,j+1]-dy2[i-1,j]; end; end;
procedure Newton(U1,U2:Vect; n:byte; var Mcoef1,Mcoef2:matrC); var dy1,dy2:matr;
b1,b2:vect;  p,s1,s2:extended;  j,i:byte;  begin  Konech_Raznoct(U1,U2,n,dy1,dy2);
p:=1; for j:=1 to n do begin p:=p*j; b1[j]:=dy1[j,0]/p; b2[j]:=dy2[j,0]/p; end;
 Mcoef1[0]:=U1[0]; Mcoef2[0]:=U2[0]; for i:=1 to n do begin s1:=0;s2:=0; for
j:=i to n do begin s1:=s1+d[i,j]*b1[j]; s2:=s2+d[i,j]*b2[j]; end; Mcoef1[i]:=s1;
Mcoef2[i]:=s2; end end;
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n]; for i:=n-1
downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
procedure Subinterval(k_,n, K_it: integer; a0, b0, Ynach1, Ynach2: extended; var
Ck1_,Ck2_:matrAll);
var hpd,a00,b00,y01,y02,h:extended; m,pod:longint; x:vect; j:byte;
i,iter,r:integer; fy1,fy2,y1,y2:vect; C1,C2:matrC;
 A1,A2:matrC; t,pp1,pp2:extended;
Begin hpd:=(b0-a0)/exp(k_*ln(2)); a00:=a0;b00:=a00+hpd; y01:=Ynach1;
y02:=Ynach2; x[0]:=a0;m:=0; pod:=0; while a00<=b0-hpd/2 do begin h:=(b00-a00)/n;
```

```
for j:=1 to n do begin inc(m); x[j]:=a0+m*h end; for i:=0 to n do begin
y1[i]:=y01; y2[i]:=y02; end; for iter:=1 to K_it do begin for i:=0 to n do begin
fy1[i]:=f1(x[i],y1[i],y2[i]); fy2[i]:=f2(x[i],y1[i],y2[i]); end;
Newton(fy1,fy2,n,A1,A2); for i:=1 to n do begin t:=(x[i]-x[0])/h;
pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1); for r:=n-1 downto 0 do begin
pp1:=pp1*t+A1[r]/(r+1); pp2:=pp2*t+A2[r]/(r+1); end; y1[i]:=pp1*h*t+y1[0];
y2[i]:=pp2*h*t+y2[0]; end; end; C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h;
C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h; C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h;
C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h; for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i;
C2[i]:=A2[i-1]*h/i; end; Ck1_[pod]:=C1; Ck2_[pod]:=C2; y01:=y1[n]; y02:=y2[n];
x[0]:=x[n]; inc(pod); a00:=a00+hpd; b00:=a00+hpd end end;
begin Viet(nn,d); assign(s_rez,'result_Gl_FPI.dat'); tnach:=Gettime; kk:=0;
a0:=A_nach; b0:=a0+vel_int; max1_y:=0; max2_y:=0;min1_y:=1e+10; min2_y:=1e+10;
 y01:=y1_nach; y02:=y2_nach; while a0 <= B_konech-vel_int/2 do begin
  Subinterval(k,n,k_iter,a0,b0,y01,y02,Ck1,Ck2);
//Вывод погрешности приближения в проверочных точках
Subinterval(k+1,n,k_iter,a0,b0,y01,y02,Ck11,Ck21);
  kk:=kk+1;if kk=tt_ then begin
  x:= b0; pod1:=trunc(exp(k*ln(2)))-1; pod2:=trunc(exp((k+1)*ln(2)))-1;
  PogrFun1:=abs(Gorner(Ck11[pod2],x)-Gorner(Ck1[pod1],x));
  PogrFun2:=abs(Gorner(Ck21[pod2],x)-Gorner(Ck2[pod1],x));
  writeln(x:5:1,'  ',Gorner(Ck1[pod1],x),'  ',PogrFun1,'  ',Gorner(Ck2[pod1],x),'
', PogrFun2); kk:=0; end;{Расчет экстремальных значений}
{if (a0=16) then begin for i:=0 to koutput-1 do begin x:= a0+i*h_int;
pod1:=trunc((x-a0)/Ck1[0,-5]);  y1:=Gorner(Ck1[pod1],x);  if (y1>max1_y) then
begin x_max1:=x; max1_y:=y1;end; end; end; if (a0=21)then begin for i:=0 to
koutput-1 do begin x:= a0+i*h_int; pod1:=trunc((x-a0)/Ck1[0,-5]);
y1:=Gorner(Ck1[pod1],x); if (y1<min1_y) then begin x_min1:=x; min1_y:=y1;end;
end; end; if (a0=26) then begin for i:=0 to koutput-1 do begin x:= a0+i*h_int;
pod1:=trunc((x-a0)/Ck1[0,-5]);  y1:=Gorner(Ck1[pod1],x);  if (y1>max2_y) then
begin x_max2:=x; max2_y:=y1;end; end; end; if (a0=31)then begin for i:=0 to
koutput-1 do begin x:= a0+i*h_int; pod1:=trunc((x-a0)/Ck1[0,-5]);
y1:= Gorner(Ck1[pod1],x); if (y1<min2_y) then begin x_min2:=x; min2_y:=y1;end;
end; end; }//Сохранение результатов приближения в файл для визуализации
//writeln(s_rez, x,' ',Gorner(Ck1[pod1],x),' ', Gorner(Ck2[pod1],x));
pod1:=trunc(exp(k*ln(2)))-1; y01:=Gorner(Ck1[pod1],b0);
y02:=Gorner(Ck2[pod1],b0); a0:=a0+vel_int; b0:=a0+vel_int; end;tkonech:=Gettime;
{Вывод амплитуды и периода колебаний}  {writeln('x_max1=',x_max1,'
max_y1=',max_y1); writeln('x_min1=',x_min1,' min_y1=',min_y1);
writeln('A1=',(max_y1-min_y1):20:20);writeln('x_max2=',x_max2,'max_y2=',max_y2);
writeln('x_min2=',x_min2,'    min_y2=',min_y2);    writeln('A2=',(max_y2-min_y2
):20:20); writeln('T=',(x_max2-x_max1):20:20);}
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln; writeln('Calculation
time  ',Hour,':',Minut,':', Sec,':',MSec); close(s_rez); end;
Begin Anach:=0; Bkonech:=100; ynach1:=1;ynach2:=4; velint:=1; Npol:=5; k_:=9;
kiter:=11; tt:=5; h_int:=1e-8; koutput:=100000000{1000};
RD(ynach1,ynach2,Anach,Bkonech,velint,Npol,k_,kiter,tt); readln; end.
```

Результат работы программы (в 1-й колонке – аргумент, во 2-й и 4-й – приближенные значения y_1 и y_2, в 3-й и 5-й – абсолютная погрешность их приближения):

```
  5.0 -1.085395915E+000 5.421010862E-019 -3.209744028E+000 1.517883041E-018
 10.0  1.140495450E+000 6.505213035E-019  3.096436360E+000 4.336808690E-019
 15.0 -1.190129569E+000 0.000000000E+000 -2.977811055E+000 1.084202172E-018
 20.0  1.235433025E+000 7.589415207E-019  2.854363353E+000 1.084202172E-018
 25.0 -1.277230396E+000 2.168404345E-019 -2.726489861E+000 3.035766083E-018
 30.0  1.316136115E+000 3.252606517E-019  2.594514632E+000 6.505213035E-019
 35.0 -1.352618263E+000 0.000000000E+000 -2.458707074E+000 4.336808690E-019
 40.0  1.387040470E+000 0.000000000E+000  2.319294561E+000 1.301042607E-018
 45.0 -1.419690164E+000 2.168404345E-019 -2.176471539E+000 3.469446952E-018
```

```
 50.0  1.450798074E+000 0.000000000E+000  2.030406229E+000 0.000000000E+000
 55.0 -1.480551999E+000 4.336808690E-019 -1.881245651E+000 1.951563910E-018
 60.0  1.509106698E+000 0.000000000E+000  1.729119471E+000 5.421010862E-019
 65.0 -1.536591141E+000 4.336808690E-019 -1.574142967E+000 1.084202172E-019
 70.0  1.563113899E+000 5.421010862E-019  1.416419373E+000 1.192622390E-018
 75.0 -1.588767211E+000 2.168404345E-019 -1.256041734E+000 2.168404345E-019
 80.0  1.613630105E+000 0.000000000E+000  1.093094405E+000 8.673617380E-019
 85.0 -1.637770801E+000 5.421010862E-019 -9.276542663E-001 1.626303259E-019
 90.0  1.661248609E+000 2.168404345E-019  7.597917238E-001 1.084202172E-019
 95.0 -1.684115423E+000 2.168404345E-019 -5.895715361E-001 1.626303259E-019
100.0  1.706416917E+000 1.084202172E-019  4.170535069E-001 1.084202172E-019

Calculation time  0:0:1:453
```

Параметры метода: `velint:=1`, $n=5$, $k=9$, $\ell=11$. В программе комментирован фрагмент кода, позволяющий вычислять и выводить в консоль экстремальные значения напряжения (y_1), амплитуду и период колебаний. С учетом данного фрагмента и при комментировании операторов вычисления и вывода погрешности приближения программа дает следующий результат:

```
x_max1= 6.37800705000000E+00 y1_max= 2.02296250096882E+00
x_min1= 1.14797689000000E+01 y1_min=-2.02296250096881E+00
A1=4.04592500193762808000
x_max2= 1.65815307400000E+01 y2_max = 2.02296250096881E+00
x_min2= 2.16832925900000E+01 y2_min =-2.02296250096881E+00
A2=4.04592500193762368000
T1=10.203523690000000000000
x_max2= 2.67850544300000E+01 y2_max = 2.02296250096881E+00
x_min2= 3.18868162800000E+01 y2_min =-2.02296250096881E+00
A3=4.04592500193762368000
T2=10.203523690000000000000
x_max2= 3.69885781200000E+01 y2_max = 2.02296250096881E+00
x_min2= 4.20903399700000E+01 y2_min =-2.02296250096881E+00
A4=4.04592500193762368000
T3=10.203523690000000000000
x_max2= 4.71921018100000E+01 y2_max = 2.02296250096881E+00
x_min2= 5.22938636600000E+01 y2_min =-2.02296250096881E+00
A5=4.04592500193762367000
T4=10.203523690000000000000
```

Полученные приближения значений амплитуды и периода колебаний, как показано в табл. 5.3 существенно уточняют аналогичные значения приближений, рассчитанные по известным разностным методам. Так, среди разностных методов наименьшим значением погрешности приближения амплитуды (порядка 10^{-11}) характеризуется метод Дормана-Принса 8-го порядка. При этом погрешность вычисления амплитуды на основе кусочно-интерполяционного метода характеризуется значением порядка 10^{-18}. Кроме того, сравнительно высокая точность с сочетании с непрерывностью кусочно-интерполяционного решения позволили уточнить динамику изменения значений напряжения на контуре автогенератора в окрестностях точек

локальных максимумов и точнее показать автоколебательный характер изменения значений напряжения (амплитуды оказались идентичными) в отличие от известного метода Адамса-Башворта-Мултона (рис. 5.11).

П5.4. Приложение к п. 5.2.2. Обсуждаемые в главе 5 результаты кусочно-интерполяционного приближения траектории движения космического аппарата с управлением при помощи «малой тяги», описываемого посредством дифференциальной модели (5.10) – (5.12), получены на основе следующей программы:

```
Program PP_KLA; {$APPTYPE CONSOLE} uses SysUtils, Math;
var kiter,koutput,k_:integer;Npol:byte;tt:longint; x_i:int64;
Anach,Bkonech,velint,ynach1,ynach2,ynach3,ynach4:extended;s_rez:text;
p,e,T0,h0,r0,v0,maxErr,maxa,maxb,mina,minb,rr,qq:extended; x0,hh0:extended;
function f1(x,y1,y2,y3,y4:extended):extended; begin f1:=y2; end;
function f2(x,y1,y2,y3,y4:extended):extended;
begin f2:=y4*y4/y1-y4*y4/p-(h0*y2)/(y1*y1)+(e*h0*h0*sin(y3))/(y1*y1*p) end;
function f3(x,y1,y2,y3,y4:extended):extended; begin f3:=y4/y1 end;
function f4(x,y1,y2,y3,y4:extended):extended;
begin f4:=-y2*y4/y1-(h0*y4)/(y1*y1)+h0*h0/(y1*y1*y1) end;
procedure RD(y1_nach, y2_nach, y3_nach, y4_nach, A_nach, B_konech, vel_int:
extended; n:byte;k,k_iter,k_output:integer;tt_:longint); const nn=15;
type matr=array[0..nn,0..nn] of extended; vect=array[0..nn] of extended;
matrC=array[-5..nn+1] of extended; matrAll=array[0..2048] of matrC;
martC_=^matrAll; var d:matr; CC1,CC2,CC3,CC4:matrC; xx:vect;
a0,b0,h,x,y01,y02,y03,y04,PogrFun1,PogrFun2,PogrFun3,PogrFun4,PogrFun:extended;
i,pod1,pod2:integer; kk,m:longint; hour,minut,sec,msec:word;
tnach,tkonech:extended; Ck1,Ck2,Ck3,Ck4,Ck11,Ck21,Ck31,Ck41:martC_;
max1_x,max1,min1_x,min1:extended; max2_x,max2:extended; r2,r1,q2,q1:extended;
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr; begin e[1,1]:=1;
e[1,0]:=0; for k:=2 to n do begin e[k,0]:=-e[k-1,0]*(k-1); for i:=1 to k-1 do
e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1] end; for k:=1 to n
do for i:=0 to k do d[i,k]:=e[k,i] end;
procedure Konech_Raznoct(fy1,fy2,fy3,fy4:vect; n:byte;
var dy1,dy2,dy3,dy4: matr); var i,j:byte; begin
for j:=0 to n-1 do begin dy1[1,j]:=fy1[j+1]-fy1[j];
dy2[1,j]:=fy2[j+1]-fy2[j]; dy3[1,j]:=fy3[j+1]-fy3[j]; dy4[1,j]:=fy4[j+1]-fy4[j];
end; for i:=2 to n do for j:=0 to n-i do begin
dy1[i,j]:=dy1[i-1,j+1]-dy1[i-1,j]; dy2[i,j]:=dy2[i-1,j+1]-dy2[i-1,j];
dy3[i,j]:=dy3[i-1,j+1]-dy3[i-1,j]; dy4[i,j]:=dy4[i-1,j+1]-dy4[i-1,j]; end; end;
procedure Newton(U1,U2,U3,U4:Vect; n:byte; var Mcoef1,Mcoef2,Mcoef3,Mcoef4:
matrC); var dy1,dy2,dy3,dy4:matr; b1,b2,b3,b4:vect; p,s1,s2,s3,s4:extended;
j,i:byte; begin Konech_Raznoct(U1,U2,U3,U4,n,dy1,dy2,dy3,dy4); p:=1; for j:=1 to
n do begin p:=p*j; b1[j]:=dy1[j,0]/p; b2[j]:=dy2[j,0]/p; b3[j]:=dy3[j,0]/p;
b4[j]:=dy4[j,0]/p; end; Mcoef1[0]:=U1[0]; Mcoef2[0]:=U2[0];
Mcoef3[0]:=U3[0]; Mcoef4[0]:=U4[0]; for i:=1 to n do begin
s1:=0;s2:=0;s3:=0;s4:=0; for j:=i to n do begin s1:=s1+d[i,j]*b1[j];
s2:=s2+d[i,j]*b2[j]; s3:=s3+d[i,j]*b3[j];
s4:=s4+d[i,j]*b4[j]; end; Mcoef1[i]:=s1; Mcoef2[i]:=s2; Mcoef3[i]:=s3;
Mcoef4[i]:=s4; end end;
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2];n:=trunc(Mcoef[-3]); s:=Mcoef[n]; for i:=n-1
downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
```

