

**Ю.В.ВАСИЛЬКОВ, Н.Н.ВАСИЛЬКОВА**

---

**КОМПЬЮТЕРНЫЕ**

---

**ТЕХНОЛОГИИ**

---

**ВЫЧИСЛЕНИЙ**

---

**в математическом  
моделировании**

Рекомендовано  
Министерством образования  
Российской Федерации  
в качестве учебного пособия  
для студентов  
высших учебных заведений,  
обучающихся по экономическим специальностям



Москва  
“Финансы и статистика”  
2002

УДК 004.942:519.86 (075)  
ББК 22.19с51я7  
В19

*РЕЦЕНЗЕНТЫ:*

Кафедра математического моделирования экономических процессов  
Финансовой академии при Правительстве Российской Федерации;  
А. Н. Канатников, кандидат физико-математических наук, доцент

**Васильков Ю. В., Василькова Н. Н.**

В19 Компьютерные технологии вычислений в математическом моделировании: Учеб. пособие. – М.: Финансы и статистика, 2002. – 256 с.: ил.

ISBN 5-279-02098-2.

Пособие содержит общие сведения об особенностях математического моделирования и теоретические основы вычислительных методов как его инструментов. Рассмотрены методы обработки данных: интерполяция, аппроксимация, решение алгебраических и дифференциальных уравнений и их систем, вычисление интегралов, методы оптимизации. Показаны способы реализации алгоритмов на Visual Basic для Excel 7.0. Даны характеристики наиболее распространенных программных средств для проведения вычислительных работ. Приведены контрольные вопросы к каждой теме и ответы на них.

Для самостоятельной работы студентов вузов очной и заочной форм обучения по экономическим специальностям, а также учащихся лицеев и гимназий.

В  $\frac{240400000 - 176}{010(01) - 2002}$  230 – 2001

УДК 004.942:519.86(075)  
ББК 22.19с51я7

ISBN 5-279-02098-2

© Ю. В. Васильков, Н. Н. Василькова, 1999

## ПРЕДИСЛОВИЕ

---

Математизация различных областей знаний в настоящее время не является чем-то новым, неожиданным. Широкое внедрение математических методов в самые разнообразные сферы деятельности сегодня уже никого не удивляет. Это не только технические и экономические науки, где эти методы давно приносят свои плоды, но и развивающиеся сейчас разнообразные прикладные науки управления: менеджмент, принятие управляющих решений, социально-экономическое прогнозирование и т. д.

Прикладные науки развиваются своим путем, используя существующий математический аппарат для решения возникающих проблем, и даже своими потребностями стимулируют развитие некоторых разделов математики. Но в них нередко царят своя терминология, свои частные приемы решения задач, свои исходные предпосылки и цели. Имеют место ситуации, когда некорректно примененные прикладниками методы тем не менее позволяют получать полезные практические результаты. Дисциплина “Математическое моделирование” давно сформировалась как прикладная наука и включена в подготовку специалистов почти по всем экономическим и техническим направлениям.

Данная книга задумывалась и писалась на основе собственного педагогического опыта авторов как пособие для самостоятельной работы студентов. Поэтому в книгу из методических соображений вошло не все, что есть в науке вычислений, а лишь то, что нужно для предполагаемого круга читателей, и в такой форме, ко-

торая нужна этим читателям с учетом их потребностей, обусловленных решаемыми задачами в своей профессии. Книга не заменяет существующих учебных пособий академического плана, которые посвящены математическим аспектам вычислительных методов, у нее другая задача. Авторы видят основную задачу в знакомстве читателей определенного круга, прежде всего студентов-нематематиков, с вычислительными методами как инструментом решения задач, встречающихся в их профессиональной деятельности, в частности при использовании методов математического моделирования. Главные вопросы, возникающие у практика: “На чем основан метод, как применить его, что он может дать для решения той или иной его задачи?” Главные же вопросы, возникающие у математика — создателя методов: “С какой скоростью сходится метод, как метод работает с теми или иными функциями, как “доказать все и вся” в методах”? — обычно не затрагивают чувства и мысли пользователя. Например, все мы успешно пользуемся формулой вычисления длины окружности, но не все знают, как она выведена и т.п. Так и многие вычислительные методы у определенного круга пользователей превращаются в рабочий инструмент, усиливающийся в последние годы автоматизацией их применения с помощью персональных компьютеров.

Поэтому в книге отсутствуют некоторые “ типовые ” для прикладной вычислительной математики разделы, например общая теория погрешностей вычислений, детальные теоретические основы методов, теоретический анализ условий существования решения, сходимости метода и т.п. Эти вопросы затрагиваются в необходимом объеме при рассмотрении того или иного метода или группы методов, так как специалисту при решении своей задачи нужно уметь получать результат с заданной погрешностью. Для него исходными данными является содержательная задача, и к ней он подбирает более или менее эффективный метод решения. С точки зрения математика (даже прикладного), первоисточником деятельности является метод, и уже к нему он подбирает ту

или иную функцию, исследуя влияние особенностей этой функции на эффективность метода.

Большие усилия математиков направлены на разработку численных алгоритмов и их исследование. В качестве одного из главных критериев принимается сходимость алгоритма: чем быстрее сходится алгоритм, тем лучше. С точки зрения вычислителя-практика, все выглядит несколько в ином свете. Важную роль играет время решения задачи, удобство обращения к алгоритму и многое другое. Становится очевидным, что решение больших задач требует неформальных действий вычислителя, возможности вмешиваться в процесс счета — так называемого диалогового режима и т.д. Поэтому сходимость алгоритма — это лишь один из многих аспектов численных методов для вычислителя-практика, т.е. для профессионала в области нематематических интересов. Вот почему вопросы сходимости алгоритмов в книге почти не рассматриваются. Далее в книге понятие сходимость характеризуется в практическом смысле числом шагов до достижения искомого результата.

*“Математика не случайно сделалась эталоном мышления. Этим она обязана представлению о строгости, которое выработывалось веками и, конечно, как-то все время деформировалось под натиском нового материала и расширения круга своих задач... Но все имеет свои разумные пределы. Интуиция, опыт — все то, что обычно называется здравым смыслом или неформальным мышлением, в такой же мере имеют законное право на существование при анализе математических задач, как и все прочее”<sup>1</sup>.*

Все вышесказанное означает, что в книге принят “физический уровень строгости”, т.е. та разумная степень глубины анализа метода, которая необходима для его использования специалистом в его практических задачах. Этот выбор стиля изложения

---

<sup>1</sup> Моисеев Н.Н., Иванилов Ю.П., Столярова Е.М. Методы оптимизации. — М.: Наука, 1978.

материала сразу делает очевидным круг читателей, на который ориентируются авторы. Своими читателями мы видим прежде всего студентов экономических вузов, как очной, так и заочной (дистанционной) формы обучения, т.е. тех, для которых прикладная математика не “родной и любимый” предмет, а необходимый инструмент для своей профессиональной деятельности.

Книга может быть полезна также и студентам технических специальностей, несмотря на то, что им для решения отдельных проблем могут потребоваться и более глубокие знания.

В книге содержится большое количество вопросов и ответы на них, которые также будут полезны для самоконтроля усвоения материала.

Авторы выражают огромную благодарность академику д-ру техн. наук, проф. С.В. Черемных за участие в формировании структуры книги и канд. физ.-мат. наук, доц. А.Н. Канатникову за коррекцию книги с точки зрения математической строгости изложения.