```
procedure    Subinterval(k_,n,K_it:integer;    a0,b0,Ynach1,Ynach2,Ynach3,Ynach4:
extended;  var  Ck1_,Ck2_,Ck3_,Ck4_:martC_);  var  hpd,a00,b00,y01,y02,y03,y04,h:
extended; m,pod:longint; x:vect; j:byte; i,iter,r:integer; C1,C2,C3,C4:matrC;
fy1,fy2,fy3,fy4,y1,y2,y3,y4:vect; A1,A2,A3,A4:matrC; t,pp1,pp2,pp3,pp4:extended;
begin     hpd:=(b0-a0)/exp(k_*ln(2));     a00:=a0;b00:=a00+hpd;     y01:=Ynach1;
y02:=Ynach2; y03:=Ynach3;y04:=Ynach4; x[0]:=a0;m:=0; pod:=0; while a00<=b0-hpd/2
do begin h:=(b00-a00)/n; for j:=1 to n do begin inc(m); x[j]:=a0+m*h end;
for i:=0 to n do begin y1[i]:=y01; y2[i]:=y02; y3[i]:=y03; y4[i]:=y04; end;
for iter:=1 to K_it do begin for i:=0 to n do begin
fy1[i]:=f1(x[i],y1[i],y2[i],y3[i],y4[i]);
fy2[i]:=f2(x[i],y1[i],y2[i],y3[i],y4[i]);
fy3[i]:=f3(x[i],y1[i],y2[i],y3[i],y4[i]);
fy4[i]:=f4(x[i],y1[i],y2[i],y3[i],y4[i]); end;
Newton(fy1,fy2,fy3,fy4,n,A1,A2,A3,A4); for i:=1 to n do begin t:=(x[i]-x[0])/h;
pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1); pp3:=a3[n]/(n+1); pp4:=a4[n]/(n+1);
for r:=n-1 downto 0 do begin pp1:=pp1*t+A1[r]/(r+1); pp2:=pp2*t+A2[r]/(r+1);
pp3:=pp3*t+A3[r]/(r+1); pp4:=pp4*t+A4[r]/(r+1); end; y1[i]:=pp1*h*t+y1[0];
y2[i]:=pp2*h*t+y2[0]; y3[i]:=pp3*h*t+y3[0]; y4[i]:=pp4*h*t+y4[0]; end; end;
C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;
C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h; C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h;
C3[0]:=y3[0]; C3[-1]:=x[0]; C3[-2]:=h; C3[-3]:=n+1;C3[-4]:=k;C3[-5]:=n*h;
C4[0]:=y4[0]; C4[-1]:=x[0]; C4[-2]:=h; C4[-3]:=n+1;C4[-4]:=k;C4[-5]:=n*h;
for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i; C2[i]:=A2[i-1]*h/i;
C3[i]:=A3[i-1]*h/i;  C4[i]:=A4[i-1]*h/i;  end;  Ck1_^[pod]:=C1;  Ck2_^[pod]:=C2;
Ck3_^[pod]:=C3; Ck4_^[pod]:=C4; y01:=y1[n]; y02:=y2[n]; y03:=y3[n]; y04:=y4[n];
x[0]:=x[n]; inc(pod); a00:=a00+hpd; b00:=a00+hpd end end;
begin Viet(nn,d); New(Ck1); New(Ck2); New(Ck3); New(Ck4); New(Ck11); New(Ck21);
New(Ck31); New(Ck41); assign(s_rez,'result_KLA_FPI.dat'); rewrite(s_rez);
 tnach:=Gettime;kk:=0;  a0:=A_nach;  b0:=a0+vel_int;  y01:=y1_nach;  y02:=y2_nach;
y03:=y3_nach; y04:=y4_nach; min1:=1e+10;min1_x:=1e+7; min2:=1e+10; min2_x:=1e+7;
max1:=0;max1_x:=0;max2:=0; max2_x:=0; while a0 <= B_konech-vel_int/2 do begin
Subinterval(k,n,k_iter,a0,b0,y01,y02,y03,y04,Ck1, Ck2, Ck3, Ck4);
//Вычисление и вывод погрешности приближения
Subinterval(k+1,n,k_iter,a0,b0,y01,y02,y03,y04,Ck11,Ck21,Ck31,Ck41); kk:=kk+1;
if kk=tt_ then begin x:= b0; pod1:=trunc(exp(k*ln(2)))-1;
pod2:=trunc(exp((k+1)*ln(2)))-1; PogrFun1:= abs(Gorner(Ck11^[pod2],x)-
Gorner(Ck1^[pod1],x)); PogrFun2:= abs(Gorner(Ck21^[pod2],x)-
Gorner(Ck2^[pod1],x));PogrFun3:=abs(Gorner(Ck31^[pod2],x)-Gorner(Ck3^[pod1],x));
PogrFun4:=abs(Gorner(Ck41^[pod2],x)-Gorner(Ck4^[pod1],x));       writeln(Format(
'%1.3e',[x]), Format(' %1.5e',[Gorner(Ck1[pod1],x)]),Format(' %1.2e',[PogrFun1])
,Format('   %1.5e',[Gorner(Ck2[pod1],x)]),Format('   %1.2e',[PogrFun2]),Format('
%1.5e',[Gorner(Ck3[pod1],x)]),Format(' %1.2e',[PogrFun3]),Format('
%1.5e',[Gorner(Ck4[pod1],x)]),Format(' %1.2e',[PogrFun4])); kk:=0; end;
{Сохранение результатов приближения в файл для визуализации}
{for i:=0 to k_output-1 do begin x:= a0+i*Vel_int/k_output;
pod1:=trunc((x-a0)/Ck1[0,-5]);
rr:=Gorner(Ck1^[pod1],x)*cos(Gorner(Ck3^[pod1],x));
qq:=Gorner(Ck1^[pod1],x)*sin(Gorner(Ck3^[pod1],x));
writeln(s_rez,x,' ',rr,' ',qq); end;}
//Расчет параметров устойчивой орбиты
{hh0:=1e-6; if a0=1154000 then begin x0:=1154657;x:=x0;x_i:=1; while x<1154658
do begin pod1:=trunc((x-a0)/Ck1[0,-5]); rr:=Gorner(Ck1^[pod1],x)*
cos(Gorner(Ck3^[pod1],x)); if (rr>max1) then begin max1_x:=x; max1:=rr;end;
x:=x0+x_i*hh0; inc(x_i); end;end; if (a0=1197000) then begin x0:=1197857; x:=x0;
x_i:=1; while x<1197858 do begin pod1:=trunc((x-a0)/Ck1[0,-5]);
rr:=Gorner(Ck1^[pod1],x)*cos(Gorner(Ck3^[pod1],x));if (rr<min1) then begin
min1_x:=x; min1:=rr;end; x:=x0+x_i*hh0; inc(x_i); end; end; if (a0=1241000)then
begin  x0:=1241057;x:=x0;x_i:=1;  while  x<1241058  do  begin  pod1:=trunc((x-
a0)/Ck1[0,-5]); rr:=Gorner(Ck1^[pod1],x)*cos(Gorner(Ck3^[pod1],x)); if (rr>max2)
then begin max2_x:=x; max2:=rr;end; x:=x0+x_i*hh0; inc(x_i); end; end;}
pod1:=trunc(exp(k*ln(2)))-1; y01:=Gorner(Ck1^[pod1],b0);
y02:= Gorner(Ck2^[pod1],b0); y03:=Gorner(Ck3^[pod1],b0);
```

```
y04:=Gorner(Ck4^[pod1],b0); a0:=a0+vel_int; b0:=a0+vel_int; end;
{Вывод параметров орбиты}{writeln('min1_x=',min1_x:20:20,' min1=',min1);
writeln('max1_x=',max1_x:20:20,' max1=',max1); writeln('A=',(max1-min1):20:20);
writeln('max2_x=',max2_x:20:20,'       max2=',max2);       writeln('T=',(max2_x-
max1_x):20:20);} Dispose(Ck1); Dispose(Ck2); Dispose(Ck3); Dispose(Ck4);
Dispose(Ck11); Dispose(Ck21);Dispose(Ck31);Dispose(Ck41); tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln; writeln('Calculation
time  ',Hour,':',Minut,':', Sec,':',MSec); close(s_rez); end;
Begin p:=36000; e:=0.5; T0:=24*3600; h0:=2*Pi*p*p/((1-e*e)*sqrt(1-e*e)*T0);
 r0:=p/(1+e);  v0:=e*h0/p;  Anach:=0;  Bkonech:=2e+6;  ynach1:=r0;  ynach2:=v0;
ynach3:=0; ynach4:=0; velint:=1000; Npol:=11; k_:=2; kiter:=10; tt:=50{3000};
koutput:=1000; RD(ynach1, ynach2, ynach3, ynach4,Anach,Bkonech, velint, Npol,
k_, kiter, koutput, tt); readln; end.
```

Результат работы программы (в 1-й колонке – аргумент, во 2-й, 4-й, 6-й и 8-й – приближенные значения y_1, y_2, y_3 и y_4 соответственно, в 3-й, 5-й, 7-й, и 9-й – соответственные значения абсолютной погрешности их приближения):

```
5.00E+004 7.6154E+004 7.1E-015  4.4757E-001 2.7E-020 2.5590E+000 0.0E+000 1.8495E+000 1.1E-019
1.00E+005 5.0908E+004 0.0E+000 -1.6668E+000 1.1E-019 3.9932E+000 2.2E-019 2.8309E+000 2.2E-019
1.50E+005 6.9609E+004 7.1E-015  5.8704E-001 5.4E-020 9.1208E+000 8.7E-019 2.0845E+000 0.0E+000
2.00E+005 2.7937E+004 0.0E+000 -1.6508E+000 0.0E+000 1.1608E+001 0.0E+000 5.1939E+000 4.3E-019
2.50E+005 7.1844E+004 7.1E-015 -1.4471E-001 2.7E-020 1.5780E+001 8.7E-019 2.0197E+000 2.2E-019
3.00E+005 3.7805E+004 0.0E+000  2.0062E+000 2.2E-019 2.0516E+001 3.5E-018 3.8382E+000 2.2E-019
3.50E+005 6.4525E+004 0.0E+000 -9.4158E-001 5.4E-020 2.2477E+001 0.0E+000 2.2488E+000 0.0E+000
4.00E+005 6.0117E+004 0.0E+000  1.2029E+000 2.2E-019 2.7635E+001 3.5E-018 2.4137E+000 2.2E-019
4.50E+005 4.5738E+004 3.6E-015 -1.8235E+000 0.0E+000 2.9405E+001 1.7E-018 3.1725E+000 0.0E+000
5.00E+005 7.0746E+004 0.0E+000  3.7772E-001 2.7E-020 3.4369E+001 3.5E-018 2.0510E+000 2.2E-019
5.50E+005 2.4005E+004 1.8E-015  7.2432E-002 1.3E-018 3.7735E+001 3.5E-018 6.0447E+000 0.0E+000
6.00E+005 7.0636E+004 7.1E-015 -3.9411E-001 8.1E-020 4.1038E+001 3.5E-018 2.0542E+000 2.2E-019
6.50E+005 4.6256E+004 3.6E-015  1.8063E+000 1.1E-019 4.6013E+001 3.5E-018 3.1370E+000 4.3E-019
7.00E+005 5.9771E+004 0.0E+000 -1.2215E+000 2.2E-019 4.7775E+001 3.5E-018 2.4277E+000 0.0E+000
7.50E+005 6.4791E+004 3.6E-015  9.2387E-001 5.4E-020 5.2931E+001 3.5E-018 2.2396E+000 2.2E-019
8.00E+005 3.7232E+004 0.0E+000 -2.0109E+000 0.0E+000 5.4912E+001 3.5E-018 3.8973E+000 0.0E+000
8.50E+005 7.1856E+004 7.1E-015  1.2742E-001 1.4E-019 5.9627E+001 6.9E-018 2.0194E+000 2.2E-019
9.00E+005 2.8487E+004 1.8E-015  1.7122E+000 6.5E-019 6.3847E+001 0.0E+000 5.0936E+000 4.3E-019
9.50E+005 6.8341E+004 0.0E+000 -6.5063E-001 1.1E-019 6.6302E+001 0.0E+000 2.1232E+000 0.0E+000
1.00E+006 5.3585E+004 3.6E-015  1.5205E+000 2.2E-019 7.1402E+001 6.9E-018 2.7079E+000 0.0E+000
1.05E+006 5.3759E+004 0.0E+000 -1.5128E+000 6.5E-019 7.3106E+001 6.9E-018 2.6992E+000 2.2E-019
1.10E+006 6.8266E+004 7.1E-015  6.5742E-001 2.7E-019 7.8208E+001 1.4E-017 2.1256E+000 2.2E-019
1.15E+006 2.8685E+004 5.3E-015 -1.7335E+000 9.8E-019 8.0646E+001 6.9E-018 5.0586E+000 0.0E+000
1.20E+006 7.1870E+004 7.1E-015 -1.2094E-001 4.4E-019 8.4883E+001 1.4E-017 2.0190E+000 0.0E+000
1.25E+006 3.7001E+004 3.6E-015  2.0124E+000 2.2E-019 8.9590E+001 0.0E+000 3.9216E+000 2.2E-019
1.30E+006 6.4896E+004 3.6E-015 -9.1678E-001 5.4E-020 9.1578E+001 6.9E-018 2.2359E+000 2.2E-019
1.35E+006 5.9631E+004 0.0E+000  1.2289E+000 5.4E-019 9.6734E+001 2.1E-017 2.4334E+000 0.0E+000
1.40E+006 4.6463E+004 0.0E+000 -1.7994E+000 3.3E-019 9.8493E+001 1.4E-017 3.1230E+000 2.2E-019
1.45E+006 7.0591E+004 7.1E-015  4.0070E-001 1.1E-019 1.0347E+002 0.0E+000 2.0556E+000 2.2E-019
1.50E+006 2.4017E+004 1.8E-015 -1.3047E-001 2.8E-018 1.0675E+002 6.9E-018 6.0418E+000 0.0E+000
1.55E+006 7.0789E+004 7.1E-015 -3.7114E-001 2.7E-020 1.1014E+002 0.0E+000 2.0498E+000 0.0E+000
1.60E+006 4.5529E+004 0.0E+000  1.8303E+000 1.1E-019 1.1510E+002 6.9E-018 3.1871E+000 2.2E-019
1.65E+006 6.0254E+004 3.6E-015 -1.1955E+000 2.2E-019 1.1687E+002 6.9E-018 2.4082E+000 0.0E+000
1.70E+006 6.4417E+004 7.1E-015  9.4869E-001 5.4E-020 1.2203E+002 6.9E-018 2.2526E+000 2.2E-019
1.75E+006 3.8035E+004 3.6E-015 -2.0038E+000 2.2E-019 1.2399E+002 0.0E+000 3.8150E+000 2.2E-019
1.80E+006 7.1801E+004 1.4E-014  1.5004E-001 4.5E-019 1.2873E+002 1.4E-017 2.0209E+000 0.0E+000
1.85E+006 2.7819E+004 0.0E+000  1.6298E+000 2.1E-018 1.3289E+002 1.4E-017 5.2161E+000 8.7E-019
1.90E+006 6.8596E+004 0.0E+000 -6.2696E-001 3.8E-019 1.3540E+002 1.4E-017 2.1153E+000 0.0E+000
1.95E+006 5.2971E+004 3.6E-015  1.5472E+000 2.2E-019 1.4050E+002 0.0E+000 2.7393E+000 2.2E-019
2.00E+006 5.4358E+004 0.0E+000 -1.4861E+000 1.1E-019 1.4220E+002 0.0E+000 2.6694E+000 2.2E-019

Calculationtime  0:0:1:796
```

Таким образом, абсолютная погрешность приближения не превышает 10^{-14} на всем промежутке интегрирования. Время работы программы при исключении

операторов вычисления и вывод погрешности приближения – $531мс$, что в сравнении с временем разностного приближения по методу Дормана-Принса 8-го порядка аппроксимации ($44s\ 810ms$) меньше на восемь десятичных порядков. В представленной выше программе комментированы фрагменты кода для рассчета и сохранения в файл декартовых координат положения КА (их визуализация представлена в главе 5 на рис. 5.12), а также для рассчета и вывода в консоль параметров орбиты устойчивого движения КА с точным значением периода $T = 86400\ c$:

```
min1_x=1197857.28021500000000000000 min1=-7.20000000000000E+0004
max1_x=1154657.28022100000000000000 max1= 2.40000000000000E+0004
A=96000.00000000000010000000
max2_x=1241057.28022100000000000000 max2= 2.40000000000000E+0004
T=86400.00000000000000000000
```

Непрерывность построенного кусочно-интерполяционного приближения позволяет показать гладкость процесса изменения координаты положения КА, тем самым повысить качество численного моделирования в сравнении с дискретным приближением на основе известных разностных методов (рис. 5.13).

П5.5. Приложение к п. 5.2.3. В главе 5 исследована математическая модель (5.13), учитывающая основные возмущения, оказывающие влияние на движение КА. Результаты кусочно-интерполяционного приближения решения данной модели, обсуждаемые в главе, получены на интервале, соответствующем ≈ 12.5 виткам спутника, при начальных значениях (5.14) на основе следующей программы:

```
Program PP_KLA; {$APPTYPE CONSOLE} uses SysUtils, Math;
Const mu=3.986004418e+14; ae=6378136; J2=1082.62575e-6; J3=-2.532435e-6;J4=-
2.37089e-6; var kiter,koutput,k_:integer;Npol:byte;tt,nf:longint; Anach,
Bkonech, velint,ynach1,ynach2,ynach3,ynach4,ynach5,ynach6:extended;s_rez:text;
ynach10, ynach20,ynach30,ynach40,ynach50,ynach60:extended; p, e, T0, h0, r0, v0,
maxErr,maxa,maxb,mina,minb,rr,qq:extended; x0,hh0:extended; x_i:int64;
y1n,y2n,y3n,y4n,y5n,y6n:extended;
function f(eq:integer;x,y1,y2,y3,y4,y5,y6:extended):extended;
var r:extended; aJ21,aJ22,aJ23,aJ31,aJ32,aJ33,aJ41,aJ42,aJ43:extended;
begin r:=sqrt(y1*y1+y2*y2+y3*y3);
aJ21:=1.5*J2*mu*ae*ae/Power(r,5)*(5*y1*sqr(y3)/sqr(r)-y1);
aJ22:=1.5*J2*mu*ae*ae/Power(r,5)*(5*y2*sqr(y3)/sqr(r)-y2);
aJ23:=1.5*J2*mu*ae*ae/Power(r,5)*(5*sqr(y3)*y3/sqr(r)-3*y3);
```

```
aJ31:=2.5*J3*mu*Power(ae,3)/Power(r,7)*(7*y1*Power(y3,3)/sqr(r)-3*y1*y3);
aJ32:=2.5*J3*mu*Power(ae,3)/Power(r,7)*(7*y2*Power(y3,3)/sqr(r)-3*y2*y3);
aJ33:=2.5*J3*mu*Power(ae,3)/Power(r,7)*(7*Power(y3,4)/sqr(r)-
6*sqr(y3)+0.6*sqr(r));
aJ41:=0.25*J4*mu*Power(ae,4)/Power(r,7)*(315*y1*Power(y3,4)/(2*Power(r,4))-
105*y1*sqr(y3)/sqr(r)+15*y1/2);
aJ42:=0.25*J4*mu*Power(ae,4)/Power(r,7)*(315*y2*Power(y3,4)/(2*Power(r,4))-
105*y2*sqr(y3)/sqr(r)+15*y2/2);
aJ43:=0.25*J4*mu*Power(ae,4)/Power(r,7)*(315*Power(y3,5)/(2*Power(r,4))-
175*Power(y3,3)/sqr(r)+75*y3/2);
case eq of 1: f:=y4; 2: f:=y5; 3: f:=y6; 4: f:=-mu*y1/(r*r*r)+aJ21+aJ31+aJ41;
5: f:=-mu*y2/(r*r*r)+aJ22+aJ32+aJ42; 6: f:=-mu*y3/(r*r*r)+aJ23+aJ33+aJ43;
end; end;
procedure RD(y1_nach,y2_nach,y3_nach,y4_nach,y5_nach, y6_nach, A_nach, B_konech,
vel_int:extended; n:byte; k, k_iter, k_output:integer; tt_:longint;var y1n, y2n,
y3n, y4n, y5n, y6n:extended); const nn=15;
type matr=array[0..nn,0..nn] of extended; vect=array[0..nn] of extended;
matrC=array[-5..nn+1] of extended; matrAll=array[0..4096] of matrC;
martC_=^matrAll; var d:matr; CC1,CC2,CC3,CC4,CC5,CC6:matrC; xx:vect;
a0,b0,h,x,y01,y02,y03,y04,y05,y06:extended; tnach,tkonech:extended;
PogrFun1,PogrFun2,PogrFun3,PogrFun4,PogrFun5,PogrFun6,PogrFun:extended;
i,pod1,pod2:integer; kk,m:longint; hour,minut,sec,msec:word;
Ck1,Ck2,Ck3,Ck4,Ck5,Ck6,Ck11,Ck21,Ck31,Ck41,Ck51,Ck61:martC_;
procedure Viet(n:byte; var d:matr); var k,i:byte; e:matr; begin e[1,1]:=1;
e[1,0]:=0; for k:=2 to n do begin e[k,0]:=-e[k-1,0]*(k-1); for i:=1 to k-1 do
e[k,k-i]:=e[k-1,k-i-1]-e[k-1,k-i]*(k-1); e[k,k]:=e[k-1,k-1] end; for k:=1 to n
do for i:=0 to k do d[i,k]:=e[k,i] end;
procedure Konech_Raznoct(fy1,fy2,fy3,fy4,fy5,fy6:vect; n:byte; var dy1,dy2, dy3,
dy4, dy5, dy6:matr); var i,j:byte; begin for j:=0 to n-1 do begin
dy1[1,j]:=fy1[j+1]-fy1[j]; dy2[1,j]:=fy2[j+1]-fy2[j]; dy3[1,j]:=fy3[j+1]-fy3[j];
dy4[1,j]:=fy4[j+1]-fy4[j]; dy5[1,j]:=fy5[j+1]-fy5[j]; dy6[1,j]:=fy6[j+1]-fy6[j];
end; for i:=2 to n do for j:=0 to n-i do begin dy1[i,j]:=dy1[i-1,j+1]-dy1[i-
1,j];   dy2[i,j]:=dy2[i-1,j+1]-dy2[i-1,j];   dy3[i,j]:=dy3[i-1,j+1]-dy3[i-1,j];
dy4[i,j]:=dy4[i-1,j+1]-dy4[i-1,j]; dy5[i,j]:=dy5[i-1,j+1]-dy5[i-1,j];
dy6[i,j]:=dy6[i-1,j+1]-dy6[i-1,j]; end; end;
procedure Newton(U1,U2,U3,U4,U5,U6:Vect; n:byte; var Mcoef1,Mcoef2, Mcoef3,
Mcoef4, Mcoef5,Mcoef6:matrC); var dy1, dy2, dy3, dy4,dy5,dy6:matr; b1,b2,b3, b4,
b5, b6: vect; p,s1,s2,s3,s4,s5,s6:extended; j,i:byte;
begin Konech_Raznoct(U1,U2,U3,U4,U5,U6,n,dy1,dy2,dy3,dy4,dy5,dy6); p:=1;
for j:=1 to n do begin p:=p*j; b1[j]:=dy1[j,0]/p; b2[j]:=dy2[j,0]/p;
b3[j]:=dy3[j,0]/p; b4[j]:=dy4[j,0]/p; b5[j]:=dy5[j,0]/p; b6[j]:=dy6[j,0]/p; end;
Mcoef1[0]:=U1[0]; Mcoef2[0]:=U2[0]; Mcoef3[0]:=U3[0]; Mcoef4[0]:=U4[0];
Mcoef5[0]:=U5[0]; Mcoef6[0]:=U6[0]; for i:=1 to n do begin s1:=0; s2:=0;
s3:=0; s4:=0; s5:=0; s6:=0; for j:=i to n do begin s1:=s1+ d[i,j]*b1[j];
s2:=s2+d[i,j]*b2[j]; s3:=s3+d[i,j]*b3[j]; s4:=s4+d[i,j]*b4[j];
s5:=s5+d[i,j]*b5[j]; s6:=s6+d[i,j]*b6[j]; end; Mcoef1[i]:=s1; Mcoef2[i]:=s2;
Mcoef3[i]:=s3; Mcoef4[i]:=s4; Mcoef5[i]:=s5; Mcoef6[i]:=s6; end end;
function Gorner(Mcoef:matrC; x:extended):extended; var i,n:byte; s,t:extended;
begin t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
 for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s end;
procedure Subinterval(k_,n,K_it:integer; a0,b0,Ynach1,Ynach2,Ynach3, Ynach4,
Ynach5, Ynach6:extended; var Ck1_,Ck2_,Ck3_,Ck4_,Ck5_,Ck6_:martC_);
var hpd, a00, b00, y01, y02, y03, y04,y05,y06,h:extended; m,pod:longint; x:vect;
j:byte; i,iter,r:integer; fy1,fy2,fy3,fy4,fy5,fy6,y1,y2,y3,y4,y5,y6:vect;
C1,C2,C3,C4,C5,C6:matrC; t,pp1,pp2,pp3,pp4,pp5,pp6:extended;
A1,A2,A3,A4,A5,A6:matrC;
Begin hpd:=(b0-a0)/exp(k_*ln(2)); a00:=a0; b00:=a00+hpd; y01:=Ynach1;
y02:=Ynach2; y03:=Ynach3;y04:=Ynach4;y05:=Ynach5;y06:=Ynach6; x[0]:=a0;m:=0;
pod:=0; repeat h:=(b00-a00)/n; for j:=1 to n do begin inc(m); x[j]:=a0+m*h end;
for i:=0 to n do begin y1[i]:=y01; y2[i]:=y02; y3[i]:=y03; y4[i]:=y04;
y5[i]:=y05; y6[i]:=y06; end; for iter:=1 to K_it do begin for i:=0 to n do
begin fy1[i]:=f(1,x[i],y1[i],y2[i],y3[i],y4[i],y5[i],y6[i]);
```

```
fy2[i]:=f(2,x[i],y1[i],y2[i],y3[i],y4[i],y5[i],y6[i]);   fy3[i]:=   f(3,   x[i],
y1[i], y2[i],y3[i],y4[i],y5[i],y6[i]); fy4[i]:= f(4, x[i], y1[i], y2[i], y3[i],
y4[i], y5[i],y6[i]); fy5[i]:=f(5,x[i],y1[i],y2[i],y3[i],y4[i],y5[i],y6[i]);
fy6[i]:=f(6,x[i],y1[i],y2[i],y3[i],y4[i],y5[i],y6[i]); end;
Newton(fy1,fy2,fy3,fy4,fy5,fy6,n,A1,A2,A3,A4,A5,A6); for i:=1 to n do begin
t:=(x[i]-x[0])/h; pp1:=a1[n]/(n+1); pp2:=a2[n]/(n+1); pp3:=a3[n]/(n+1);
pp4:=a4[n]/(n+1); pp5:=a5[n]/(n+1); pp6:=a6[n]/(n+1); for r:=n-1 downto 0 do
begin  pp1:=pp1*t+A1[r]/(r+1);  pp2:=pp2*t+A2[r]/(r+1);  pp3:=pp3*t+A3[r]/(r+1);
pp4:=pp4*t+A4[r]/(r+1); pp5:=pp5*t+A5[r]/(r+1); pp6:=pp6*t+A6[r]/(r+1); end;
y1[i]:=pp1*h*t+y1[0]; y2[i]:=pp2*h*t+y2[0]; y3[i]:=pp3*h*t+y3[0];
y4[i]:=pp4*h*t+y4[0]; y5[i]:=pp5*h*t+y5[0]; y6[i]:=pp6*h*t+y6[0]; end; end;
C1[0]:=y1[0]; C1[-1]:=x[0]; C1[-2]:=h; C1[-3]:=n+1;C1[-4]:=k;C1[-5]:=n*h;
C2[0]:=y2[0]; C2[-1]:=x[0]; C2[-2]:=h; C2[-3]:=n+1;C2[-4]:=k;C2[-5]:=n*h;
C3[0]:=y3[0]; C3[-1]:=x[0]; C3[-2]:=h; C3[-3]:=n+1;C3[-4]:=k;C3[-5]:=n*h;
C4[0]:=y4[0]; C4[-1]:=x[0]; C4[-2]:=h; C4[-3]:=n+1;C4[-4]:=k;C4[-5]:=n*h;
C5[0]:=y5[0]; C5[-1]:=x[0]; C5[-2]:=h; C5[-3]:=n+1;C5[-4]:=k;C5[-5]:=n*h;
C6[0]:=y6[0]; C6[-1]:=x[0]; C6[-2]:=h; C6[-3]:=n+1;C6[-4]:=k;C6[-5]:=n*h;
for i:=1 to n+1 do begin C1[i]:=A1[i-1]*h/i; C2[i]:=A2[i-1]*h/i; C3[i]:=A3[i-
1]*h/i; C4[i]:=A4[i-1]*h/i; C5[i]:=A5[i-1]*h/i; C6[i]:=A6[i-1]*h/i; end;
Ck1_^[pod]:=C1; Ck2_^[pod]:=C2; Ck3_^[pod]:=C3; Ck4_^[pod]:=C4; Ck5_^[pod]:=C5;
Ck6_^[pod]:=C6; y01:=y1[n]; y02:=y2[n]; y03:=y3[n]; y04:=y4[n]; y05:=y5[n];
y06:=y6[n];  x[0]:=x[n];  inc(pod);  a00:=a00+hpd;  b00:=a00+hpd  until  abs(a00-
b0)<abs(hpd)/2; end; begin Viet(nn,d); New(Ck1); New(Ck2); New(Ck3); New(Ck4);
New(Ck5);  New(Ck6);  New(Ck11);  New(Ck21);  New(Ck31);  New(Ck41);  New(Ck51);
New(Ck61);  tnach:=Gettime;kk:=0;  a0:=A_nach;  b0:=a0+vel_int;  y01:=y1_nach;
y02:=y2_nach; y03:=y3_nach; y04:=y4_nach;
 y05:=y5_nach; y06:=y6_nach; repeat
Subinterval(k,n,k_iter,a0,b0,y01,y02,y03,y04,y05,y06,Ck1,Ck2,Ck3,Ck4,Ck5,Ck6);
//Вывод погрешности приближения в проверочных точках
{Subinterval(k+1,n,k_iter,a0,b0,y01,y02,y03,y04,y05,y06,Ck11,Ck21,Ck31,Ck41,Ck51
,Ck61); kk:=kk+1;if kk=tt_ then begin x:= b0; pod1:=trunc(exp(k*ln(2)))-1;
pod2:=trunc(exp((k+1)*ln(2)))-1;     PogrFun1:=abs(Gorner(Ck11^[pod2],x)-Gorner(
Ck1^[pod1], x)); PogrFun2:=abs(Gorner(Ck21^[pod2],x)-Gorner(Ck2^[pod1],x));
PogrFun3:=abs(Gorner(Ck31^[pod2],x)-Gorner(Ck3^[pod1],x));
PogrFun4:=abs(Gorner(Ck41^[pod2],x)-Gorner(Ck4^[pod1],x));
PogrFun5:=abs(Gorner(Ck51^[pod2],x)-Gorner(Ck5^[pod1],x));
PogrFun6:=abs(Gorner(Ck61^[pod2],x)-Gorner(Ck6^[pod1],x));
writeln(x,'   ',PogrFun1,'  ',PogrFun2,'  ',PogrFun3);writeln(x,'    ',PogrFun4,'
',PogrFun5,' ',PogrFun6); kk:=0; end;}
//Сохранение результатов приближения в файл для визуализации
{if Bkonech>Anach then for i:=0 to k_output-1 do begin
x:=a0+i*Vel_int/k_output; pod1:=trunc((x-a0)/Ck1[0,-5]);
writeln(s_rez,x,' ',Gorner(Ck1^[pod1],x),' ',Gorner(Ck2^[pod1],x),'
',Gorner(Ck3^[pod1],x)); end;} pod1:=trunc(exp(k*ln(2)))-1;
y01:=Gorner(Ck1^[pod1],b0); y02:=Gorner(Ck2^[pod1],b0);
y03:=Gorner(Ck3^[pod1],b0); y04:=Gorner(Ck4^[pod1],b0);
y05:=Gorner(Ck5^[pod1],b0); y06:=Gorner(Ck6^[pod1],b0); a0:=a0+vel_int;
b0:=a0+vel_int; until abs(a0-B_konech) < abs(vel_int)/2; y1n:=y01; y2n:=y02;
y3n:=y03; y4n:=y04;y5n:=y05;y6n:=y06; Dispose(Ck1);Dispose(Ck2);Dispose(Ck3);
Dispose(Ck4); Dispose(Ck5);Dispose(Ck6); Dispose(Ck11); Dispose(Ck21);
Dispose(Ck31); Dispose(Ck41);Dispose(Ck51);Dispose(Ck61); tkonech:=Gettime;
DecodeTime(tnach-tkonech, Hour, Minut, Sec, MSec); writeln;
writeln('Calculation time  ',Hour,':',Minut,':', Sec,':',MSec); end;
Begin assign(s_rez,'result_Lares_FPI.dat'); rewrite(s_rez); Anach:=0;
Bkonech:=Anach+86400; ynach10:=-2609529.0721402253; ynach20:=-7369242.879058394;
ynach30:=106.34893915203872; ynach40:=2360.4002869474456;
ynach50:=-829.8446403508121; ynach60:=6692.7567793414635;
velint:=(Bkonech-Anach)/100; Npol:=7; k_:=4; kiter:=12; tt:=10; koutput:=1000;
RD(ynach10,ynach20,ynach30,ynach40,ynach50,ynach60,Anach,Bkonech,velint,Npol,k_,
kiter,koutput,tt,y1n,y2n,y3n,y4n,y5n,y6n);
writeln(Bkonech,' ',y1n:20:20,' ',y2n:20:20,' ',y3n:20:20);
writeln(Bkonech,' ',y4n:20:20,' ',y5n:20:20,' ',y6n:20:20);
```

```
close(s_rez); Anach:=Bkonech; Bkonech:=0{10*3600+44*60+2.839776};
 ynach1:=y1n; ynach2:=y2n; ynach3:=y3n; ynach4:= y4n;  ynach5:=y5n; ynach6:=y6n;
 velint:=(Bkonech-Anach)/100;
RD(ynach1,ynach2,ynach3,ynach4,ynach5,ynach6,Anach,Bkonech,velint,Npol,k_,kiter,
koutput,tt,y1n,y2n,y3n,y4n,y5n,y6n);
 writeln(Bkonech,' ',y1n-ynach10,' ',y2n-ynach20,' ',y3n-ynach30);
writeln(Bkonech,' ',y4n-ynach40,' ',y5n-ynach50,' ',y6n-ynach60); readln; end.
```

Результат работы программы:

```
8.64000000E+0004 1963762.867451869870000 7273926.346831519550000 -2138676.257023278140000
                  -2978.543632566772020   -1072.349578631733200    -6390.303849260424400
Calculationtime 0:0:0:985
-1.15960574476048E-0011 -1.40971678774803E-0011 -2.65699129364805E-0011
-2.17603712826531E-0014 -4.56301663120939E-0014 -9.76996261670138E-0015
```

В последних двух строках представлены значения абсолютной погрешности приближения компонентов системы $y_1 \; - \; y_6$ соответственно. Точность приближения рассчитывалась путем обратного интегрирования задачи (5.13), (5.14) на интервале $t \in [t_0 + \Delta t,\, t_0]$ с отрицательным значением значением длины интервала (`velint`) и начальными данными, полученными при решении прямой задачи в конечной точке $t = t_0 + \Delta t$. В первых двух строках результата работы программы даны: значение аргумента $t = t_0 + \Delta t$ (`8.64E+0004`) и полученные при решении прямой задачи приближения компонентов системы в этой точке. Время решения прямой задачи – $985\,ms$. В главе даты результаты численного моделирования, полученные на основе известных методов, в том числе специализированных для решения задач небесной механики. Так, в сравнении с результатами моделирования на основе специализированного метода Гаусса-Эверхарта 19-го порядка уточнение координат и скорости возмущенного движения спутника выполнено более чем на два десятичных порядка. При этом время расчета кусочно-интерполяционного приближения на порядок меньше.

П5.6. Приложение к п. 5.3. Выполняется дополнительное исследование погрешности варьируемой кусочно-полиномиальной аппроксимации функций и вычисления интегралов на основе варьируемой кусочно-полиномиальной аппроксимации подынтегральной функции. Ниже выполнены видоизменения

данных в главе 2 оценок в условиях ослабленных ограничений. С целью отличия излагаемого ниже подхода от подхода, представленного в главе 2, изменим некоторые обозначения.

Функцию $u(x)$ обозначим $y = f(x)$, при этом

$$f(x) \equiv u(x) \qquad \forall x \in [\alpha, \beta], \tag{П5.1}$$

где $u(x)$ определена на $[\alpha, \beta]$ при разбиении из (2.1) и затем аппроксимируется согласно (2.2). Помимо того, расстояние между соседними узлами интерполяции на подынтервале $[x_i, x_{i+1}]$ из (2.1) обозначим h_k – в соответствии с числом подынтервалов $P = 2^k$, $k \in \{0, 1, \ldots\}$. При каждом k для полинома степени n, по построению, $h_k = x_{i\,(\ell+1)} - x_{i\ell} = \dfrac{x_{i+1} - x_i}{n}$ $\forall \ell = \overline{0, n-1}$, где $x_{i\ell}$ – равноотстоящие узлы интерполяции на подынтервале $[x_i, x_{i+1}]$ из (2.1), h_0 соответствует отсутствию разбиения на начальном отрезке $[\alpha, \beta]$.

В этих обозначениях интерполяционный полином Ньютона (2.4) запишется в виде

$$\Psi_{in}(t) = f(x_{i0}) + \sum_{j=1}^{n} \frac{\Delta^j f_{i0}}{j!} \prod_{\ell=0}^{j-1} (t - \ell) \;\; \forall \; i = 0, 1, \ldots, 2^k - 1. \tag{П5.2}$$

где $t = \dfrac{x - x_{i0}}{h_k}$, $h_k = \dfrac{\beta - \alpha}{2^k n}$. Всюду в дальнейшем, если не оговорено иное, предполагается непрерывность и однократная непрерывная дифференцируемость функции (П5.1) на $[\alpha, \beta]$ с соответствующими односторонними производными на границах отрезка.

Для удобства рассуждений полином (П5.2) целесообразно представить в эквивалентном виде:

$$\Psi_{in}(x) = f(x_{i0}) + \sum_{j=1}^{n} \frac{\Delta^j f_{i0}}{j!\,(h_k)^j} \prod_{\ell=0}^{j-1} (x - x_{i\ell}), \tag{П5.3}$$

h_k – расстояние между соседними узлами. Предполагается, что для всех полиномов (П5.3) вариация степени ограничена постоянным значением: $n \le N$, где $N = \text{const}$.

В данных предположениях и обозначениях требуется оценить разность

$$\delta = |f(x) - \Psi_{in}(x)| \quad \forall x \in [x_i, x_{i+1}] \ \forall i = \overline{0, 2^k - 1}. \tag{П5.4}$$

Применение формулы Лагранжа к функции под знаком модуля в (П5.4) в промежутке длины h_k между произвольно выбранными соседними узлами интерполяции на рассматриваемом подынтервале влечет:

$$\begin{aligned} &\frac{f(x) - \Psi_{in}(x) - (f(x_{i\ell}) - \Psi_{in}(x_{i\ell}))}{x - x_{i\ell}} = \\ &\qquad = f'(x_{i\ell} + \theta(x - x_{i\ell})) - \Psi'_{in}(x_{i\ell} + \theta(x - x_{i\ell})), \end{aligned} \tag{П5.5}$$

где $0 < \theta < 1$, ℓ – номер узла интерполяции на $[x_i, x_{i+1}]$ из (2.1), такой что $x_{i(\ell+1)} < x < x_{i\ell}$, $\ell = \overline{0, n-1}$. Отсюда

$$\begin{aligned} f(x) - \Psi_{in}(x) = f(x_{i\ell}) - \Psi_{in}(x_{i\ell}) + \\ + (f'(x_{i\ell} + \theta(x - x_{i\ell})) - \Psi'_{in}(x_{i\ell} + \theta(x - x_{i\ell}))) (x - x_{i\ell}). \end{aligned} \tag{П5.6}$$

Из (П5.6) следует, что

$$\left| f(x) - \Psi_{in}(x) \right| \le |f(x_{i\ell}) - \Psi_{in}(x_{i\ell})| + (\max_{[\alpha, \beta]} |f'(x)| + \max_{[\alpha, \beta]} |\Psi'_{in}(x)|) \, h_k, \tag{П5.7}$$

где $h_k = x_{i(\ell+1)} - x_{i\ell} = \dfrac{x_{i+1} - x_i}{n} \ \forall \ell = \overline{0, n-1}$, $\forall i = 0, 1, \ldots, 2^k - 1$.

В (П5.7) $f(x_{i\ell}) = \Psi_{in}(x_{i\ell})$ по условиям интерполяции, а по непрерывности $f'(x)$ выполняется соотношение:

$$\max_{[\alpha, \beta]} |f'(x)| = c, \ c = \text{const} \ \forall x \in [\alpha, \beta]. \tag{П5.8}$$

С целью подстановки в (П5.7) аналогичное (П5.8) соотношение требуется получить для $\Psi'_{in}(x)$ одновременно для всех подынтервалов $[x_i, x_{i+1}]$, $i = \overline{0,\, 2^k - 1}$, и всех $n \le N$, где $N = \text{const}$.

Будем предполагать $n \ge 2$. Требуемые соотношения для производных полинома $\Psi_{in}(x)$ получатся следующим образом.

Рассмотрим полином под знаком Σ в (П5.2) вместе с множителем

$$\widetilde{P}_{ij}(t) = \frac{\Delta^j f_{i0}}{j!} \prod_{\ell=0}^{j-1} (t-\ell), \;\; t = \frac{x - x_{i0}}{h_k}. \qquad \text{(П5.9)}$$

Рассмотрим также аналогичное выражение, входящее в производную от полинома Ньютона (П5.2), предварительно преобразовав ее выражение.

Имеем:

$$\Psi'_{in}(x) = \frac{1}{h_k}\, \Psi'_{in}(t), \; \Psi'_{in}(t) = \sum_{j=1}^{n} \frac{\Delta^j f_{i0}}{j!} \left(\prod_{\ell=0}^{j-1} (t-\ell) \right)'_t .$$

Отсюда и из (2.6) следует, что

$$\Psi'_{in}(t) = \sum_{j=1}^{n} \frac{\Delta^j f_{i0}}{j!} \left(\sum_{\ell=0}^{j} d_{j\ell}\, t^{\ell} \right)'_t ,$$

где согласно (2.8) с учетом $z_\ell = \ell$ получается:

$$\begin{pmatrix} d_{jj} \\ d_{j(j-1)} \\ \dots \\ d_{j0} \end{pmatrix} = \prod_{\ell=0}^{j-1} \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ -\ell & 1 & \dots & 0 & 0 \\ 0 & -\ell & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & -\ell & 1 \\ 0 & 0 & \dots & 0 & -\ell \end{pmatrix}. \qquad \text{(П5.10)}$$

С другой стороны,

$$\Psi'_{in}(t) = \sum_{j=1}^{n} \frac{\Delta^j f_{i0}}{j!} \sum_{\ell=1}^{j} \ell\, d_{j\ell}\, t^{\ell-1} .$$

Поэтому

$$\Psi'_{in}(x) = \sum_{j=1}^{n} \frac{\Delta^j f_{i0}}{j!\, h_k} \sum_{\ell=1}^{j} \ell\, d_{j\ell} \left(\frac{x - x_{i0}}{h_k} \right)^{\ell-1}. \qquad (П5.11)$$

С учетом (П5.10) все $d_{j\ell}$ ограничены следующей числовой константой:

$$\left\| \begin{pmatrix} d_{jj} \\ d_{j(j-1)} \\ \dots \\ d_{j0} \end{pmatrix} \right\| \le \prod_{\ell=0}^{j-1} \left\| \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ -\ell & 1 & \dots & 0 & 0 \\ 0 & -\ell & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & -\ell & 1 \\ 0 & 0 & \dots & 0 & -\ell \end{pmatrix} \right\|.$$

Здесь и ниже выбрана каноническая норма матрицы, согласованная с нормой вектора.

Отсюда

$$\max_{0 \le \ell \le j} |d_{j\ell}| \le \prod_{\ell=0}^{j-1} (1 + \ell) = j!. \qquad (П5.12)$$

Лемма П5.1. Полиномы $\prod_{\ell=0}^{j-1} (t - \ell) = \sum_{\ell=0}^{j} d_{j\ell}\, t^{\ell}$, $t = \frac{x - x_{i0}}{h_k}$, на всем отрезке $x \in [\alpha, \beta]$ ограничены величиной из правой части неравенства:

$$\left| \prod_{\ell=0}^{j-1} (t - \ell) \right| \le j!\, \frac{n^{j+1} - 1}{n - 1}. \qquad (П5.13)$$

Первые две производные от них соответственно ограничены величинами из правых частей неравенств:

$$\left| \left(\prod_{\ell=0}^{j-1} (t - \ell) \right)'_t \right| \le j \times j!\, \frac{n^j - 1}{n - 1} \qquad (П5.14)$$

и

$$\left| \left(\prod_{\ell=0}^{j-1} (t - \ell) \right)''_t \right| \le j(j-1) \times j! \frac{n^{j-1} - 1}{n - 1}. \qquad (П5.15)$$

Доказательство. Очевидно, для любого номера подынтервала i выполняются неравенство:

$$\left| \sum_{\ell=0}^{j} d_{j\ell} t^{\ell} \right| \le \sum_{\ell=0}^{j} \left| d_{j\ell} \right| \left| \frac{x - x_{i0}}{h_k} \right|^{\ell}.$$

Тем более,

$$\left| \sum_{\ell=0}^{j} d_{j\ell} t^{\ell} \right| \le \sum_{\ell=0}^{j} \left| d_{j\ell} \right| \left| \frac{x_{in} - x_{i0}}{h_k} \right|^{\ell}.$$

Поскольку $x_{in} = x_{i0} + n h_k$, то с учетом (П5.12) получится:

$$\left| \sum_{\ell=0}^{j} d_{j\ell} t^{\ell} \right| \le j! \sum_{\ell=0}^{j} n^{\ell},$$

или,

$$\left| \prod_{\ell=0}^{j-1} (t - \ell) \right| \le j! \frac{n^{j+1} - 1}{n - 1}.$$

Неравенство (П5.13) доказано.

Для доказательства (П5.14) заметим, что

$$\left(\prod_{\ell=0}^{j-1} (t - \ell) \right)'_t = \left(\sum_{\ell=0}^{j} d_{j\ell} t^{\ell} \right)'_t,$$

поэтому

$$\left(\prod_{\ell=0}^{j-1} (t - \ell) \right)'_t = \sum_{\ell=1}^{j} \ell\, d_{j\ell} t^{\ell-1}.$$

Как и раньше,

$$\left| \sum_{\ell=1}^{j} \ell\, d_{j\ell} t^{\ell-1} \right| \le \sum_{\ell=1}^{j} \ell \left| d_{j\ell} \right| \left| \frac{x_{in} - x_{i0}}{h_k} \right|^{\ell-1},$$

и с учетом (П5.12) –

$$\left| \sum_{\ell=1}^{j} \ell\, d_{j\ell} t^{\ell-1} \right| \le j! \sum_{\ell=1}^{j} \ell\, n^{\ell-1}.$$

Отсюда

$$\left| \sum_{\ell=1}^{j} \ell\, d_{j\ell}\, t^{\ell-1} \right| \le j \times j! \sum_{\ell=1}^{j} n^{\ell-1} = j \times j!\, \frac{n^j - 1}{n-1}\,.$$

Окончательно,

$$\left| \left(\prod_{\ell=0}^{j-1} (t-\ell) \right)'_t \right| \le j \times j!\, \frac{n^j - 1}{n-1}\,.$$

Неравенство (П5.14) доказано.

Аналогично предыдущему,

$$\left(\prod_{\ell=0}^{j-1} (t-\ell) \right)''_t = \sum_{\ell=2}^{j} \ell\big(\ell-1\big) d_{j\ell}\, t^{\ell-2}\,.$$

Отсюда

$$\left| \left(\prod_{\ell=0}^{j-1} (t-\ell) \right)''_t \right| \le j\big(j-1\big) \times j! \sum_{\ell=2}^{j} n^{\ell-2}\,.$$

Окончательно,

$$\left| \left(\prod_{\ell=0}^{j-1} (t-\ell) \right)''_t \right| \le j\big(j-1\big) \times j! \frac{n^{j-1} - 1}{n-1}\,.$$

Неравенство (П5.15) и лемма доказаны.

На основании леммы П5.1 с учетом (П5.11) и (П5.14) выполняется неравенство:

$$|\Psi'_{in}(x)| \le \sum_{j=1}^{n} \left| \frac{\Delta^j f_{i0}}{h_k} \right| \times j\, \frac{n^j - 1}{n-1},$$

или,

$$|\Psi'_{in}(x)| \le \frac{1}{n-1} \sum_{j=1}^{n} \left| \frac{\Delta^j f_{i0}}{h_k} \right| \big(j\, n^j - 1 \big). \qquad \text{(П5.16)}$$

Дробь под знаком суммы в (П5.16) с разложением по определению конечных разностей преобразуется к виду:

$$\frac{\Delta^j f_{i\,0}}{h_k} = \frac{\Delta^{j-1} f_{i\,1} - \Delta^{j-1} f_{i\,0}}{h_k} = \frac{\Delta^{j-1} f(x_{i\,0} + h_k) - \Delta^{j-1} f(x_{i\,0})}{h_k}.$$

Выражение конечных разностей высшего порядка через значения функции приводит к равенству:

$$\Delta^{j-1} f(x_{i\,0} + h_k) = f(x_{i\,0} + j\,h_k) - (j-1)\,f(x_{i\,0} + (j-1)\,h_k) + \\ + \frac{(j-1)(j-2)}{2} f(x_{i\,0} + (j-2)\,h_k) - \ldots + (-1)^{j-1}\, f(x_{i\,0} + h_k),$$

аналогично,

$$\Delta^{j-1} f(x_{i\,0}) = f(x_{i\,0} + (j-1)\,h_k) - (j-1)\,f(x_{i\,0} + (j-2)\,h_k) + \\ + \frac{(j-1)(j-2)}{2} f(x_{i\,0} + (j-3)\,h_k) - \ldots + (-1)^{j-1}\, f(x_{i\,0}).$$

Почленное вычитание в обеих частях двух последних равенств влечет:

$$\Delta^{j-1} f(x_{i\,0} + h_k) - \Delta^{j-1} f(x_{i\,0}) = f(x_{i\,0} + j\,h_k) - f(x_{i\,0} + (j-1)\,h_k) - \\ - (j-1)\,(f(x_{i\,0} + (j-1)\,h_k) - f(x_{i\,0} + (j-2)\,h_k)) + \\ + \frac{(j-1)(j-2)}{2} (f(x_{i\,0} + (j-2)\,h_k) - f(x_{i\,0} + (j-3)\,h_k)) - \ldots \\ \ldots + (-1)^{j-1}\,(f(x_{i\,0} + h_k) - f(x_{i\,0})).$$

Применяя формулу Лагранжа для каждой пары уменьшаемых и вычитаемых с учетом сдвига аргумента функции на h_k, получим:

$$\Delta^{j-1} f(x_{i\,0} + h_k) - \Delta^{j-1} f(x_{i\,0}) = \\ = f'(x_{i\,0} + (j-1)\,h_k + \theta_1\,h_k)\;h_k - (j-1)\;f'(x_{i\,0} + (j-2)\,h_k + \theta_2\,h_k)\;h_k + \\ + \frac{(j-1)(j-2)}{2} f'(x_{i\,0} + (j-3)\,h_k + \theta_3\,h_k)\;h_k - \ldots + (-1)^{j-1} f'(x_{i\,0} + \theta_j\,h_k)\;h_k,$$

где $0 < \theta_\ell < 1 \;\; \forall\, \ell = \overline{1,\; j}$. Таким образом, имеет место

Лемма П5.2. В рассматриваемых условиях выполнено равенство:

$$\begin{aligned}&\Delta^{j-1} f(x_{i\,0}+h_k)-\Delta^{j-1} f(x_{i\,0})=\\&=h_k\sum_{\ell=0}^{j-1}(-1)^{\ell}\,C_{j-1}^{\ell}\,f'(x_{i\,0}+(j-\ell-1)h_k+\theta_{\ell+1}h_k).\end{aligned}\qquad(\text{П5.17})$$

Деление обеих частей (П5.17) на h_k влечет

$$\frac{\Delta^{j-1} f(x_{i\,0}+h_k)-\Delta^{j-1} f(x_{i\,0})}{h_k}=\sum_{\ell=0}^{j-1}(-1)^{\ell}\,C_{j-1}^{\ell}\,f'(x_{i\,0}+(j-\ell-1)h_k+\theta_{\ell+1}h_k).$$

Отсюда с учетом ограниченности $f'(x)$ на $[\alpha,\beta]$ согласно (П5.8) вытекает

Следствие П5.1. В условиях леммы П5.2 верно соотношение:

$$\left|\frac{\Delta^{j} f_{i\,0}}{h_k}\right|\le c\sum_{\ell=0}^{j-1}C_{j-1}^{\ell}=2^{j-1}c,\qquad(\text{П5.18})$$

где 2^{j-1} – сумма биномиальных коэффициентов, а $c=\text{const}$ из (П5.8).

Подстановка (П5.18) в (П5.16), приводит к неравенству

$$|\Psi'_{in}(x)|\le\frac{c}{n-1}\sum_{j=1}^{n}2^{j-1}\left(j\,n^{j}-1\right).$$

Отсюда

$$|\Psi'_{in}(x)|\le c\max_{1\le j\le n}\frac{j}{n-1}\sum_{j=1}^{n}2^{j-1}\left(n^{j}-1\right),$$

или,

$$|\Psi'_{in}(x)|\le c\max_{1\le j\le n}\frac{j}{n-1}\sum_{j=1}^{n}\left(2^{j-1}n^{j}-2^{j-1}\right).$$

Поскольку, с учетом $2\le n$,

$$\max_{1\le j\le n}\frac{j}{n-1}\le\frac{n}{n-1}\le 2,$$

то

$$|\Psi'_{in}(x)| \le c\left(\sum_{j=1}^{n}(2n)^j - \sum_{j=1}^{n}2^j\right).$$

Добавление начальных единичных слагаемых под оба знака суммы оценки не изменит:

$$|\Psi'_{in}(x)| \le c\left(\sum_{j=0}^{n}(2n)^j - \sum_{j=0}^{n}2^j\right).$$

В результате,

$$|\Psi'_{in}(x)| \le c\left(\frac{(2n)^{n+1}-1}{2n-1} - \left(2^{n+1}-1\right)\right).$$

Отсюда

$$|\Psi'_{in}(x)| < c\left(2^{n+1}\left(\frac{n^{n+1}}{2n-1}-1\right)+1\right).$$

Деление числителя и знаменателя дроби на n влечет:

$$|\Psi'_{in}(x)| < c\left(2^{n+1}\left(\frac{n^{n}}{2-1/n}-1\right)+1\right),$$

тем более,

$$|\Psi'_{in}(x)| < c\left(2^{n+1}\left(n^{n}-1\right)+1\right).$$

Отсюда, с учетом $n \le N$, $N = \text{const}$, окончательно получаем искомую оценку:

$$|\Psi'_{in}(x)| < c_{00},\ c_{00} = c\left(2^{N+1}\left(N^{N}-1\right)+1\right). \qquad (\text{П5.19})$$

При этом

$$c_{00} = \text{const}\ \ \forall x \in [x_i, x_{i+1}] \wedge \forall i = \overline{0,\ 2^k-1} \wedge \forall k = 0, 1, \ldots \wedge \forall n : 2 \le n \le N.$$

Таким образом, согласно (П5.8) и (П5.19), с учётом условий интерполяции, а также оценки (П5.7), имеет место

Теорема П5.1. В рассматриваемых условиях, тех же, которые предполагаются выполненными для леммы П5.1, имеет место оценка

погрешности приближения функции при помощи варьируемого кусочно-полиномиального метода:

$$\left| f(x) - \Psi_{in}(x) \right| \le \widetilde{c}\, h_k, \quad \widetilde{c} = \text{const}, \tag{П5.20}$$

где $\widetilde{c} = c + c_{00}$. При этом c и c_{00} – константы из (П5.8), (П5.19), которые не зависят от x, от количества подынтервалов 2^k, а также от номера подынтервала i, степени полинома n, в ограничениях $2 \le n \le N$, и от расстояния между узлами интерполяции h_k на произвольно взятом подынтервале на $[\alpha, \beta]$ из (П5.1).

Согласно (П5.2) $h_k = \dfrac{\beta - \alpha}{2^k n}$. Отсюда и из (П5.20) вытекает

Следствие П5.2. В условиях и обозначениях теоремы П5.1 погрешность кусочно-полиномиальной интерполяции оценивается из неравенства:

$$\left| f(x) - \Psi_{in}(x) \right| < \widetilde{c}\, \frac{\beta - \alpha}{2^k n}\ . \tag{П5.21}$$

Поскольку правая часть неравенства (П5.20) инвариантна относительно k, i, а также относительно n, $2 \le n \le N$, то при $k = 0$, $2^k = 1$, и неравенство (П5.20) выполняется на исходном отрезке $[\alpha, \beta]$ для полинома n-й степени, $2 \le n \le N$, при этом $h_0 = h$ – расстояние между узлами интерполяции:

$$h_0 = \frac{\beta - \alpha}{n}. \tag{П5.22}$$

С учетом (П5.20) и (П5.22) имеет место

Следствие П5.3. В условиях и обозначениях теоремы П5.1, в частности, при непрерывности и двукратной непрерывной дифференцируемости функции $f(x)$, на всем отрезке $[\alpha, \beta]$ выполняется неравенство:

$$\left| f(x) - \Psi_n(x) \right| \le \widetilde{c}\ h, \ \widetilde{c} = \text{const}, \tag{П5.23}$$

где $\Psi_n(x)$ – интерполирущий данную функцию полином Ньютона для интерполирования вперед с равноотстоящими на расстояние h узлами интерполяции, $h = h_0$ из (П5.22).

С другой стороны, выбор в знаменателе правой части (П5.21) минимума $n = 2$ влечет:

$$\left| f(x) - \Psi_{in}(x) \right| < \frac{\widetilde{c}\ (\beta - \alpha)}{2^{k+1}} \quad \forall n : 2 \le n \le N,\ N = \text{const}. \qquad (\text{П5.24})$$

Оценка (П5.24) показывает, что решающую роль в точности приближения имеет число подынтервалов, а не степень полинома. Это иллюстрирует следующая таблица практически выполненных приближений.

Таблица П5.1

Степени интерполяционного полинома Ньютона, аппроксимирующего функцию $y = \sqrt{arctg\left(e^{\sin\sqrt[3]{(1/x)}}\right)}$, $x \in [0.5, 1]$ *по кусочно-полиномиальной схеме*

ε \ k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
10^{-4}	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10^{-5}	4	3	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
10^{-6}	6	5	3	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1
10^{-7}	8	6	4	3	3	2	2	1	1	1	1	1	1	1	1	1	1	1
10^{-8}	10	7	5	4	3	2	2	2	2	1	1	1	1	1	1	1	1	1
10^{-9}	11	8	6	5	4	3	3	2	2	2	2	1	1	1	1	1	1	1
10^{-10}		9	7	6	4	4	3	3	2	2	2	2	1	1	1	1	1	1
10^{-11}		11	8	6	5	4	3	3	3	2	2	2	2	2	1	1	1	1
10^{-12}		12	9	7	5	5	4	4	3	3	2	2	2	2	2	2	1	1
10^{-13}			9	8	6	5	4	4	4	3	3	2	2	2	2	2	2	1
10^{-14}			11	8	7	6	5	4	4	3	3	3	3	2	2	2	2	2
10^{-15}			12	9	7	6	5	5	4	4	3	3	3	2	2	2	2	2
10^{-16}			12	10	8	7	6	5	4	4	3	3	3	3	3	2	2	2
10^{-17}				10	9	7	6	6	5	4	4	3	3	3	3	2	2	2
10^{-18}						8	7	6	5	5	4	4	3	3	3	3	3	2

В табл. П5.1 названия столбцов $k = \overline{0, 17}$ соответствуют k-му этапу разбиения (количество подынтервалов $P = 2^k$), названия строк соответствуют

значениям априори заданной абсолютной погрешности приближения функции ε согласно условию (2.2). В ячейках таблицы представлены значения минимальной степени интерполяционного полинома Ньютона, позволяющего приблизить по кусочно-полиномиальной схеме значения функции $y=\sqrt{arctg\left(e^{\sin\sqrt[3]{(1/x)}}\right)}$ с погрешностью ε на k-м этапе разбиения. Отсутствие значений в некоторых ячейках таблицы означает невозможноть достижения соответственной точности приближения на данном этапе разбиения.