## ВВЕДЕНИЕ

---

Математическое моделирование как инструмент познания завоевывает все новые и новые позиции в различных областях деятельности человека. Оно становится главенствующим направлением в проектировании и исследовании новых систем, анализе свойств существующих систем, выборе и обосновании оптимальных условий их функционирования и т.п.

Математическое моделирование широко проникло в различные области знаний и их приложения: технические, экономические, социальные, биологические и многие другие, на первый взгляд, далекие от математики. Поэтому специалистам различных направлений необходимо владеть концепциями и методами математического моделирования, иметь представление об инструментари, применяемом при моделировании.

Первый и главный этап математического моделирования — собственно построение модели — очень часто опирается на некоторые имеющиеся исходные данные. При этом широко применяются вычислительные методы обработки данных: методы интерполяции, аппроксимации и др.

Основная задача моделирования различного рода процессов и систем с целью исследования объектов, прогнозирования их поведения или поиска наилучших условий функционирования сводится к расчету анализируемых показателей по математической модели при тех или иных значениях (или функциях) входных величин. Важное значение при этом приобретают вычислительные алгоритмы, с помощью которых можно получить при моделировании решение конкретной математической задачи.

Знакомству с идеями и алгоритмами решения наиболее распространенных задач вычислительной математики, применяющихся при математическом моделировании, получению практических навыков их применения и посвящено данное учебное пособие. Оно включает в себя следующие основные темы.

- Интерполяция.
- Аппроксимация.
- Решение нелинейных уравнений и их систем.
- Решение систем линейных уравнений.
- Вычисление интегралов.
- Основы решения дифференциальных уравнений.
- Методы оптимизации.

Эти темы охватывают широкий спектр методов и являются минимумом, необходимым для дальнейшего успешного решения различных задач математического моделирования, возникающих при исследовании реальных объектов промышленного производства, экономических, финансовых и других и управления ими. Все рассмотренные методы снабжены достаточно подробными примерами реализации вычислительных алгоритмов.

Пособие содержит также контрольные вопросы по всем темам и ответы на них, которые представляют специфическую форму самообразования. Причем вопросы составлены с расчетом на некоторый активный опыт исследования вычислительных методов, что является необходимым условием самостоятельной работы с книгой.

Приведенные в книге ответы помогут лучше понять те особенности методов, которые не очевидны из теоретического материала, “не лежат на поверхности”, а выявляются в результате приобретения опыта их применения.

Определенное внимание в книге уделено также реализации вычислительных алгоритмов на ЭВМ, т.е. технологии вычислительных работ с опорой на современные технические и программные средства.

Пособие в теоретическом плане не заменяет имеющуюся литературу по численным методам. В отличие от академической учебной литературы, где рассмотрены вопросы истории развития

методов, доказательства и обоснования различных положений и выводов, большое число методов и их модификаций, в данном пособии изложены концепции и основные идеи, на которых базируются вычислительные методы математического моделирования и даны практические примеры их применения. Оно не является полным, всеобъемлющим, а содержит лишь самые необходимые методы и сведения о них и призвано сформировать концептуальное понимание вычислительных методов для решения различных задач, закрепить и систематизировать знания, полученные из учебников, в соответствии с программой соответствующего учебного курса. Данное пособие ориентировано прежде всего на самостоятельную работу студентов.

## **ОСНОВЫ МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ**

---

### **КОНЦЕПЦИЯ МОДЕЛИРОВАНИЯ**

Моделирование в научных исследованиях стало применяться еще в глубокой древности и постепенно захватывало все новые области научных знаний: техническое конструирование, строительство и архитектуру, астрономию, физику, химию, биологию и, наконец, общественные науки. Большие успехи и признание практически во всех отраслях современной науки принес моделированию XX век. Однако методология моделирования долгое время развивалась независимо в различных областях приложения. Отсутствовала единая система понятий, единая терминология. Лишь постепенно стала осознаваться роль моделирования как универсального метода научного познания.

*Модель* — это такой материальный или мысленно представляемый объект, который в процессе исследования замещает объект-оригинал так, что его непосредственное изучение дает новые знания об объекте-оригинале.

Под *моделированием* понимается триединый процесс построения, изучения и применения моделей. Моделирование тесно

связано с такими категориями, как абстракция, аналогия, гипотеза и др. Процесс моделирования обязательно включает и построение абстракций, и умозаключения по аналогии, и конструирование научных гипотез. Модель выступает как своеобразный инструмент познания, который исследователь ставит между собой и объектом и с помощью которого изучает интересующий его объект. Именно эта особенность метода моделирования определяет специфические формы использования абстракций, аналогий, гипотез, других категорий и методов познания.

Необходимость использования метода моделирования определяется тем, что многие объекты (или проблемы, относящиеся к этим объектам) непосредственно исследовать или вовсе невозможно, или же это исследование требует много времени и средств.

Пусть имеется или необходимо создать некоторый объект **A**. Мы конструируем (материально или мысленно) или находим в реальном мире другой объект **B** — модель объекта **A**. Процесс построения модели предполагает наличие некоторых знаний об объекте-оригинале. Познавательные возможности модели обуславливаются тем, что модель отражает какие-либо существенные черты объекта-оригинала. Вопрос о необходимости и достаточной мере сходства оригинала и модели требует конкретного анализа. Очевидно, модель утрачивает свой смысл как в случае тождества с оригиналом (тогда она перестает быть моделью), так и в случае чрезмерного во всех существенных отношениях отличия от оригинала.

Таким образом, изучение одних сторон моделируемого объекта осуществляется ценой отказа от отражения других сторон. Поэтому любая модель замещает оригинал лишь в строго ограниченном смысле. Из этого следует, что для одного объекта может быть построено несколько “специализированных” моделей, концентрирующих внимание на определенных сторонах исследуемого объекта или же характеризующих объект с разной степенью детализации.

В процессе изучения свойств объекта при моделировании модель выступает как самостоятельный объект исследования. Одной из форм такого исследования является проведение “модельных” экспериментов, при которых сознательно изменяются условия функционирования модели и систематизируются данные о ее “поведении”. Конечным результатом этого этапа является множество знаний о модели.

В процессе применения моделей осуществляется перенос знаний с модели на оригинал — формирование множества знаний об объекте. Этот процесс переноса знаний проводится по определенным правилам. Знания о модели должны быть скорректированы с учетом тех свойств объекта-оригинала, которые не нашли отражения или были изменены при построении модели. Мы можем с достаточным основанием переносить какой-либо результат с модели на оригинал, если этот результат связан с признаками сходства оригинала и модели.

Если же определенный результат модельного исследования связан с отличием модели от оригинала, то этот результат переносить неправомерно.

Здесь же происходит практическая проверка получаемых с помощью моделей знаний, т.е. проверка адекватности модели и их использования для построения обобщающей теории объекта, его преобразования или управления им.

Существуют по крайней мере две точки зрения на результаты моделирования. Одна отталкивается от того, что при синтезе модели в нее закладываются такие связи, соотношения, которые уже известны исследователю (естественно, что неизвестные заложить в модель нельзя). Поэтому из модели нельзя получить новых знаний об объекте. В этом случае модель может выступать только как расчетный объект, на котором можно проводить численные эксперименты, в том числе в таких ситуациях, в которых сам объект или не существовал, или не существует. Вторая точка зрения на результаты моделирования исходит из того, что при конструировании в модель закладываются известные сведения (связи, соотношения) об элементах объекта, но они в соответствии со спецификой сложной системы могут в совокупности проявить

качественно новые свойства, не присущие отдельным элементам. В этом случае математическое моделирование способно дать новые, до сих пор неизвестные знания об объекте. Второй подход к результатам моделирования более оптимистичен по сравнению с первым, он не ограничивает мысль исследователя, не сковывает его какими-то рамками, сохраняет надежду на новые знания, на научный и практический прогресс, наполняет процесс моделирования более глубоким смыслом.