Степень полинома, тем не менее, оказывает положительное влияние на точность в соответствии с (П5.21), что также видно из приведенной таблицы.

Наконец, границы степеней вошли в константу $\widetilde{c}$ из (П5.20), (П5.21). При этом на практике они могли оказать отрицательное влияние согласно (П5.19), (П5.20), что соответствует варьируемой кусочной интерполяции, представленной в табл. П5.1.

Замечание П5.1. По сравнению с оценкой погрешности (1.15), полученной в предположении $(n+1)$-кратной непрерывной дифференцируемости аппроксимируемой функции, оценка (П5.21) теряет 2^{n+1} в знаменателе, а также h^{n+1} в числителе. Поскольку в (1.15) $h^{n+1}=h\,\dfrac{(\beta-\alpha)^n}{n^n}$, то при $\beta-\alpha<1$, это означало существенное снижение погрешности приближения, что соответствовало практике аппроксимации (табл. П5.1).

Однако оценка (П5.21) получена в более общих условиях лишь однократной непрерывной дифференцируемости аппроксимируемой функции и показывает в этих условиях возможность достижения высокой точности аппроксимации за счет сужения подынтервалов.

Оценка погрешности вычисления интегралов на основе варьируемой кусочно-полиномиальной аппроксимации подынтегральной функции. Пусть промежуток интегрирования разбивается на 2^k равных частей, на каждом из которых подынтегральная функция аппроксимируется в условиях

теоремы П5.1 с наибольшей точностью и наименьшей степенью полинома вида (П5.2), преобразованного к виду (2.9) по алгоритму, изложенному для $y = f(x)$ на $[\alpha, \beta] = \bigcup_{i=0}^{2^k-1} [x_i\ ,\ x_{i+1}]$. Пусть для некоторого произвольно выбранного $\varepsilon > 0$, начиная с некоторого k_0, при всех $k \geq k_0$ выполняется $|f(x) - \Psi_{in}(x)| \leq \varepsilon$, $i = \overline{0,\ 2^k - 1}$.

По аддитивности интеграла по промежутку имеет место равенство:

$$\int_{\alpha}^{\beta} f(x)\,dx = \sum_{i=0}^{2^k-1} \int_{x_i}^{x_{i+1}} f(x)\,dx.$$

Рассматриваемое приближение подынтегральной функции на i-м подынтервале полиномом из (П5.3) влечет:

$$\int_{x_i}^{x_{i+1}} f(x)\,dx \approx \int_{x_i}^{x_{i+1}} \Psi_{in}(x)\,dx.$$

Относительно подынтегральной функции, как и в теореме П5.1, предполагается её непрерывность и однократная непрерывная дифференцируемость на $[\alpha, \beta]$, так что при любом $i = 0,1, ..., 2^k - 1$ выполнено (П5.20), (П5.21).

Вводится обозначение:

$$\delta_i = \left| \int_{x_i}^{x_{i+1}} f(x)\,dx - \int_{x_i}^{x_{i+1}} \Psi_{in}(x)\,dx \right|.$$

Очевидно,

$$\delta_i \leq \int_{x_i}^{x_{i+1}} \left| f(x) - \Psi_{in}(x) \right| dx.$$

Отсюда с учетом (П5.21) следует неравенство:

$$\delta_i \leq \widetilde{c} \int_{x_i}^{x_{i+1}} \frac{\beta - \alpha}{2^k n}\,dx,$$

или,

$$\delta_i \le \widetilde{c}\,\frac{\beta-\alpha}{2^k n}\,(x_{i+1}-x_i).$$

Поскольку $x_{i+1}-x_i=\dfrac{\beta-\alpha}{2^k}$, последнее неравенство примет вид:

$$\delta_i \le \widetilde{c}\,\frac{(\beta-\alpha)^2}{2^{2k} n}. \qquad (П5.25)$$

Суммирование обеих частей (П5.25) по всему промежутку интегрирования влечет

$$\sum_{i=0}^{2^k-1}\delta_i \le 2^k\,\widetilde{c}\,\frac{(\beta-\alpha)^2}{2^{2k} n}.$$

Окончательно,

$$\left|\int_\alpha^\beta f(x)\,dx-\sum_{i=0}^{2^k-1}\int_{x_i}^{x_{i+1}}\Psi_{in}(x)\,dx\right| \le \widetilde{c}\,\frac{(\beta-\alpha)^2}{2^k n}. \qquad (П5.26)$$

Таким образом, имеет место

Теорема П5.2. В условиях теоремы П5.1 оценка погрешности вычисления интеграла при приближении подынтегральной функции с помощью варьируемого кусочно-полиномиального метода имеет вид (П5.26), где значение $\widetilde{c}=\mathrm{const}$ то же, что в (П5.20), число подынтервалов разбиения промежутка интегрирования 2^k взято из (2.1). Степень n интерполяционного полинома постоянна на каждом подынтервале и может быть выбрана как минимальная для произвольно фиксированной правой части (П5.26) в ограничениях $2 \le n \le N$.

Следствие П5.4. Выбор минимального n в правой части (П5.26) влечет:

$$\left|\int_\alpha^\beta f(x)\,dx-\sum_{i=0}^{2^k-1}\int_{x_i}^{x_{i+1}}\Psi_{in}(x)\,dx\right| \le \widetilde{c}\,\frac{(\beta-\alpha)^2}{2^{k+1}}. \qquad (П5.27)$$

Оценки (П5.26), (П5.27) означают сходимость рассматриваемых приближений со скоростью геометрической прогрессии от числа

подынтервалов в условиях непрерывности и однократной непрерывной дифференцируемости подынтегральной функции.

Следствие П5.5. В рассматриваемых условиях для $\forall\ \varepsilon_0 > 0\ \ \exists\, k_0 = \text{const}$, такое что неравенство (П5.27) выполнено для $\forall\ k \geq k_0$. Для всех таких k имеет место оценка:

$$\left| \int_{\alpha}^{\beta} f(x)\,dx - \sum_{i=0}^{2^k-1} \int_{x_i}^{x_{i+1}} \Psi_{in}(x)\,dx \right| \leq \varepsilon_0,$$

В частности, если $k_0 = \left\lceil \log_2 \dfrac{\widetilde{c}\,(\beta - \alpha)^2}{\varepsilon_0} \right\rceil - 1$.

П5.7. Приложение к п. 5.3.5. Как отмечалось в приложении к главе 3 трудоемкость кусочно-интерполяционного решения задачи Коши для ОДУ существенно снижается при отказе от автоматического подбора параметров и замене его пользовательским подбором и фиксированием параметров метода, при этом сохраняется высокая точность и непрерывный характер приближения.

Ниже представлены результаты численного эксперимента по приближенному решению задачи Коши для систем ОДУ, имеющих аналитически представимые точные решения, сравнением с которыми определялась каноническая норма абсолютной погрешности приближения. Программные реализации использованных в эксперименте разностных методов Эйлера, Эйлера-Коши, Рунге-Кутты 4-го, Бутчера 6-го и Дормана-Принса 8-го порядков аппроксимации приведены в приложении к главе 2, программная реализация модификации кусочно-интерполяционного метода с итерационным уточнением при фиксированных значениях параметров представлена в приложении к главе 3. Длина промежутка интегрирования принята равной 10^4, величина шага разностных методов – $h = 10^{-4}$, параметры кусочно-интерполяционного метода с итерационным уточнением указаны для каждой задачи непосредственно после таблиц с результатами приближений.

Пример П5.1. Рассматривается задача Коши

$$y_1' = t + 2y_1/t - \sqrt{y_2},\ y_2' = 2\sqrt{y_2},\ y_1(1) = 2,\ y_2(1) = 4,$$

с аналитическим решением $y_1 = t + t^2$, $y_1 = (t+1)^2$. Норма абсолютной погрешности его приближения разностными и кусочно-интерполяционным методами на отрезке $t \in [1, 10001]$ представлена в табл. П5.2 наряду с временем выполнения программ.

Таблица П5.2

Погрешность и время решения задачи из примера П5.1 разностными и кусочно-интерполяционным методами

t	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*FPI*
	0:6:298	0:16:329	0:22:961	0:41:139	1:15:542	0:0:109
1001	6.573**e+01**	3.758**e-03**	5.400**e-12**	9.095**e-13**	8.697**e-12**	0.000**e+00**
2001	2.622**e+02**	1.502**e-02**	1.955**e-11**	1.387**e-11**	4.570**e-11**	0.000**e+00**
3001	5.894**e+02**	3.377**e-02**	4.911**e-11**	1.273**e-11**	1.128**e-10**	0.000**e+00**
4001	1.047**e+03**	6.003**e-02**	6.548**e-11**	1.910**e-11**	2.174**e-10**	0.000**e+00**
5001	1.636**e+03**	9.378**e-02**	6.549**e-11**	7.276**e-11**	3.511**e-10**	0.000**e+00**
6001	2.355**e+03**	1.350**e-01**	2.146**e-10**	9.823**e-11**	4.184**e-10**	0.000**e+00**
7001	3.205**e+03**	1.838**e-01**	1.273**e-10**	1.273**e-10**	7.276**e-10**	0.000**e+00**
8001	4.185**e+03**	2.400**e-01**	1.746**e-10**	1.237**e-10**	9.641**e-10**	0.000**e+00**
9001	5.297**e+03**	3.038**e-01**	4.438**e-10**	1.965**e-10**	9.532**e-10**	0.000**e+00**
10001	6.539**e+03**	3.750**e-01**	8.586**e-10**	3.201**e-10**	1.593**e-09**	0.000**e+00**

Во второй строке таблицы представлено время работы программ: двоеточия отделяют минуты, секунды и миллисекунды. Кусочно-интерполяционной метод дает «нулевые» значения нормы погрешности за время $109ms$ (параметры метода: `velint:=1000`, $n=3$, $k=11$, $\ell=25$), разностные методы существенно уступают предложенному методу при решении данной задачи как в точности, так и в быстродействии. Так, предложенный метод оказался в 150 раз быстрее метода Эйлера-Коши, в 210 раз быстрее метода Рунге-Кутты, в 377 раз быстрее метода Бутчера и в 693 раза быстрее метода Дормана-Принса. Метод Эйлера с принятой величиной шага дает неверные значения приближения.

Пример П5.2. Задача Коши

$$y_1' = 2y_1 + 4y_2 + \cos(t),\ \ y_2' = -y_1 - 2y_2 + \sin(t),\quad y_1(0) = -4,\ y_2(0) = 1,$$

имеет аналитическое решение $y_1 = -3\sin(t) - 2\cos(t) - 2$, $y_2 = 2\sin(t) + 1$. Погрешность его приближения на отрезке $t \in [0, 10^4]$ вместе свременем работы программ представлена в табл. П5.3.

Таблица П5.3

Погрешность и время решения задачи из примера П5.2 разностными и кусочно-интерполяционным методами

t	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*FPI*
	0:9:931	0:27:70	0:38:597	1:9:247	2:6:414	0:0:922
1000	5.764**e-01**	3.336**e-06**	2.518**e-13**	8.568**e-14**	7.255**e-13**	5.186**e-15**
2000	9.624**e-01**	6.668**e-06**	1.314**e-13**	6.609**e-13**	5.838**e-13**	1.337**e-14**
3000	1.222**e+00**	9.998**e-06**	1.106**e-12**	2.945**e-12**	8.296**e-13**	2.281**e-14**
4000	1.396**e+00**	1.333**e-05**	2.485**e-12**	6.730**e-12**	7.732**e-13**	5.494**e-14**
5000	1.514**e+00**	1.666**e-05**	4.618**e-12**	9.390**e-12**	2.247**e-12**	7.701**e-14**
6000	1.592**e+00**	2.000**e-05**	7.697**e-12**	1.141**e-11**	5.673**e-12**	1.057**e-13**
7000	1.644**e+00**	2.334**e-05**	1.063**e-11**	1.320**e-11**	8.458**e-12**	1.292**e-13**
8000	1.678**e+00**	2.667**e-05**	1.328**e-11**	1.517**e-11**	8.355**e-12**	1.433**e-13**
9000	1.701**e+00**	3.000**e-05**	1.541**e-11**	1.615**e-11**	1.035**e-11**	1.471**e-13**
10000	1.717**e+00**	3.333**e-05**	1.724**e-11**	1.865**e-11**	1.077**e-11**	1.543**e-13**

Граница погрешности кусочно-интерполяционного приближения не превышает порядка 10^{-13} (параметры метода: `velint:=10`, $n=10$, $k=4$, $\ell=15$), время работы программы $922ms$. Метод Эйлера, также как и в предыдущем примере, дает неверные результаты приближения. Разностные методы высшего порядка, характеризуясь границей погрешности порядка 10^{-11}, уступают в быстродействии предложенному методу: метод Рунге-Кутты – в 42 раза, метод Бутчера – в 75 раз, метод Дормана-Принса – в 137 раз.

Пример П5.3. Рассматривается задача Коши

$$y_1' = y_1/t - t^2 y_1 y_2^{\,2}, y_2' = -y_2/t,\ y_1(1) = e^{-1},\ y_2(1) = 1,$$

с аналитическим решением $y_1 = te^{-t}$, $y_2 = 1/t$. Каноническая норма абсолютной погрешности его приближения на отрезке $t \in [1, 10001]$ и время работы рассматриваемых программ даны в табл. П5.4.

Таблица П5.4

Погрешность и время решения задачи из примера П5.3 разностными и кусочно-интерполяционным методами

t	*Euler*	*Euler-Cauchy*	*Runge-Kutta 4*	*Butcher_6*	*Dormand-Prince 8*	*FPI*
	0:5:453	0:13:680	0:19:692	0:34:440	1:6:23	0:0:171
1001	9.980**e-08**	4.616**e-20**	4.606**e-20**	5.474**e-20**	9.455**e-20**	4.235**e-22**
2001	4.995**e-08**	3.176**e-21**	3.123**e-21**	8.449**e-20**	1.880**e-19**	4.235**e-22**
3001	3.331**e-08**	1.977**e-20**	1.980**e-20**	4.468**e-20**	8.611**e-20**	7.941**e-23**
4001	2.499**e-08**	2.374**e-20**	2.377**e-20**	1.242**e-19**	1.397**e-19**	7.941**e-23**
5001	1.999**e-08**	9.516**e-21**	9.542**e-21**	9.381**e-20**	1.536**e-19**	2.250**e-22**
6001	1.666**e-08**	3.150**e-21**	3.176**e-21**	7.185**e-20**	1.275**e-19**	1.323**e-22**
7001	1.428**e-08**	4.738**e-21**	4.725**e-21**	6.559**e-20**	6.411**e-20**	2.912**e-22**
8001	1.250**e-08**	5.466**e-21**	5.453**e-21**	3.113**e-20**	6.554**e-20**	2.250**e-22**
9001	1.111**e-08**	2.627**e-21**	2.614**e-21**	3.451**e-20**	6.000**e-20**	3.640**e-22**
10001	9.998**e-09**	4.659**e-21**	4.645**e-21**	2.774**e-20**	2.714**e-20**	2.912**e-22**

Значения погрешности кусочно-интерполяционного приближения с параметрами: `velint:=10`, $n=6$, $k=4$, $\ell=15$, не превышают порядка 10^{-22}, время работы программы $171ms$. Разностные методы высшего порядка уступают на два десятичных порядка и в точности, и в быстродействии. Методы Эйлера и Эйлера-Коши уступают в быстродействии в 32 и в 80 раз соответственно.

Пример П5.4. Задача Коши

$$y_1' = y_1 \sin t + y_2 \cos t,\ y_2' = -y_1 \cos t + y_2 \sin t,\ y_1(0) = e^{-1},\ y_2(0) = e^{-1},$$

имеет аналитическое решение $y_1 = \left(\cos(\sin t) + \sin(\sin t)\right) e^{-\cos t}$, $y_2 = \left(-\sin(\sin t) + \cos(\sin t)\right) e^{-\cos t}$. Значения погрешностей его приближения на отрезке $t \in [0, 10^4]$ с помощью рассматриваемых программ даны в табл. П5.5 наряду с временем решения.

Таблица П5.5

Погрешность и время решения задачи из примера П5.4 разностными и кусочно-интерполяционным методами

t	*Euler*	*Euler-Cauchy*	*Runge-Kutta 4*	*Butcher_6*	*Dormand-Prince 8*	*FPI*
	0:16:448	0:46:956	1:4:379	1:57:240	3:35:449	0:4:698
1000	2.028**e-02**	6.090**e-10**	4.830**e-17**	1.738**e-16**	1.281**e-16**	1.464**e-18**
2000	1.059**e-01**	2.366**e-09**	2.539**e-16**	5.820**e-16**	8.747**e-16**	4.879**e-19**

3000	2.710**e-01**	1.818**e-09**	8.236**e-16**	1.103**e-15**	2.808**e-15**	5.855**e-18**
4000	2.888**e-01**	3.657**e-09**	5.566**e-16**	9.147**e-16**	1.525**e-15**	7.427**e-18**
5000	1.211**e-01**	1.542**e-09**	7.264**e-17**	5.809**e-16**	2.304**e-16**	3.144**e-18**
6000	8.794**e-02**	2.194**e-10**	3.650**e-17**	2.272**e-16**	1.980**e-16**	1.030**e-18**
7000	7.579**e-02**	2.824**e-10**	7.763**e-17**	2.187**e-16**	2.567**e-16**	9.758**e-19**
8000	2.541**e-01**	1.814**e-09**	2.883**e-16**	5.642**e-16**	5.005**e-16**	3.551**e-18**
9000	5.424**e-01**	3.559**e-09**	4.606**e-16**	1.712**e-15**	9.431**e-16**	3.036**e-18**
10000	8.833**e-01**	2.082**e-09**	5.987**e-16**	2.784**e-15**	1.630**e-15**	9.758**e-18**

Граница погрешности кусочно-интерполяционного приближения не превышает порядка 10^{-18} (параметры метода: `velint:=10`, $n=10$, $k=6$, $\ell=18$) при времени решения $4s\ 689ms$. При решении данной задачи предложенный метод оказался в 3 раз быстрее метода Эйлера, в 14 раз быстрее метода Рунге-Кутты, в 25 раз быстрее метода Бутчера и в 46 раз быстрее метода Дормана-Принса, превышая последние три по точности на два и более десятичных порядка.

Пример П5.5. Рассматривается задача Коши

$$y_1' = -y_2\sqrt{y_1^2+y_2^2},\ y_2' = y_1\sqrt{y_1^2+y_2^2},\ y_1(0)=1,\ y_2(0)=1,$$

с точным решением $y_1=\sqrt{2}\cos\left(\sqrt{2}t+\pi/4\right),\quad y_2=\sqrt{2}\sin\left(\sqrt{2}t+\pi/4\right).$ В табл. П5.6 даны значения погрешностей его приближения рассматриваемыми методами на отрезке из примера П5.4 и время работы программ.

Таблица П5.6

Погрешность и вермя решения задачи из примера П5.5 разностными и кусочно-интерполяционным методами

t	*Euler*	*Euler-Cauchy*	*Runge-Kutta_4*	*Butcher_6*	*Dormand-Prince_8*	*FPI*
	0:6:357	0:17:803	0:23:483	0:41:584	1:17:576	0:2:568
1000	1.718**e+00**	1.646**e-05**	1.076**e-14**	3.541**e-14**	1.298**e-13**	2.591**e-16**
2000	1.753**e+00**	3.457**e-05**	2.283**e-14**	2.687**e-13**	3.310**e-13**	2.512**e-15**
3000	2.394**e+00**	4.150**e-05**	5.513**e-14**	7.176**e-13**	3.521**e-13**	3.321**e-15**
4000	2.505**e-01**	6.965**e-05**	1.828**e-13**	1.232**e-12**	5.089**e-13**	7.994**e-16**
5000	1.566**e+00**	9.505**e-05**	3.213**e-13**	1.783**e-12**	3.317**e-13**	8.704**e-15**
6000	2.112**e+00**	9.569**e-05**	4.042**e-13**	1.947**e-12**	1.574**e-13**	1.794**e-14**
7000	2.994**e-01**	1.272**e-04**	6.799**e-13**	2.968**e-12**	4.046**e-13**	3.525**e-14**
8000	1.040**e+00**	1.651**e-04**	9.161**e-13**	4.474**e-12**	4.556**e-13**	5.847**e-14**
9000	2.168**e+00**	1.629**e-04**	7.852**e-13**	4.836**e-12**	3.253**e-14**	7.033**e-14**
10000	2.368**e+00**	1.870**e-04**	8.399**e-13**	5.543**e-12**	2.076**e-13**	9.533**e-14**

Предложенный метод с параметрами: `velint:=10`, $n=10$, $k=5$, $\ell=20$, дает решение задачи из примера П5.5 с абсолютной погрешностью, не

превышающей порядка 10^{-14}, за $2s\,568ms$, что согласно табл. П5.6 в 7 раз быстрее метода Эйлера-Коши, в 9 раз быстрее метода Рунге-Кутты, в 16 раз быстрее метода Бутчера и в 30 раз быстрее метода Дормана-Принса. При этом точность кусочно-интерполяционного приближения на один, два десятичных порядка выше точности разностных методов высшего порядка. Применение метода Эйлера с шагом 10^{-4} не позволило получить корректное приближение решения на рассматриваемом отрезке. Следует отметить, что применение кусочно-интерполяционного метода с параметрами: `velint:=100`, $n=10$, $k=10$, $\ell=15$, дает решение этой же задачи с границей абсолютной погрешности порядка 10^{-15} за $5s\,836ms$, оставаясь при этом быстрее рассмотренных разностных методов.

Таким образом, во всех рассмотренных выше примерах при принятых параметрах сравнения предложенная модификация кусочно-интерполяционного метода по быстродействию в несколько раз превосходит метод Эйлера, в десятки раз – метод Рунге-Кутты 4-го порядка, до сотни раз оказалась быстрее методов Бутчера и Дормана-Принса при условии меньшей погрешности приближения.

В приложении представлены программные реализации инвариантного метода кусочно-интерполяционного решения жестких и нежестких систем ОДУ, входящие в программный комплекс, на основе которых получены обсуждаемые в главе 5 результаты уточненного численного моделирования периодических автоколебательных реакций, автоколебаний в задачах радиоэлектроники, движения космических аппаратов с управлением при помощи «малой тяги», возмущенного движения космического аппарата. Также приведены результаты численного эксперимента, подтверждающие сравнительно высокое быстродействие модификации предложенного метода, заключающейся в замене автоматического подбора параметров пользовательским подбором и фиксированием параметров метода, при этом сохраняются высокая точность и непрерывный характер приближения.

ПРИЛОЖЕНИЕ К ГЛАВЕ 6

П6.1. Приложение к п. 6.3. Приводятся результаты расширенного численного эксперимента по кусочно-интерполяционному решению системы (6.1), предложенной в интерфейсном контрольном документе ГЛОНАСС для выражения математической модели орбитального движения НКА ГЛОНАСС. Результаты численного моделирования системы (6.1) с начальными данными (6.6), определяющими положение и компонены вектора скорости движения НКА ГЛОНАСС №730 в системе ПЗ-90.11 на момент времени $t_b = 11700$ даты 05.08.2021 ($N_T = 583$, $N_4 = 7$) шкалы МДВ, представлены нижедля отрезков интегрирования $[t_b - \Delta t, t_b]$, $\Delta t \in \{300\,c,\ 600\,c,\ 900\,c\}$. Результат работы программы *fpi_glonass* (глава 6, листинг 6.1) при $t_i = 11400$:

```
Calculationtime:  2.29000000000000E-0001

 x =  2.50732546741584E+0007  y =  3.70132538988291E+0005  z = -4.72005283502666E+0006
vx = -6.55064088682716E+0002 vy = -9.49971553422365E+0001 vz = -3.48535231920772E+0003

 P1 = 0,0000E+000 P2 = 9,0949E-013 P3 = 0,0000E+000
 P4 = 0,0000E+000 P5 = 0,0000E+000 P6 = 0,0000E+000
```

при $t_i = 11100$:

```
Calculation time:  2.29100000000000E-0001

 x =  2.52479703502125E+0007  y =  4.02293478734320E+0005  z = -3.66972692057345E+0006
vx = -5.09373974562800E+0002 vy = -1.18316598624554E+0002 vz = -3.51555999693191E+0003

 P1 = 0,0000E+000 P2 = 9,0949E-013 P3 = 0,0000E+000
 P4 = 0,0000E+000 P5 = 0,0000E+000 P6 = 0,0000E+000
```

при $t_i = 10800$:

```
Calculation time:  2.28600000000000E-0001

 x =  2.53787058994140E+0007  y =  4.40461175256494E+0005  z = -2.61147611825596E+0006
vx = -3.61952522189109E+0002 vy = -1.35023964705675E+0002 vz = -3.53817540705169E+0003

 P1 = 0,0000E+000 P2 = 0,0000E+000 P3 = 0,0000E+000
 P4 = 0,0000E+000 P5 = 0,0000E+000 P6 = 0,0000E+000
```

Значения координат НКА и погрешность их приближения указаны в метрах, значения составляющих вектора скорости центра масс НКА и погрешностей их приближения – в метрах в секунду, время решения указано в миллисекундах. Параметры кусочно-интерполяционного метода: $n = 8$, $\ell = 12$. Следует отметить, что программа *fpi_glonass* позволяет выполнить численное

моделирование движения НКА как на отрезке $[t_b - \Delta t, t_b], \Delta t \le 900c$, так и на отрезке $[t_b, t_b + \Delta t]$, $\Delta t \le 900c$, без внесения каких-либо дополнительных изменений в программный код.

Представленный в главе 6, на рис. 6.2 график изменения погрешности приближения координат центра масс НКА ГЛОНАСС №730 на 15-минутном интервале прогнозирования построен на основе данных, которые рассчитаны и сохранены в файл для визуализации по программе, код которой представлен в листинге П6.1.

Листинг П6.1