Для понимания сущности моделирования важно не упускать из виду, что моделирование — не единственный источник знаний об объекте. Процесс моделирования “погружен” в более общий процесс познания. Это обстоятельство учитывается не только непосредственно при построении модели, но и при завершении моделирования, когда происходит объединение и обобщение результатов исследования, получаемых на основе многообразных средств познания.

Моделирование — циклический процесс. Это означает, что за первым трехуровневым циклом может последовать второй, третий и т.д. При этом знания об исследуемом объекте расширяются и уточняются, а исходная модель постепенно совершенствуется. Недостатки, обнаруженные после первого цикла моделирования, обусловленные малым знанием объекта и ошибками в построении модели, можно исправить в последующих циклах. В методологии моделирования, таким образом, заложены большие возможности саморазвития.

Проникновение математики в экономическую науку связано с преодолением значительных трудностей. Главные из них заключаются в природе экономических процессов, в специфике экономической науки. Большинство объектов, изучаемых экономической наукой, может быть охарактеризовано кибернетическим понятием “сложная система”. Наиболее распространено понимание системы как совокупности элементов, находящихся во взаимодействии и образующих некоторую целостность, единство. Важным качеством любой системы является наличие таких свойств, которые не присущи ни одному из элементов, входящих

в систему. Поэтому при изучении систем недостаточно просто пользоваться методом их расчленения на элементы с последующим изучением этих элементов в отдельности. Одна из трудностей экономических исследований в том, что почти не существует экономических объектов, которые можно было бы рассматривать как отдельные (внесистемные) элементы.

Сложность системы любой природы (технической, экономической, биологической, социальной и т.д.) определяется количеством входящих в нее элементов, связями между этими элементами, а также взаимоотношениями между системой и средой. Экономика обладает всеми признаками очень сложной системы. Она объединяет огромное число элементов, отличается многообразием внутренних связей и связей с другими системами (природной средой, экономической деятельностью других субъектов, социальными отношениями и т.д.). В народном хозяйстве взаимодействуют природные, технологические, социальные процессы, объективные и субъективные факторы.

Сложностью экономических отношений нередко обосновывали невозможность моделирования экономики, изучения ее средствами математики. Моделировать можно объект любой природы и любой сложности. Сложные объекты представляют наибольший интерес для моделирования; именно здесь моделирование может дать результаты, которые нельзя получить другими методами исследования.

Потенциальная возможность математического моделирования любых экономических объектов и процессов не означает, разумеется, ее успешной осуществимости при данном уровне экономических и математических знаний, имеющейся конкретной информации и вычислительной техники. И хотя нельзя указать абсолютные границы математической формализуемости экономических проблем, всегда будут существовать еще неформализованные проблемы, а также ситуации, где математическое моделирование недостаточно эффективно.

В различных отраслях знаний этапы процесса моделирования приобретают свои специфические черты. Но во всех случаях

можно выделить несколько этапов, присущих в той или иной мере процессу моделирования в любой сфере. Приведенные ниже этапы охватывают в целом процесс моделирования, без разделения его на три упомянутые выше составные части.

*1. Постановка проблемы и ее качественный анализ.* Главное здесь — четко сформулировать сущность проблемы, принимаемые допущения и те вопросы, на которые требуется получить ответы. Этот этап включает выделение важнейших черт и свойств моделируемого объекта и абстрагирование от второстепенных; изучение структуры объекта и основных зависимостей, связывающих его элементы; формулирование гипотез (хотя бы предварительных), объясняющих поведение и развитие объекта.

*2. Построение математической модели.* Это — этап формализации проблемы, выражения ее в виде конкретных математических зависимостей и отношений (функций, уравнений, неравенств и т.д.). Обычно сначала определяется (или задается в случае применения формальных моделей) основная конструкция (тип) математической модели, а затем уточняются детали этой конструкции (конкретный перечень переменных и параметров, форма связей). Таким образом, построение модели подразделяется в свою очередь на несколько стадий.

Неправильно полагать, что чем больше факторов (т.е. входных и выходных переменных состояния) учитывает модель, тем она лучше “работает” и дает лучшие результаты. То же можно сказать о таких характеристиках сложности модели, как используемые формы математических зависимостей (линейные и нелинейные), учет факторов случайности и неопределенности и т.д. Излишняя сложность и громоздкость модели затрудняют процесс исследования. Нужно не только учитывать реальные возможности информационного и математического обеспечения, но и сопоставлять затраты на моделирование с получаемым эффектом (при возрастании сложности модели нередко рост затрат на моделирование может превысить рост эффекта от внедрения моделей в задачи управления). Естественно, необходимо стремиться к тому, чтобы получить модель, принадлежащую хорошо изученному

классу математических задач, пути и методы решения которых известны и хорошо разработаны. Часто это удается сделать путем некоторого упрощения исходных предпосылок модели, не искажающих существенных черт моделируемого объекта. Однако возможна и такая ситуация, когда формализация проблемы приводит к неизвестной ранее математической структуре, в этом случае актуальность приобретают вычислительные методы, с помощью которых можно исследовать модель и ее свойства (в конечном счете — свойства исходного объекта).

*3. Математический анализ модели.* Целью этого этапа является выяснение общих свойств модели. Здесь применяются чисто математические приемы исследования. Наиболее важный момент — доказательство существования решений в сформулированной модели (теорема существования). Если удастся доказать, что математическая задача не имеет решения, то необходимость в последующей работе по первоначальному варианту модели отпадает; следует скорректировать либо постановку задачи, либо способы ее математической формализации. При аналитическом исследовании модели выясняются такие вопросы, как, например, единственно ли решение, какие переменные могут входить в решение, каковы будут соотношения между ними, в каких пределах и в зависимости от каких исходных условий они изменяются, каковы тенденции их изменения и т.д.

Модели сложных объектов с большим трудом поддаются аналитическому исследованию. В тех случаях, когда аналитическими методами не удастся выяснить общих свойств модели, а упрощения модели приводят к недопустимым результатам, связанным с потерей ее адекватности, переходят к численным методам исследования.

*4. Подготовка исходной информации.* Моделирование предъявляет жесткие требования к системе информации. В процессе подготовки информации широко используются методы теории вероятностей, теоретической и математической статистики. При системном математическом моделировании исходная информа-

ция, используемая в одних моделях, является результатом функционирования других моделей.

5. *Численное решение.* Этот этап включает разработку алгоритмов для численного решения задачи, составления программ на ЭВМ и непосредственное проведение расчетов. Здесь приобретают актуальность различные методы обработки данных, решения разнообразных уравнений, вычисления интегралов и т.п. Нередко расчеты по математической модели носят многовариантный, имитационный характер. Благодаря высокому быстродействию современных ЭВМ удается проводить многочисленные “модельные” эксперименты, изучая “поведение” модели при различных изменениях некоторых условий. Для решения таких задач важное значение имеют методы оптимизации, т.е. поиска наилучших (экстремальных) значений каких-либо функций и функционалов. Исследование, проводимое численными методами, может существенно дополнить результаты аналитического исследования, а для многих моделей оно является единственно осуществимым. Класс задач, которые можно решать численными методами, значительно шире, чем класс задач, доступных аналитическим методами.

6. *Анализ численных результатов и их применение.* На этом заключительном этапе цикла встает вопрос о правильности и полноте результатов моделирования, об адекватности модели, о степени ее практической применимости. Математические методы проверки результатов могут выявлять некорректность построения модели и тем самым сужать класс потенциально правильных моделей. Неформальный анализ теоретических выводов и численных результатов, получаемых посредством модели, сопоставление их с имеющимися знаниями и фактами действительности также позволяют обнаруживать недостатки исходной постановки задачи, сконструированной математической модели, ее информационного и математического обеспечения.

Поскольку современные математические задачи могут быть сложны по своей структуре, иметь большую размерность, то часто случается, что известные алгоритмы и программы для ЭВМ не

позволяют решить задачу в первоначальном виде. Если невозможно в короткий срок разработать новые алгоритмы и программы, исходную постановку задачи и модель упрощают: снимают и объединяют условия, уменьшают число учитываемых факторов, нелинейные соотношения заменяют линейными т.д.

Недостатки, которые не удается исправить на промежуточных этапах моделирования, устраняются в последующих циклах. Но результаты каждого цикла имеют и вполне самостоятельное значение. Начав исследование с построения простой модели, можно быстро получить полезные результаты, а затем перейти к созданию более совершенной модели, дополняемой новыми условиями, включающей уточненные математические зависимости.

Теория математического анализа моделей экономики развилась в особую ветвь современной математики — *математическую экономику*. Модели, изучаемые в рамках математической экономики, теряют непосредственную связь с экономической реальностью; они имеют дело с исключительно идеализированными экономическими объектами и ситуациями. При построении таких моделей главным принципом является не столько приближение к реальности, сколько получение возможно большего числа аналитических результатов посредством математических доказательств. Ценность этих моделей для экономической теории и практики состоит в том, что они служат теоретической базой для моделей прикладного типа.

Можно выделить по крайней мере четыре аспекта применения математических методов в решении практических проблем.

*1. Совершенствование системы сбора информации о сложном объекте.* Математические методы позволяют упорядочить систему информации, выявлять недостатки в имеющейся информации и вырабатывать требования для подготовки новой информации или ее корректировки. Разработка и применение математических моделей указывают пути совершенствования системы сбора и анализа информации, ориентированной на решение определенных задач планирования и управления. Прогресс в информационном обеспечении планирования и управления опирается

на бурно развивающиеся технические и программные средства информатики.

2. *Интенсификация и повышение точности технических и экономических расчетов.* Формализация проектных технических и экономических задач и применение ЭВМ многократно ускоряют типовые, массовые расчеты, повышают точность и сокращают трудоемкость, позволяют проводить многовариантные технические и экономические обоснования сложных мероприятий, недоступные при “ручной” технологии.

3. *Углубление количественного анализа проблем в технических, экономических и других приложениях.* Благодаря применению метода моделирования значительно усиливаются возможности конкретного количественного анализа: изучение многих факторов, оказывающих влияние на процессы, количественная оценка последствий изменения условий развития объектов и т.п.

4. *Решение принципиально новых научных и практических задач в любой сфере приложений.* Посредством математического моделирования удастся решать такие задачи, которые иными средствами решить невозможно, например: нахождение оптимального варианта выпуска продукции, создание объекта любой природы с заранее заданными свойствами, автоматизация контроля за функционированием сложных технико-экономических объектов и т.п.

## ПРИМЕРЫ ЗАДАЧ МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ

Рассмотрим некоторые встречающиеся в практической деятельности задачи математического моделирования в содержательной, а не в формальной математической трактовке.

**Трендовые модели.** При математическом моделировании экономических процессов с целью определения параметров динамики абсолютных приростов строят трендовые модели различной структуры, отражающие такие особенности, как постоянный рост (или снижение), увеличивающийся рост (снижение), уменьшающийся рост (снижение) и т.д. При этом на основании имеющихся

фактических данных методами аппроксимации подбирают наилучшую модель, параметры которой интерпретируются как те или иные показатели экономического развития. Например, случай развития с увеличивающимся абсолютным приростом описывается показательной или экспоненциальной функциями:

$$X(t) = a(1 + b)^t, \quad (1) \quad X(t) = ae^{bt}, \quad (2)$$

где  $a$  — теоретический начальный уровень,  $b$  — параметр, определяющий постоянный темп прироста. Для анализа динамики этого типа развития часто используется обобщенная экспоненциальная функция, исходящая из предположения переменных темпов прироста  $\rho(t) = x(t)^{-1} dx / dt$ . Путем интегрирования этого выражения получаем

$$X_t = X_0 \exp \int_0^t \rho(t) dt. \quad (3)$$

В качестве функций  $\rho(t)$  могут использоваться линейная, экспоненциальная, параболическая и другие функции.

Характерным свойством трендовых моделей, описывающих рост с качественным изменением характеристик на протяжении рассматриваемого периода, является наличие точки перегиба  $t^*$ , в которой абсолютное ускорение равно нулю и меняет свой знак. Нахождение этой точки возможно путем решения нелинейного уравнения, получающегося приравниванием к нулю второй производной от функции роста по времени. Эта процедура является достаточно сложной, так как для описания функций роста применяют в этом случае такие функции, как линейно-логарифмическая второго порядка, логистическая, первая функция Торнквиста и др.

При построении и использовании трендовых моделей очевидной является необходимость знания методов аппроксимации, интерполяции, численного интегрирования, решения нелинейных уравнений, так как результаты решения той или иной задачи и трудоемкость их получения могут существенно зависеть от выбо-

ра того или иного математического метода и корректности его применения.

### Построение и использование расчетных зависимостей.

Один из видов анализа доходов населения с точки зрения их дифференциации состоит в расчете накопленных частот (долей) и построении кривой Лоренца. Неравенство доходов характеризу-

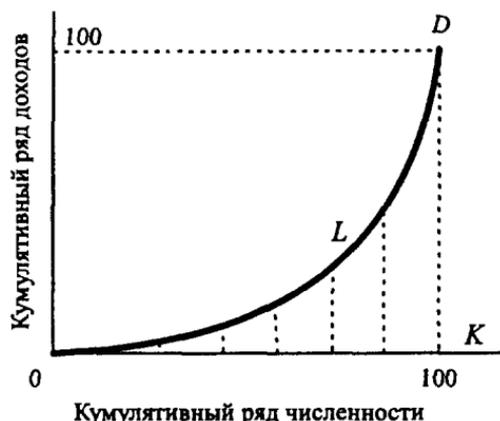


Рис. 1. Степень отклонения кривой Лоренца от биссектрисы координатного угла

ется степенью отклонения кривой Лоренца от биссектрисы координатного угла (рис. 1). Отклонение можно измерить через отношение площади криволинейной фигуры  $ODL$  к площади  $\triangle ODK$ . Эта величина называется коэффициентом концентрации, или коэффициентом Джинни  $G$ . Чем ближе  $G$  к 1, тем выше дифференциация, а чем ближе  $G$  к 0, тем равномернее доходы.

Обычно на кривой, получаемой по данным статистических исследований, содержится немного точек, и поэтому искомая площадь находится приближенно. Точность вычисления коэффициента Джинни можно существенно повысить, применяя эффективные методы численного интегрирования, например методы Чебышева, Гаусса, которые даже при малом числе точек обеспечивают малую погрешность интегрирования. В ряде практических задач требуется найти значения кумулятивного ряда доходов при значениях численности, для которых нет статистических данных. Это легко можно сделать, используя методы интерполяции или аппроксимации.