```
program fpi_glonass_with_vis; {$APPTYPE CONSOLE}
uses SysUtils, Math, System.Diagnostics; type vectNeq = array[1..6] of extended;
const N4 = 7; NT = 583; tb = 11700; ti = 12600;
      y00:vectNeq = (24855158.20312, 345943.8476562, -5760185.546875,
              -798.4914779663, -65.19222259521, -3447.617530823);
procedure pi_calculation_of_ephemeris_glonass(N4,NT:integer; tb,ti:extended;y00:vectNeq);
const
 Neq=6; w3=7.2921151467e-5; J2=1082.62575e-6; GM=3.986004418e+14; ae=6378136;
 G_l=4902.799e+9; G_c=13271244.0e+13; a_l=3.84385243e+8; a_c=1.49598e+11;
 e_l=0.054900489; e_c=0.016719; i_l=0.0898041080;
type Prc=array[1..8] of extended; matr= array[0..8,0..8] of integer;
     matrC=array[-3..9] of extended; MmatrC=array[1..Neq] of matrC;
var yn_etalon15,yn_etalon10,yn_etalon5,y0,yn,yn0,yr: vectNeq; JD0,GMST,ERA,dT,S:extended;
i:byte; SW: TStopwatch; vPrc:array[0..8] of Prc; Npol,k_iter:integer;
    MTime:array[1..10000] of extended; ii:integer;C_et, C_calc:MmatrC;s_rez:text;
    ll, k_output:longint;xpr:extended;
procedure ls_acceleration_prep(x:extended; var KEKr_:Prc);
var            sinv_l,cosv_l,sinv_c,cosv_c,w_c,eps,Gamma_,Ksi11,Ksi12,Eta11,Eta12,Kapa11,
Kapa12:extended; q_l,q_c, T, Omega_l, Ksi_, Eta_, Kapa_,El,Ec:extended;
function Ekk(q_k,e_k:extended):extended; var Ek_,Ek_new:extended;
begin Ek_new:=q_k;
 repeat Ek_:=Ek_new; Ek_new:=q_k+e_k*sin(Ek_); until abs(Ek_new-Ek_)<1e-8;
 Ekk:=Ek_; end;
begin T:=(JD0+(x-10800)/86400-2451545.0)/36525;
q_l:=2.3555557435+8328.6914257190*T+0.0001545547*sqr(T);
q_c:=6.2400601269+628.3019551714*T-(2.6820e-6)*sqr(T);
Omega_l:=2.1824391966-33.7570459536*T+0.0000362262*sqr(T);
Gamma_:=1.4547885346+71.0176852437*T-0.0001801481*sqr(T);
w_c:=-7.6281824375+0.0300101976*T+(7.9741e-6)*sqr(T); eps:=0.4090926006-0.0002270711*T;
Ksi_:=1-sqr(cos(Omega_l))*(1-cos(i_l)); Eta_:=sin(Omega_l)*sin(i_l);
Kapa_:=cos(Omega_l)*sin(i_l); Ksi11:=sin(Omega_l)*cos(Omega_l)*(1-cos(i_l));
Ksi12:=1-sqr(sin(Omega_l))*(1-cos(i_l)); Eta11:=Ksi_*cos(eps)-Kapa_*sin(eps);
Eta12:=Ksi11*cos(eps)+Eta_*sin(eps); Kapa11:=Ksi_*sin(eps)+Kapa_*cos(eps);
Kapa12:=Ksi11*sin(eps)-Eta_*cos(eps); El:=Ekk(q_l,e_l); Ec:=Ekk(q_c,e_c);
sinv_l:=(sqrt(1-sqr(e_l))*sin(El))/(1-e_l*cos(El));
sinv_c:=(sqrt(1-sqr(e_c))*sin(Ec))/(1-e_c*cos(Ec));
cosv_l:=(cos(El)-e_l)/(1-e_l*cos(El)); cosv_c:=(cos(Ec)-e_c)/(1-e_c*cos(Ec));
KEKr_[1]:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Ksi11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Ksi12;
KEKr_[2]:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Eta11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Eta12;
KEKr_[3]:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Kapa11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Kapa12;
KEKr_[4]:=cosv_c*cos(w_c)-sinv_c*sin(w_c); KEKr_[7]:=a_l*(1-e_l*cos(El));
KEKr_[5]:=(sinv_c*cos(w_c)+cosv_c*sin(w_c))*cos(eps);
KEKr_[6]:=(sinv_c*cos(w_c)+cosv_c*sin(w_c))*sin(eps); KEKr_[8]:=a_c*(1-e_c*cos(Ec));
end;
```

```
function f(eq:integer;x:extended;yy:vectNeq;KEKr:Prc):extended;
var r,GM_,y1_,y2_,y3_,ro,jx0_c,jy0_c,jz0_c,jx0_l,jy0_l,jz0_l:extended;
procedure ls_acceleration(y1,y2,y3:extended;KEKr:Prc;
var jx0_c,jy0_c,jz0_c,jx0_l,jy0_l,jz0_l:extended);
var delta_c,delta_l,x_c,y_c,z_c,x_l,y_l,z_l,Gl_,Gc_:extended;
begin x_l:=y1/KEKr[7]; y_l:=y2/KEKr[7]; z_l:=y3/KEKr[7]; Gl_:=G_l/sqr(KEKr[7]);
x_c:=y1/KEKr[8]; y_c:=y2/KEKr[8]; z_c:=y3/KEKr[8]; Gc_:=G_c/sqr(KEKr[8]);
delta_l:=Power(sqr(KEKr[1]-x_l)+sqr(KEKr[2]-y_l)+sqr(KEKr[3]-z_l),1.5);
delta_c:=Power(sqr(KEKr[4]-x_c)+sqr(KEKr[5]-y_c)+sqr(KEKr[6]-z_c),1.5);
jx0_l:=Gl_*((KEKr[1]-x_l)/delta_l-KEKr[1]); jy0_l:=Gl_*((KEKr[2]-y_l)/delta_l-KEKr[2]);
jz0_l:=Gl_*((KEKr[3]-z_l)/delta_l-KEKr[3]); jx0_c:=Gc_*((KEKr[4]-x_c)/delta_c-KEKr[4]);
jy0_c:=Gc_*((KEKr[5]-y_c)/delta_c-KEKr[5]); jz0_c:=Gc_*((KEKr[6]-z_c)/delta_c-KEKr[6]);
end;
begin
 case eq of 1: f:=yy[4]; 2: f:=yy[5]; 3: f:=yy[6];
 else begin r:=sqrt(yy[1]*yy[1]+yy[2]*yy[2]+yy[3]*yy[3]); GM_:=GM/sqr(r);
   y1_:=yy[1]/r; y2_:=yy[2]/r; y3_:=yy[3]/r; ro:=ae/r;
   ls_acceleration(yy[1],yy[2],yy[3],KEKr,jx0_c,jy0_c,jz0_c,jx0_l,jy0_l,jz0_l);
   case eq of
    4: f:=-GM_*y1_-1.5*J2*GM_*ro*ro*y1_*(1-5*sqr(y3_))+jx0_c+jx0_l;
    5: f:=-GM_*y2_-1.5*J2*GM_*ro*ro*y2_*(1-5*sqr(y3_))+jy0_c+jy0_l;
    6: f:=-GM_*y3_-1.5*J2*GM_*ro*ro*y3_*(3-5*sqr(y3_))+jz0_c+jz0_l;
end; end; end; end;
function Gorner(Mcoef:matrC; x:extended):extended;
var i,n:byte; s,t:extended;
begin
 t:=(x-Mcoef[-1])/Mcoef[-2]; n:=trunc(Mcoef[-3]); s:=Mcoef[n];
 for i:=n-1 downto 0 do s:=t*s+Mcoef[i]; Gorner:=s
end;
procedure RD(yin:vectNeq; A_nach,B_konech:extended; var yout:vectNeq; var C:MmatrC);
const dp: array[1..8,1..8] of extended =
((1, -1/2, 2/6, -6/24, 24/120, -120/720,  720/5040, -5040/40320),
 (0,  1/2,-3/6, 11/24,-50/120,  274/720,-1764/5040, 13068/40320),
 (0,    0, 1/6, -6/24, 35/120, -225/720, 1624/5040,-13132/40320),
 (0,    0,   0,  1/24,-10/120,   85/720, -735/5040,  6769/40320),
 (0,    0,   0,     0,  1/120,  -15/720,  175/5040, -1960/40320),
 (0,    0,   0,     0,      0,    1/720,  -21/5040,   322/40320),
 (0,    0,   0,     0,      0,        0,    1/5040,   -28/40320),
 (0,    0,   0,     0,      0,        0,         0,     1/40320));
type Mmatr=array[0..8,0..8] of vectNeq; vect=array[0..10] of extended;
     Mvect= array[0..8] of vectNeq;
var i,k,j,iter,r:byte; h:extended; x:vect; pp,s:vectNeq; y,fy,A:Mvect; dy:Mmatr;
begin h:=(B_konech-A_nach)/Npol;
 for j:=0 to Npol do begin x[j]:=A_nach+j*h; ls_acceleration_prep(x[j],vPrc[j]);end;
 for i:=1 to Neq do y[0,i]:=yin[i];
 for j:=1 to Npol do
  for i:=1 to Neq do
   begin fy[j-1,i]:=f(i,x[j-1],y[j-1],vPrc[j-1]); y[j,i]:=y[j-1,i]+h*fy[j-1,i]; end;
 for i:=1 to Neq do begin fy[Npol,i]:=f(i,x[Npol],y[Npol],vPrc[Npol]);
A[0,i]:=fy[0,i];end;
 for iter:=1 to k_iter do begin
  if iter>1 then for j:=1 to Npol do for i:=1 to Neq do fy[j,i]:=f(i,x[j],y[j],vPrc[j]);
   for j:=0 to Npol-1 do for i:=1 to Neq do dy[1,j,i]:=fy[j+1,i]-fy[j,i];
   for k:=2 to Npol do for j:=0 to Npol-k do
    for i:=1 to Neq do dy[k,j,i]:=dy[k-1,j+1,i]-dy[k-1,j,i];
   for k:=1 to Npol do begin
     for i:=1 to Neq do s[i]:=0;
     for j:=k to Npol do for i:=1 to Neq do s[i]:=s[i]+dp[k,j]*dy[j,0,i];
     for i:=1 to Neq do A[k,i]:=s[i];
    end;
   for j:=1 to Npol do
    begin
     for i:=1 to Neq do pp[i]:=A[Npol,i]/(Npol+1);
     for r:=Npol-1 downto 0 do for i:=1 to Neq do pp[i]:=pp[i]*j+A[r,i]/(r+1);
     for i:=1 to Neq do y[j,i]:=pp[i]*h*j+y[0,i];
end;end;
 for i:=1 to Neq do yout[i]:=y[Npol,i];
//Для расчета погрешности и "плотного" вывода результатов приближения
for i:=1 to Neq do
```

```
  begin C[i,0]:=y[0,i]; C[i,-1]:=x[0]; C[i,-2]:=h; C[i,-3]:=Npol+1; end;
 for i:=1 to Neq do for j:=1 to Npol+1 do C[i,j]:=A[j-1,i]*h/j;
end;
begin
 assign(s_rez,'result_Glonass_FPI_Err.dat'); rewrite(s_rez);
 JD0:=1461*(N4-1)+NT+2450082.5;
 ERA:=2*PI*(0.7790572732640+1.00273781191135448*(JD0-2451545.0));
 dT:=(JD0-2451545.0)/36525;
 GMST:=ERA+7.03270726e-8+0.0223603658710194*dT+
      6.7465784654e-6*power(dT,2)-2.1332e-12*power(dT,3)-
      1.452308e-10*power(dT,4)-1.784e-13*power(dT,5);
 S:=GMST+w3*(tb-10800);
 y0[1]:=y00[1]*cos(S)-y00[2]*sin(S); y0[2]:=y00[1]*sin(S)+y00[2]*cos(S);
 y0[3]:=y00[3]; y0[4]:=y00[4]*cos(S)-y00[5]*sin(S)-w3*y0[2];
 y0[5]:=y00[4]*sin(S)+y00[5]*cos(S)+w3*y0[1]; y0[6]:=y00[6];
 Npol:= 8; k_iter:= 12;
 RD(y0,tb,ti,yn0,C_et);
 S:=GMST+w3*(ti-10800);
 yn[1]:=yn0[1]*cos(S)+yn0[2]*sin(S); yn[2]:=-yn0[1]*sin(S)+yn0[2]*cos(S);
 yn[3]:=yn0[3]; yn[4]:=yn0[4]*cos(S)+yn0[5]*sin(S)+w3*yn[2];
 yn[5]:=-yn0[4]*sin(S)+yn0[5]*cos(S)-w3*yn[1]; yn[6]:=yn0[6];
 writeln(' x = ',yn[1]:30:20,'  y = ',yn[2]:30:20,'  z = ',yn[3]:30:20);
 writeln('vx = ',yn[4]:30:20,' vy = ',yn[5]:30:20,' vz = ',yn[6]:30:20);
 Npol := 5; k_iter := 7; RD(y0,tb,ti,yr,C_calc);
 k_output:=900;
 for ll:=0 to k_output do
  begin xpr:=tb+ll*(ti-tb)/k_output;
   writeln(s_rez,xpr,' ',abs(Gorner(C_et[1],xpr)-Gorner(C_calc[1],xpr)),' ',
   abs(Gorner(C_et[2],xpr)-Gorner(C_calc[2],xpr)),' ',abs(Gorner(C_et[3],xpr)-
Gorner(C_calc[3],xpr)));
  end;
 close(s_rez);
end;
begin
pi_calculation_of_ephemeris_glonass(N4,NT,tb,ti,y00); readln;
end.
```

В отличие от программы *fpi_glonass* представленная выше программа *fpi_glonass_with_vis* позволяет хранить коэффициенты полиномиальных приближений компоненов решения уравнений движения, рассчитывать значения приближения по схеме Горнера (процедура *Gorner*), рассчитывать и сохранять в файл *result_Glonass_FPI_Err.dat* погрешности приближения компонентов решения в равномерно распределенных моментах времени из интервала приближения, количество которых определяется значением переменной *k_output*.

П6.2. Приложение к п. 6.4. Программа для расчета координат и составляющих вектора скорости НКА ГЛОНАСС на заданный момент времени по данным эфемерид на основе метода Рунге-Кутты 4 порядка представлена в листинге П6.2. Для расчета с использованием метода Дормана-Принса 8-го порядка аппроксимации достаточно изменить код процедуры *RK4(x,h,ys,yr)*.

Листинг П6.2