Аналогичные задачи возникают и при работе с функциями “спрос — предложение”, “расходы — доходы”, полезности, эластичности замещения ресурсов и т.п. Помимо подбора наилучшей функции для описания той или иной зависимости, для чего

применяются методы аппроксимации, часто требуется решать задачи расчета одного показателя при заданном другом, определения наилучшего сочетания параметров при выполнении каких-либо условий (ограничений). Например, нужно найти максимум полезности при ограничении на располагаемый доход. При решении такой задачи по одному продукту можно воспользоваться графическим методом, нанеся на карту безразличия линии различного уровня полезности и бюджетную линию, но при учете нескольких видов продукции, на приобретение которых расходуется доход, необходимо использовать математические методы оптимизации с ограничениями.

**Задачи динамики.** Существует большой класс задач математического моделирования, при описании которых используются дифференциальные уравнения и их системы. Это и простейшая модель воспроизводства национального дохода, и модель динамики фондовооруженности в расширенной модели воспроизводства, и динамические модели В. Леонтьева, Л.В. Канторовича, Дж. фон Неймана и др.; к этому типу моделей относится большое число моделей чисто технического характера (модели движения различных объектов, модели управляемых технологических и производственных объектов и др.). Модели в экономике описываются дифференциальными уравнениями в обычных производных, а в технике нередко и в частных производных, которые решаются значительно сложнее, чем в обычных. Для решения задач прогноза поведения моделируемой системы, задач оптимального управления требуется знание особенностей и возможностей различных методов решения дифференциальных уравнений.

**Оптимизационные задачи.** Рассмотрим некоторые задачи, в которых требуется найти наилучшее с какой-то точки зрения решение; при этом, как правило, существуют различные ограничения на область изменения управляющих переменных, что не позволяет воспользоваться методами классического математического анализа, а требует применения разнообразных вычислительных методов.

**Задачи оптимального распределения ресурсов.** В общем виде эти задачи могут быть описаны следующим образом. Имеется некоторое количество ресурсов, под которыми можно понимать денежные средства, материальные ресурсы (например, сырье, полуфабрикаты, трудовые ресурсы, различные виды оборудования и т.д.). Эти ресурсы необходимо распределить между различными объектами их использования по отдельным промежуткам времени или по различным объектам так, чтобы получить максимальную суммарную эффективность от выбранного способа распределения. Показателем эффективности может служить, например, прибыль, товарная продукция, фондоотдача (задачи максимизации критерия оптимальности) или суммарные затраты, себестоимость, время выполнения данного объема работ и т. п. (задачи минимизации критерия оптимальности).

Имеется начальное количество средств  $P_0$ , которое необходимо распределить в течение  $n$  лет между  $s$  предприятиями. Средства  $u_{ki}$  ( $k = 1, \dots, n; i = 1, \dots, s$ ), выделенные в  $k$ -м году  $i$ -му предприятию, приносят доход в размере  $f_{ki}(u_{ki})$  и к концу года возвращаются в количестве  $\varphi_{ki}(u_{ki})$ . В последующем распределении доход может либо участвовать (частично или полностью), либо не участвовать.

Требуется определить такой способ распределения ресурсов (количество средств, выделяемых каждому предприятию в каждом плановом году), чтобы суммарный доход от  $s$  предприятий за  $n$  лет был максимальным. Следовательно, в качестве показателя эффективности процесса распределения ресурсов за  $n$  лет принимается суммарный доход, полученный от  $s$  предприятий:

$$Z = \sum_{i=1}^s \sum_{k=1}^n f_{ki}(u_{ki}). \quad (4)$$

Количество ресурсов в начале  $k$ -го года будем характеризовать величиной  $P_{n-1}$  (параметр состояния). Управление на  $k$ -м шаге состоит в выборе переменных  $u_{k1}, u_{k2}, \dots, u_{ks}$ , обозначающих ресурсы, выделяемые в  $k$ -м год  $i$ -му предприятию.

Если предположить, что доход в дальнейшем распределении не участвует, то уравнение состояния процесса имеет вид

$$P_k = P_{k-1} - \sum_{i=1}^s u_{ki} + \sum_{i=1}^s \varphi_{ki}(u_{ki}). \quad (5)$$

Если же некоторая часть дохода участвует в дальнейшем распределении в каком-нибудь году, то к правой части последнего равенства прибавляется соответствующая величина.

Требуется определить  $n_s$  неотрицательных переменных  $u_{ki}$ , удовлетворяющих условиям (5) и максимизирующих функцию (4).

**Оптимальное управление запасами.** Класс задач, в которых рассматривается оптимальное управление запасами, является одним из наиболее сложных. Это обусловлено тем, что в задачах управления запасами процесс, естественно, разворачивается во времени, причем управление заключается в том, что решение на данном промежутке времени принимается с учетом того состояния, к которому пришла система за предшествующие периоды. Кроме того, эти задачи связаны, как правило, с дискретным характером переменных и, следовательно, решаются довольно сложно.

Проблема управления запасами является одной из важнейших областей практического приложения экономико-математических методов, в том числе методов математического программирования.

При формулировке задач управления запасами используют следующие понятия.

**Запасы** — это любые денежные или материальные ценности, которые периодически пополняются (производятся, доставляются и т. д.) и некоторое время сохраняются с целью расходования их в последующие промежутки времени. Уровень запасов в любой момент времени определяется начальным уровнем запасов плюс пополнение и минус расход за промежуток времени от начального момента до текущего.

Управление запасами в общем случае состоит в воздействии на соотношение между двумя основными факторами — пополнением и расходом. Цель управления — оптимизация некоторого

критерия, зависящего от расходов на хранение запасов, стоимости поставок, затрат, связанных с пополнением, штрафов и т. д.

В такой общей постановке подобные задачи могут иметь самое разнообразное практическое применение. Например, под запасами можно понимать продукцию предприятия, которая производится непрерывно (пополнение) и отгружается потребителям определенными дискретными партиями (расход). При этом спрос на продукцию предполагается наперед заданным (детерминированный спрос) или подверженным случайным колебаниям (стохастическая задача). Управление запасами состоит в определении размеров необходимого выпуска продукции для удовлетворения заданного спроса. Цель — минимизация суммарных затрат на хранение и пополнение запасов.

Под запасами можно понимать запасы сырья или других материалов, поставляемых дискретными партиями (пополнение), которые должны обеспечить непрерывное потребление в процессе производства (расход). Критерием оптимальности могут служить суммарные затраты на хранение запасов, замораживание оборотных средств и поставки запасов.

Запасами могут быть товары, поставляемые в магазин определенными партиями и предназначенные для удовлетворения непрерывного, но подверженного случайным колебаниям покупательского спроса. Критерий оптимальности — суммарные затраты на поставки, хранение запасов и изменение производственного ритма; связи с вариациями спроса.

Запасами могут быть и сезонные товары, сохраняющиеся на складе ограниченной емкости. Товары можно покупать и продавать в различных количествах по ценам, меняющимся во времени. Задача состоит в определении политики покупок и продаж, обеспечивающих максимум суммарной прибыли, и является примером задачи складирования.

**Задачи о замене.** Одной из важных экономических проблем, с которыми приходится встречаться на практике, является определение оптимальной стратегии в замене старых станков, производственных зданий, агрегатов, машин и т.д., другими словами, старого оборудования на новое.

Старение оборудования включает его физический и моральный износ, в результате чего растут производственные затраты по выпуску продукции на старом оборудовании, увеличиваются затраты на его ремонт и обслуживание, а вместе с тем снижаются производительность и так называемая ликвидная стоимость.

Наступает момент, когда старое оборудование более выгодно продать, заменить новым, чем эксплуатировать ценой больших затрат. При этом оборудование можно заменить либо новым оборудованием того же вида, либо новым, более совершенным в техническом отношении с учетом технического прогресса.

Оптимальная стратегия замены оборудования состоит в определении оптимальных сроков замены. Критерием оптимальности при определении сроков замены может служить либо прибыль от эксплуатации оборудования, которую следует максимизировать, либо суммарные затраты на эксплуатацию в течение рассматриваемого промежутка времени, подлежащие минимизации.

*Задачи оптимального управления.* Обычно к этому типу задач относят задачи, связанные с нахождением распределенного во времени непрерывного управляющего воздействия. В экономике это прежде всего задачи прогнозирования тенденций развития, долгосрочных инвестиций и др., например задача оптимизации суммарного фонда потребления, где в качестве управляющего воздействия рассматривается величина инвестиций как функция времени (задача может быть сформулирована с учетом и без учета инвестиционного лага), задача максимизации дисконтированного потребления и т.д.

Все упомянутые классы задач (при этом их состав далеко не полон) требуют для своего решения применения специальных математических методов линейного и нелинейного программирования, динамического программирования, принципа максимума и некоторых других. Составной частью вычислительных работ при решении рассмотренных проблем могут являться задачи решения нелинейных уравнений и их систем, вычисления интегралов, решение дифференциальных уравнений и т.д.

## ИНСТРУМЕНТЫ МОДЕЛИРОВАНИЯ

Напомним, что *моделированием* называется процесс исследования свойств объекта по его модели. Этот процесс связан с необходимостью решения модели, как правило, на ПК. Следовательно, необходимо уметь решать различные задачи вычислительного характера, которые могут встретиться при моделировании (или по крайней мере знать, как это нужно делать и какой “ценой” это можно сделать). Вычисления здесь понимаются в самом широком смысле: это и “привычные” математические операции, и разнообразные логические, применяющиеся при анализе различных взаимосвязей с целью выявления тех или иных причинно-следственных связей. Отсюда вытекает важность одного из краеугольных камней математического моделирования — вычислительных алгоритмов.

С ростом парка высокопроизводительных компьютеров появляется некоторое “обожествление” моделей и моделирования вообще. Это связано с кажущейся легкостью получения результата, с наличием большого числа моделей (правильных и неправильных, адекватных и неадекватных в конкретном случае) и моделирующих программ. При этом нередко встречаются случаи, когда “ типовые ” программы с “ типовыми ” алгоритмами дают не очень объяснимые решения или когда результаты предсказаний по модели не совпадают с действительностью. Может также оказаться, что время решения на первый взгляд несложной задачи недопустимо велико. Не всегда рассматриваемые проблемы связаны с неадекватностью моделей или недостоверностью исходных данных, в ряде случаев они могут быть объяснены неэффективностью применяемых методов, особенно если имеют место несопоставимые с задачей затраты времени на ее решение. Для правильного распределения усилий исследователя, ресурсов компьютеров необходимо знать основные особенности и области применения различных вычислительных методов, использующихся при моделировании и являющихся инструментом модели-

рования. Этим аспектам математического моделирования и посвящена настоящая книга.

Вычислительные методы применяются как на стадии построения модели, так и на стадии решения модели на ПК.

Построение модели включает в себя, в частности, такие элементы, как *структурный синтез* и *идентификация параметров модели*. Структурный синтез предполагает построение структуры модели или исходя из содержательного анализа объекта с учетом входящих в него элементов, их связи между собой (в этом случае синтез выполняется, как правило, в аналитической форме) или формально (просто задаются структурой исходя из априорных сведений), или еще более формально, например выбирают в качестве формы полином. Во втором случае обязательно за синтезом следует стадия идентификации — определение параметров модели исходя из обеспечения наилучшей ее точности с использованием некоторых экспериментальных данных (экспериментальный метод построения модели). В первом случае параметры модели могут полностью браться из существующих нормативов, соответствующих фундаментальных законов, таблиц и т.п. (аналитический метод построения модели) или частично определяться из обеспечения наилучшей близости модели и имеющихся исходных данных (экспериментально-аналитический метод построения модели). В последнем случае этап идентификации обязательно присутствует при создании модели.

Процессы идентификации моделей базируются на различных вычислительных методах интерполяции и аппроксимации, которые в свою очередь при реализации могут использовать численные методы решения систем линейных и нелинейных конечных и дифференциальных уравнений, вычисления интегралов, поиска минимума функций и функционалов и др. Таким образом, уже на стадии построения моделей могут широко применяться в качестве инструментария вычислительные методы.

В процессе анализа свойств моделей на компьютере (моделирование в узком смысле слова) на первый план выходят вычислительные методы. Именно от них в первую очередь зависит не

только собственно результат и время его получения, но и сама возможность его достижения. Это обусловлено тем, что многие вычислительные методы имеют итерационный характер и требуют выполнения специальных условий сходимости, при несоблюдении которых результат вообще не будет получен. В процессе решения моделей находят применение вычислительные методы решения отдельных нелинейных уравнений и их систем, систем линейных уравнений, дифференциальных уравнений и их систем, вычисления интегралов, интерполяции и аппроксимации, методы поиска экстремума функций и функционалов и др. Многие из рассмотренных классов методов используют при своей реализации в качестве вспомогательных методы других классов. Все это делает необходимым знание основ вычислительных методов, их особенностей и областей применения. Именно рассмотрению этих проблем и посвящена книга.

---

# ОБРАБОТКА ТАБЛИЧНЫХ ДАННЫХ

---

В этом разделе рассматриваются только основные направления обработки данных: интерполяция и аппроксимация, являющиеся базой для решения всех других задач обработки табличных данных, а также методы численного интегрирования. Другие направления, такие, как сглаживание, получение наилучших моделей и т.п., основанные на идеях интерполяции и аппроксимации, здесь не рассматриваются.

## ИНТЕРПОЛЯЦИЯ

---

Основная задача *интерполяции* — нахождение значения таблично заданной функции в тех точках внутри данного интервала, где она не задана. *Экстраполяция* — несколько более “широкое” понятие, оно сводится к восстановлению функции в точках за пределами заданного интервала. В обоих случаях исходные табличные данные могут быть получены как экспериментально (в этом случае принципиально отсутствуют промежуточные данные без дополнительных работ), так и расчетным путем по сложным зависимостям (в этом случае найти с помощью интерполяции значение сложной функции бывает проще, чем непосредственным вычислением по сложной формуле).

## КОНЦЕПЦИЯ ИНТЕРПОЛЯЦИИ

Решение задач интерполяции и экстраполяции обеспечивается построением интерполяционной функции  $L(x)$ , приближенно заменяющей исходную  $f(x)$ , заданную таблично, и проходящей через все заданные точки — *узлы интерполяции*. С помощью этой функции можно рассчитать искомое значение исходной функции в любой точке.

В связи с интерполяцией рассматриваются три основные проблемы.

- 1) выбор интерполяционной функции  $L(x)$ ;
- 2) оценка погрешности интерполяции  $R(x)$ ;
- 3) размещение узлов интерполяции для обеспечения наивысшей возможной точности восстановления функции  $(x_1, x_2, \dots, x_n)$ .