```
program rk4_glonass; {$APPTYPE CONSOLE} uses SysUtils, Math, System.Diagnostics;
const Neq = 6; J2 = 1082.62575e-6; GM = 3.986004418e+14;  ae = 6378136;
G_l = 4902.799e+9; G_c = 13271244.0e+13; a_l = 3.84385243e+8; a_c = 1.49598e+11;
 e_l = 0.054900489;  e_c = 0.016719; i_l = 0.0898041080; w3 = 7.2921151467e-5;
type vectNeq = array[1..Neq] of extended;
consth=1;//шаг интегрирования
y00:vectNeq = (24855158.20312, 345943.8476562, -5760185.546875,
-798.4914779663, -65.19222259521, -3447.617530823);
      N4 = 7; NT = 583; tb = 11700; ti = 12600;
var y0,yn,ys,yr: vectNeq; m:int64; i:word; SW: TStopwatch;
 x,S,JD0,GMST,ERA,dT,jx0_c,jy0_c,jz0_c,jx0_l,jy0_l,jz0_l:extended;
function Ekk(q_k, e_k:extended):extended;var Ek_,Ek_new:extended;
begin Ek_new:=q_k;
repeat Ek_:=Ek_new; Ek_new:=q_k+e_k*sin(Ek_); until abs(Ek_new-Ek_)<1e-15;
 Ekk:=Ek_;end;
procedure ls_acceleration(x,y1,y2,y3:extended; var jx0_c,jy0_c,jz0_c,jx0_l,jy0_l,
jz0_l:extended);
varKsi_c,Eta_c,Kapa_c,Ksi_l,Eta_l,Kapa_l:extended;
 r_c,r_l,delta_c, delta_l,x_c,y_c,z_c,x_l,y_l,z_l:extended;
sinv_l,cosv_l,sinv_c,cosv_c,w_c,eps,Gamma_,Ksi11,Ksi12,Eta11,Eta12,Kapa11,Kapa12:extended
;
 q_l,q_c, T, Omega_l, Ksi_, Eta_, Kapa_,El,Ec:extended; Gl_,Gc_:extended;
begin
T:=(JD0+(x-10800)/86400-2451545.0)/36525;
q_l:=2.3555557435+8328.6914257190*T+0.0001545547*sqr(T);
q_c:=6.2400601269+628.3019551714*T-(2.6820e-6)*sqr(T);
Omega_l:=2.1824391966-33.7570459536*T+0.0000362262*sqr(T);
Gamma_:=1.4547885346+71.0176852437*T-0.0001801481*sqr(T);
w_c:=-7.6281824375+0.0300101976*T+(7.9741e-6)*sqr(T);eps:=0.4090926006-0.0002270711*T;
Ksi_:=1-sqr(cos(Omega_l))*(1-cos(i_l));Eta_:=sin(Omega_l)*sin(i_l);
Kapa_:=cos(Omega_l)*sin(i_l);
Ksi11:=sin(Omega_l)*cos(Omega_l)*(1-cos(i_l));Ksi12:=1-sqr(sin(Omega_l))*(1-cos(i_l));
Eta11:=Ksi_*cos(eps)-Kapa_*sin(eps);Eta12:=Ksi11*cos(eps)+Eta_*sin(eps);
Kapa11:=Ksi_*sin(eps)+Kapa_*cos(eps);Kapa12:=Ksi11*sin(eps)-Eta_*cos(eps);
El:=Ekk(q_l,e_l); Ec:=Ekk(q_c,e_c);
sinv_l:=(sqrt(1-sqr(e_l))*sin(El))/(1-e_l*cos(El));
sinv_c:=(sqrt(1-sqr(e_c))*sin(Ec))/(1-e_c*cos(Ec));
cosv_l:=(cos(El)-e_l)/(1-e_l*cos(El));cosv_c:=(cos(Ec)-e_c)/(1-e_c*cos(Ec));
Ksi_l:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Ksi11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Ksi12;
Eta_l:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Eta11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Eta12;
Kapa_l:=(sinv_l*cos(Gamma_)+cosv_l*sin(Gamma_))*Kapa11+(cosv_l*cos(Gamma_)-
sinv_l*sin(Gamma_))*Kapa12;
r_l:=a_l*(1-e_l*cos(El));r_c:=a_c*(1-e_c*cos(Ec));
Ksi_c:=cosv_c*cos(w_c)-sinv_c*sin(w_c);
Eta_c:=(sinv_c*cos(w_c)+cosv_c*sin(w_c))*cos(eps);
Kapa_c:=(sinv_c*cos(w_c)+cosv_c*sin(w_c))*sin(eps);
x_l:=y1/r_l; y_l:=y2/r_l; z_l:=y3/r_l; Gl_:=G_l/sqr(r_l);
x_c:=y1/r_c; y_c:=y2/r_c; z_c:=y3/r_c; Gc_:=G_c/sqr(r_c);
delta_l:=Power(sqr(Ksi_l-x_l)+sqr(Eta_l-y_l)+sqr(Kapa_l-z_l),1.5);
delta_c:=Power(sqr(Ksi_c-x_c)+sqr(Eta_c-y_c)+sqr(Kapa_c-z_c),1.5);
jx0_l:=Gl_*((Ksi_l-x_l)/delta_l-Ksi_l);jy0_l:=Gl_*((Eta_l-y_l)/delta_l-Eta_l);
jz0_l:=Gl_*((Kapa_l-z_l)/delta_l-Kapa_l);jx0_c:=Gc_*((Ksi_c-x_c)/delta_c-Ksi_c);
jy0_c:=Gc_*((Eta_c-y_c)/delta_c-Eta_c);jz0_c:=Gc_*((Kapa_c-z_c)/delta_c-Kapa_c);
end;
function f(eq:integer;x:extended;yy:vectNeq):extended;
var r,GM_,y1_,y2_,y3_,ro:extended;
begincase eq of 1: f:=yy[4]; 2: f:=yy[5]; 3: f:=yy[6]; else
  begin r:=sqrt(yy[1]*yy[1]+yy[2]*yy[2]+yy[3]*yy[3]); GM_:=GM/sqr(r);
   y1_:=yy[1]/r; y2_:=yy[2]/r; y3_:=yy[3]/r; ro:=ae/r;
   ls_acceleration(x,yy[1],yy[2],yy[3],jx0_c,jy0_c,jz0_c,jx0_l,jy0_l,jz0_l);
   case eq of
    4: f:=-GM_*y1_-1.5*J2*GM_*ro*ro*y1_*(1-5*sqr(y3_))+jx0_c+jx0_l;
    5: f:=-GM_*y2_-1.5*J2*GM_*ro*ro*y2_*(1-5*sqr(y3_))+jy0_c+jy0_l;
    6: f:=-GM_*y3_-1.5*J2*GM_*ro*ro*y3_*(3-5*sqr(y3_))+jz0_c+jz0_l;
end; end;end;end;
```

```
procedure RK4(x,h:extended; y:vectNeq; var yr_:vectNeq);
vark: array[1..4] of vectNeq; ytemp:vectNeq; i:byte;
begin for i := 1 to Neq do k[1,i]:=h*f(i,x,y);
 for i := 1 to Neq do ytemp[i]:=y[i]+0.5*k[1,i];
 for i := 1 to Neq do k[2,i]:=h*f(i,x+0.5*h,ytemp);
 for i := 1 to Neq do ytemp[i]:=y[i]+0.5*k[2,i];
 for i := 1 to Neq do k[3,i]:=h*f(i,x+0.5*h,ytemp);
 for i := 1 to Neq do ytemp[i]:=y[i]+k[3,i];
 for i := 1 to Neq do k[4,i]:=h*f(i,x+h,ytemp);
 for i := 1 to Neq do yr_[i]:= y[i] + 1/6 * (k[1,i]+2*k[2,i]+2*k[3,i]+k[4,i]);
end;
begin JD0:=1461*(N4-1)+NT+2450082.5;dT:=(JD0-2451545.0)/36525;
ERA:=2*PI*(0.7790572732640+1.00273781191135448*(JD0-2451545.0));
GMST:=ERA+7.03270726e-8+0.0223603658710194*dT+6.7465784654e-6*power(dT,2)
-2.1332e-12*power(dT,3)-1.452308e-10*power(dT,4)-1.784e-13*power(dT,5);
SW:= TStopwatch.StartNew;SW.Start;S:=GMST+w3*(tb-10800);
y0[1]:=y00[1]*cos(S)-y00[2]*sin(S);y0[2]:=y00[1]*sin(S)+y00[2]*cos(S);
y0[3]:=y00[3];y0[4]:=y00[4]*cos(S)-y00[5]*sin(S)-w3*y0[2];
y0[5]:=y00[4]*sin(S)+y00[5]*cos(S)+w3*y0[1];y0[6]:=y00[6];
ys:=y0; m:=1;x:=tb;
if ti > tb then while x<ti-h/2 do begin RK4(x,h,ys,yr); x:=tb+m*h; inc(m); ys:=yr; end;
elsewhile x>ti+h/2 do begin RK4(x,-h,ys,yr); x:=tb-m*h; inc(m); ys:=yr; end;
S:=GMST+w3*(ti-10800);
yn[1]:=yr[1]*cos(S)+yr[2]*sin(S);yn[2]:=-yr[1]*sin(S)+yr[2]*cos(S);
yn[3]:=yr[3];yn[4]:=yr[4]*cos(S)+yr[5]*sin(S)+w3*yn[2];
yn[5]:=-yr[4]*sin(S)+yr[5]*cos(S)-w3*yn[1];yn[6]:=yr[6];
SW.Stop;write('Calculation time: '); writeln(SW.Elapsed.TotalMilliseconds,' ms');
writeln;writeln('y1=',yn[1],' y2=',yn[2],' y3=',yn[3]);
writeln('y4=',yn[4],' y5=',yn[5],' y6=',yn[6]);writeln;
m:=1;x:=ti;
if ti>tb thenwhile x > tb+h/2 do begin RK4(x,-h,ys,yr); x:=ti-m*h; inc(m); ys:=yr; end
else while x < tb-h/2 do begin RK4(x,h,ys,yr); x:=ti+m*h; inc(m); ys:=yr; end;
for i:=1 to Neq do write(' P',i,' =',Format(' %1.5e',[abs(yr[i]-y0[i])]));
readln; end.
```

Результаты расчета положения и компоненов вектора скорости движения НКА ГЛОНАСС №730 в системе ПЗ-90.11 с момента времени $t_b = 11700$ даты 05.08.2021 ($N_T = 583$, $N_4 = 7$) шкалы МДВ, представлены ниже для моментов времени $t_i = 10800$, $t_i = 11100$, $t_i = 11400$. Расчет выполняется на основе численного приближения решения задачи (6.1), (6.6) с использованием метода Рунге-Кутта 4-го порядка $h = 1c$ (программа *rk4_glonass* (листинг П6.1)):

при $t_i = 11400$:

```
Calculation time:  1.20753000000000E+0001 ms

y1= 2.50732546741584E+0007 y2= 3.70132538988291E+0005 y3=-4.72005283502666E+0006
y4=-6.55064088682716E+0002 y5=-9.49971553422365E+0001 y6=-3.48535231920772E+0003

 P1 = 0,0000E+000 P2 = 9,0949E-013 P3 = 0,0000E+000
 P4 = 0,0000E+000 P5 = 0,0000E+000 P6 = 0,0000E+000
```

при $t_i = 11100$:

```
Calculation time:  2.49930000000000E+0001 ms

y1= 2.52479703502125E+0007 y2= 4.02293478734320E+0005 y3=-3.66972692057345E+0006
y4=-5.09373974562800E+0002 y5=-1.18316598624554E+0002 y6=-3.51555999693191E+0003
```

```
 P1 = 0,0000E+000 P2 = 9,0949E-013 P3 = 0,0000E+000
P4 = 0,0000E+000 P5 = 0,0000E+000 P6 = 0,0000E+000
```

при $t_i = 10800$:

```
Calculation time:  3.64895000000000E+0001 ms

y1= 2.53787058994140E+0007 y2= 4.40461175256494E+0005 y3=-2.61147611825596E+0006
y4=-3.61952522189109E+0002 y5=-1.35023964705675E+0002 y6=-3.53817540705169E+0003

 P1 = 0,0000E+000 P2 = 9,0949E-013 P3 = 4,5475E-013
 P4 = 0,0000E+000 P5 = 0,0000E+000 P6 = 0,0000E+000
```

Таким образом, представленные результаты расчетов для отрезков интегрирования $[t_b - \Delta t,\, t_b]$, $\Delta t \in \{300\,c,\, 600\,c,\, 900\,c\}$ подтверждают выводы, сформулированные в главе 6. Разностное приближение решения с помощью метода Рунге-Кутты 4-го порядка с шагом интегрирования $h = 1c$ также как и приближение, полученное на основе кусочно-интерполяционного метода (см. П6.1), характеризуется точностью, превышающей точность начальных данных (6.6), однако время расчета выше в десятки и сотни раз в сравнении с временем кусочно-интерполяционного приближения.

АКТЫ ОБ ИСПОЛЬЗОВАНИИ РЕЗУЛЬТАТОВ ДИССЕРТАЦИОННОЙ РАБОТЫ

УТВЕРЖДАЮ

Генеральный директор,
Главный конструктор АО НКБ ВС,
кандидат технических наук

_____ С.А. Сивцов

05 2023 г.

А К Т

об использовании результатов диссертационной работы Джанунца Г.А.
в АО «Научно-конструкторское бюро вычислительных систем»

Настоящий акт составлен в том, что в АО «Научно-конструкторское бюро вычислительных систем» используются методы и программы, разработанные Джанунцем Г.А. в диссертационной работе «Методы обработки данных в информационно-вычислительных системах для моделей периодических процессов», представленной на соискание ученой степени доктора технических наук.

1. Предложенный автором диссертационной работы метод варьируемого разностно-полиномиального решения задачи Коши для систем обыкновенных дифференциальных уравнений (ОДУ) с автоматическим выбором параметров, снижающим погрешность, принят к использованию для моделирования движения и автоматического управления подвижными объектами.

2. При расширении библиотеки стандартных программ вычисления элементарных, повторяющихся и специальных функций для бортовых вычислителей систем автоматического управления движением подвижными объектами и с целью вычисления композиций сложных функций с минимальной временной сложностью при высокой точности приближения функций, апробирован предложенный автором метод компьютерной кусочно-полиномиальной аппроксимации функций с вариацией длины подынтервала и степени полинома.

3. Представленные в диссертационной работе метод варьируемого кусочно-интерполяционного решения задачи Коши для систем ОДУ с итерационным уточнением, использующий динамическую коррекцию начальных значений, и его модификация для решения частного случая двухточечной задачи Коши, приняты к использованию для повышения точности численного моделирования автоколебательных процессов при автоматизированном управлении движением объектов.

4. Комплекс программ, представленный в диссертации и в приложении к ней, используется для численного моделирования процессов управления с широко варьируемыми параметрами, что позволяет достигать уточнения физико-технологических и динамических характеристик моделируемых процессов.

Научный руководитель АО НКБ ВС,
кандидат технических наук  И.И. Итенберг

Технический директор АО НКБ ВС,
кандидат технических наук  С.А. Бачило

УТВЕРЖДАЮ

Заместитель Генерального
директора АО «ВНИИЖТ»
доктор технических наук

_____ С.Е. Ададуров

«23» 05 2023г.

А К Т

об использовании результатов диссертационной работы Джанунца Г.А.
в Научно-исследовательском институте железнодорожного транспорта

Настоящий акт составлен в том, что в АО «ВНИИЖТ» приняты к использованию методы, алгоритмы и программы, разработанные Джанунцем Г.А. в диссертационной работе «Методы обработки данных в информационно-вычислительных системах для моделей периодических процессов», представленной на соискание ученой степени доктора технических наук.

1. Представленный в диссертационной работе метод варьируемого кусочно-интерполяционного решения задачи Коши для систем ОДУ с итерационным уточнением, использующий автоматический выбор параметров, принят к использованию с целью повышения точности прогнозирования положения объекта, а также для моделирования навигационного управления.

2. Принят к использованию метод кусочно-интерполяционного приближения функций, производных и интегралов. Метод позволяет вычислять композиции сложных функций с высокой точностью за минимальное время. Метод применим для расширения библиотеки стандартных программ вычисления элементарных, часто повторяющихся и специальных функций, позволяет существенно ускорить процесс численного моделирования, а также уточнить

значения параметров в моделях дистанционного зондирования с помощью спутников и БПЛА. Также принят к использованию предложенный автором метод кусочно-интерполяционного приближения функций двух переменных с вариацией размеров подобласти и степени полинома.

3. Комплекс программ, представленный в диссертации и в приложении к ней, принят к использованию с целью уточнения физико-технологических характеристик моделируемых процессов в реальном времени. Программная реализация предложенных методов в режиме реального времени осуществляется за счет математической минимизации временной сложности численных алгоритмов.

4. Представленный в диссертационной работе программный комплекс для численного моделирования движения навигационных космических аппаратов ГЛОНАСС принят к использованию с целью ускорения и повышения точности расчета координат местоположения и скорости движения исследуемых объектов.

Директор Центра М.А. Галицын

Подпись М.А. Галицына заверяю
Зам. генерального директора по
управлению персоналом и кадр. вопросам
АО «ВНИИЖТ» А.А. Пахалев

УТВЕРЖДАЮ
Проректор по учебной работе
ФГБОУ ВО «РГЭУ (РИНХ)»,
канд. экон. наук, доцент
_____В.Ю. Боев
«19» 05 2023 г.

А К Т
об использовании результатов диссертационной работы Джанунца Г.А.
в учебном процессе кафедры информатики
ТИ имени А.П. Чехова (филиала) ФГБОУ ВО «РГЭУ (РИНХ)»

Настоящий акт составлен в том, что в учебном процессе кафедры информатики ТИ имени А.П. Чехова (филиала) ФГБОУ ВО «РГЭУ (РИНХ)» используются следующие материалы диссертационной работы Джанунца Г.А. «Методы обработки данных в информационно-вычислительных системах для моделей периодических процессов», представленной на соискание ученой степени доктора технических наук.

1. Описанный в первой главе диссертации анализ современного состояния проблем обработки данных моделей периодических процессов в информационно-вычислительных системах (ИВС) наряду с описанными в пятой и шестой главах диссертации результатами моделирования периодических процессов с применением кусочно-интерполяционной обработки данных используются в качестве учебного материала в лекционных курсах и при проведении лабораторных работ по дисциплинам «Компьютерное моделирование», «Математическое и имитационное моделирование», «Математическое моделирование и численные эксперименты», «Алгоритмы численного интегрирования и анализа устойчивости», «Численные методы в анализе данных», «Визуализация данных», «Специальные разделы информатики», «Современные инструментальные средства».

2. Изложенные во второй, третьей и четвертой главах диссертации метод варьируемой разностно-полиномиальной обработки данных в ИВС с автоматическим выбором параметров для моделей периодических процессов, метод кусочно-интерполяционной обработки данных с итерационным уточнением для моделей периодических процессов и метод варьируемой кусочно-интерполяционной обработки данных для модели переноса с итерационным уточнением соответственно используются в курсах «Численные методы в анализе данных», «Программирование», «Современные методы построения программ», «Алгоритмы численного интегрирования и анализа устойчивости», «Абстрактная и компьютерная алгебра», «Параллельные алгоритмы» в качестве лекционного учебного материала и материала для лабораторных работ, а также в качестве тематики курсовых и выпускных квалификационных работ факультета физики, математики, информатики.

3. Представленный в диссертации и в приложении к ней комплекс программ обработки данных в ИВС на основе кусочно-интерполяционного метода для моделей жестких и нежестких задач с автоматическим и пользовательским выбором параметров для адаптации к классам моделей, включающий также программы моделирования процессов переноса и программы обработки данных эфемерид на модели движения навигационного космического аппарата ГЛОНАСС используется в курсах «Абстрактная и компьютерная алгебра», «Численные методы», «Алгоритмы численного интегрирования и анализа устойчивости», «Программирование», «Компьютерное моделирование», «Современные технологии программирования» в качестве учебного материала и в качестве тематики курсовых и выпускных квалификационных работ факультета физики, математики, информатики.

Директор ТИ имени А.П. Чехова
доктор политических наук,
кандидат филологических наук, доцент А.Ю. Голобородько