Специальные методы интерполяции позволяют определить искомое значение функции без непосредственного прямого построения интерполяционной функции. В принципе все интерполяционные методы, базирующиеся на использовании в качестве интерполяционной функции полиномов, дают одни и те же результаты, но с разными затратами. Это объясняется тем, что полином  $n$ -й степени, содержащий  $n + 1$  параметр и проходящий через все заданные  $n + 1$  точки, — единственный. Кроме того, полином можно представить как усеченный ряд Тейлора, в который разложили исходную дифференцируемую функцию. Это, пожалуй, одно из главных достоинств полинома как интерполяционной функции. Поэтому чаще первая проблема интерполяции решается выбором в качестве интерполяционной функции именно полинома, хотя могут применяться и другие функции (например, тригонометрические полиномы, другие функции, выбранные из неформальных условий содержательной задачи).

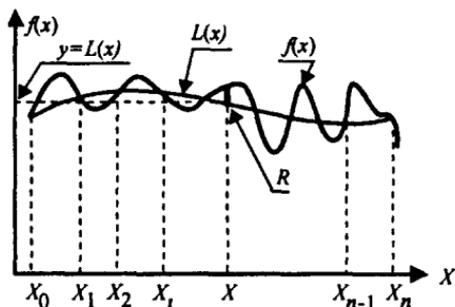


Рис. 2. Иллюстрация интерполяции

Выбор вида интерполяционной функции является в общем случае важной задачей, особенно если помнить, что через заданные точки можно провести любое количество функций (рис. 2). Следует отметить, что существует очевидный способ построения интерполяционной функции: из условия прохождения функции через все точки составляет

система уравнений, из решения которой и находятся ее параметры. Однако этот путь далеко не самый эффективный, особенно при большом числе точек.

# ОСНОВНЫЕ МЕТОДЫ

## МЕТОД ЛАГРАНЖА

Пусть известны значения некоторой функции  $f(x)$  в  $n+1$  различных произвольных точках  $y_i = f(x_i)$ ,  $i = 0, \dots, n$ . Для интерполирования (восстановления) функции в какой-либо точке  $x$ , принадлежащей отрезку  $[x_0, x_n]$ , необходимо построить интерполяционный полином  $n$ -го порядка, который в методе Лагранжа представляется следующим образом:

$$L_n(x) = y_0 Q_0(x) + \dots + y_j Q_j(x) + \dots + y_n Q_n(x),$$

$$\text{где } Q_j(x) = \frac{(x-x_0)\dots(x-x_{j-1})(x-x_{j+1})\dots(x-x_n)}{(x_j-x_0)\dots(x_j-x_{j-1})(x_j-x_{j+1})\dots(x_j-x_n)}.$$

Причем нетрудно заметить, что  $Q_j(x_i) = 0$ , если  $i \neq j$ , и  $Q_j(x_i) = 1$ , если  $i = j$ . Если раскрыть произведение всех скобок в числителе (в знаменателе все скобки — числа), то получим полином  $n$ -го порядка от  $x$ , так как в числителе содержится  $n$  сомножителей первого порядка. Следовательно, интерполяционный полином Лагранжа не что иное, как обычный полином  $n$ -го порядка, несмотря на специфическую форму записи.

Оценить погрешность интерполяции в точке  $x$  из  $[x_0, x_n]$  (т.е. решить вторую проблему интерполяции) можно по формуле

$$R(x) = |f(x) - L_n(x)| \leq \frac{M_{n+1}}{(n+1)!} \prod_{i=0}^n (x - x_i).$$

В формуле  $M_{n+1} = \max |f^{(n+1)}(x)|$  — максимальное значение  $(n+1)$ -й производной исходной функции  $f(x)$  на отрезке  $[x_0, x_n]$ . Следовательно, для того чтобы оценить погрешность интерполяции, необходима некоторая дополнительная информация об исходной функции (это должно быть понятно, так как через заданные исходные точки может проходить бесчисленное количество различных функций, для которых и погрешность будет разной). Такой информацией является производная  $n+1$  порядка, которую не так просто найти. Ниже будет показано, как выйти из такого положения. Отметим также, что применение формулы погрешности возможно, только если функция дифференцируема  $n+1$  раз.

**Пример 1.**

Дана таблично заданная функция

$x$	0	1	2	6
$y$	-1	-3	3	1187

Требуется найти  $y$  при  $x=4$ . В данном случае  $n=3$ . Запишем функцию Лагранжа подробно:

$$\begin{aligned}
 L_3(x) &= y_0 \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + y_1 \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} + \\
 &+ y_2 \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} + y_3 \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} = \\
 &= -1 \frac{(4-1)(4-2)(4-6)}{(0-1)(0-2)(0-6)} - 3 \frac{(4-0)(4-2)(4-6)}{(1-0)(1-2)(1-6)} + \\
 &+ 3 \frac{(4-0)(4-1)(4-6)}{(2-0)(2-1)(2-6)} + 1187 \frac{(4-0)(4-1)(4-2)}{(6-0)(6-1)(6-2)} = 255.
 \end{aligned}$$

В данном примере отсутствует информация о значении четвертой производной исходной функции, поэтому оценить погрешность не представляется возможным.

**Пример 2.**

Дана функция (для иллюстрации метода будем считать ее сложной)  $y = \sqrt{x}$ . Найти погрешность интерполяции функции при  $x = 115$ . “По памяти” запишем:

$x$	100	121	144
$y$	10	11	12

Имеем всего три узла интерполяции,  $n=2$ . Оценим максимальное значение третьей производной для оценки погрешности

$$y' = x^{-1/2} / 2; \quad y'' = -x^{-3/2} / 4; \quad y''' = 3x^{-5/2} / 8.$$

$$M_3 = \max |y'''| = 3(100)^{-5/2} / 8.$$

Погрешность при интерполяции по трем узлам будет:

$$R \leq 3(100)^{-5/2}/8|(115-100)(115-121)(115-144)|/(1 \cdot 2 \cdot 3) \approx 1,6 \cdot 10^{-3}.$$

Ошибка получилась достаточно малой, а само значение  $y(115)$  читатель может попробовать найти самостоятельно и проверить его по таблицам или на калькуляторе.

### МЕТОД НЬЮТОНА

Пусть известны значения некоторой функции  $f(x)$  в  $n+1$  произвольных, попарно не совпадающих точках  $y_i = f(x_i)$ ,  $i = 0, \dots, n$ . В общем случае интерполяция по формулам Ньютона может производиться для произвольно расположенных узлов интерполяции, но чаще применяется для равномерно расположенных. Поэтому далее рассматривается только случай с равномерным расположением узлов. Тогда  $x_{i+1} = x_i + h$ , где  $h = (x_0 - x_n) / n$ .

Метод использует понятие конечных разностей. Конечная разность  $k$ -го порядка в  $i$ -й точке вычисляется следующим образом:

$$\Delta^k y_i = \Delta^{k-1} y_{i+1} - \Delta^{k-1} y_i,$$

т.е. через конечные разности более низкого порядка. Можно заметить, что при наличии  $n+1$  точки  $(0, 1, 2, \dots, n)$  конечную разность первого порядка можно вычислить только для первых  $n$  точек, конечную разность  $n$ -го порядка — только для нулевой точки, конечную разность  $k$ -го порядка — только для первых  $n - k + 1$  точек.

Интерполяционный многочлен Ньютона записывается следующим образом:

$$P_n(x) = y_0 + \frac{\Delta y_0(x-x_0)}{h} + \frac{\Delta^2 y_0(x-x_0)(x-x_1)}{2! h^2} + \\ + \frac{\Delta^3 y_0(x-x_0)(x-x_1)(x-x_2)}{3! h^3} + \dots + \frac{\Delta^i y_0(x-x_0)(x-x_1) \dots (x-x_{i-1})}{i! h^i} + \\ + \frac{\Delta^n y_0(x-x_0)(x-x_1) \dots (x-x_{n-1})}{n! h^n}.$$

Нетрудно заметить, что, как и в предыдущем рассмотренном методе, это выражение есть не что иное, как обычный полином  $n$ -го порядка от  $x$ , только записанный в другой форме. Это можно доказать, раскрыв все скобки и приведя подобные члены. Конечные разности в выражении — это числовые коэффициенты, вычисленные по заданным точкам.

Часто вводят безразмерную переменную  $q$ , показывающую, сколько содержится шагов от  $x_0$  до заданной точки  $x$ :  $q = \frac{x - x_0}{h}$ .

В этом случае выражение для интерполяционного полинома запишется так:

$$P_n(x) = y_0 + \Delta y_0 q + \frac{\Delta^2 y_0 q(q-1)}{2!} + \frac{\Delta^3 y_0 q(q-1)(q-2)}{3!} + \dots$$

$$\dots + \frac{\Delta^i y_0 q(q-1)\dots(q-i+1)}{i!} + \dots + \frac{\Delta^n y_0 q(q-1)\dots(q-n+1)}{n!}.$$

Обе приведенные формулы носят название первой интерполяционной формулы Ньютона (только записанные в разных переменных —  $x$  и  $q$ ) и рекомендуются для применения при интерполяции вперед (в сторону увеличения  $x$ ) или при экстраполяции назад (левее  $x_0$ ).

Существует вторая формула, которая рекомендуется для применения при интерполяции назад (т.е. в конце интервала, но левее  $x_n$ ) или при экстраполяции вперед, правее  $x_n$ . Она записывается следующим образом:

$$P_n(x) = y_n + \Delta y_{n-1} q + \frac{\Delta^2 y_{n-2} q(q+1)}{2!} + \frac{\Delta^3 y_{n-3} q(q+1)(q+2)}{3!} + \dots$$

$$\dots + \frac{\Delta^i y_{n-i} q(q+1)\dots(q+i-1)}{i!} + \dots + \frac{\Delta^n y_0 q(q+1)\dots(q+n-1)}{n!}.$$

Погрешность интерполяции можно оценить так же, как и в предыдущем методе. Хотя при использовании относительной переменной  $q$  можно указать и другую формулу (которая, естественно, полностью по результату совпадает с предыдущей формулой погрешности):

$$R(x) = |f(x) - L_n(x)| \leq \frac{M_{n+1} h^{n+1}}{(n+1)!} q(q-1)(q-2)\dots(q-n).$$

Использование конечных разностей, которые для дискретных функций являются своеобразными аналогами производных непрерывных функций, помогает находить погрешность интерполяции. Учет соотношения:

$$f'(x) \approx \frac{\Delta y}{h}; \quad f''(x) \approx \frac{\Delta^2 y}{h^2}; \quad f'''(x) \approx \frac{\Delta^3 y}{h^3}; \quad \dots; \quad f^{(i)}(x) \approx \frac{\Delta^i y}{h^i},$$

тогда для получения приближенного значения  $M_{n+1}$  достаточно иметь несколько (или одну) дополнительных точек  $x_{n+1}, x_{n+2}, \dots$ , с использованием которых легко найти максимальное значение конечной разности (или даже одно значение  $\Delta^{(n+1)}y(x_0)$ , если есть только одна дополнительная точка)  $(n+1)$ -го порядка  $\max\{\Delta^{(n+1)}y(x)\}$ .

$$R(x) = |f(x) - L_n(x)| \leq \frac{\max\{|\Delta^{(n+1)}y(x)|\}}{(n+1)!} q(q-1)(q-2)\dots(q-n).$$

Таким же путем можно поступить и в методе Лагранжа, но при условии, если он реализуется при равномерном шаге.

### *Пример.*

Даны следующие точки:

$x$	2,0	2,1	2,2	2,3	2,4	2,5	2,6
$y$	0,0540	0,0440	0,0355	0,0283	0,0224	0,175	0,0136

Необходимо найти  $y(2,05)$ . Используем для интерполяции только три первые точки, а остальные используем для оценки погрешности.

Следовательно,  $n = 2$ , относительное значение аргумента  $q = (x - x_0) / h$  ( $h = 0,1$ , что видно из исходных данных) будет равно  $q = 0,5$ . Воспользуемся первой интерполяционной формулой Ньютона для безразмерной переменной  $q$ , для чего предварительно составим таблицу конечных разностей (табл. 1).

Таблица 1

x	y	$\Delta y$	$\Delta^2 y$	$\Delta^3 y$
2,0	0,0540	-0,0100	0,0015	-0,0002
2,1	440	85	13	0
2,2	355	72	13	3
2,3	283	59	10	10
2,4	224	49	0	
2,5	175	49		
2,6	136			

Для большей наглядности в таблице опущены нули и знаки, которые показаны лишь в первой строчке каждого столбика.

$$y(2,05) = 0,0540 + 0,5(-0,01) + 0,5(0,5-1) \cdot 0,0015/3! = 0,0488125.$$

Оценим погрешность найденного значения  $y$ . Из таблицы находим, что  $M_3 = 0,0010$ , тогда

$$R \leq \frac{0,001 \cdot |0,5(0,5-1)(0,5-2)|}{3!} = 0,0000625.$$

### МЕТОД ЧЕБЫШЕВА

Этот метод в основном является иллюстрацией решения третьей проблемы интерполяции: выбор узлов интерполяции (если, конечно, это возможно при решении конкретной задачи) для получения при заданном числе узлов минимально возможной погрешности. Применение метода Чебышева позволяет обосновать размещение узлов интерполяции в заданных пределах на числовой оси. Предварительно переменная  $x$  преобразуется в переменную  $z$ , изменяющуюся от  $-1$  до  $1$  при изменении  $x$  от  $x_0$  до  $x_n$ .

$$z = -\frac{x_0 + x_n}{x_n - x_0} + \frac{2x}{x_n - x_0}.$$

В свою очередь переменная  $z$  на интервале  $[-1, 1]$  формирует узлы следующим образом:

$$z_i = -\cos\left(\frac{i\pi}{n}\right).$$

Из этого соотношения видно, что узлы интерполяции располагаются неравномерно, сгущаясь к концам отрезка  $[x_0, x_n]$ . В вычислительном плане для интерполяции можно пользоваться интерполяционной формулой Лагранжа.

Пример применения метода не приводим, так как этот метод не дает ничего нового с точки зрения вычислений.

## МЕТОД СПЛАЙНОВ

В ряде случаев возникает задача восстановления не только значений функции, но также ее первой и второй производной. Это, конечно, можно сделать путем дифференцирования интерполяционного полинома, если его получить. Но оказывается, что точность восстановления производных в этих случаях неудовлетворительно низкая. Для решения указанного класса задач успешно применяется *сплайновая интерполяция*.

Наибольшее распространение получила интерполяция с помощью кубических сплайнов. *Сплайн* — это функция, которая на каждом междуузловом интервале совпадает с некоторым полиномом, своим для каждого интервала. Полиномы соседних интервалов стыкуются так, чтобы функция была непрерывной. Дополнительно требуют непрерывности нескольких производных (в кубических сплайнах — двух). *Кубический сплайн* склеивается из полиномов третьей степени, которые для  $i$ -го участка записываются так:

$$y = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i.$$

Для всего интервала будет соответственно  $n$  кубических полиномов, отличающихся коэффициентами  $a_i, b_i, c_i, d_i$ . Чаше всего узлы при сплайновой интерполяции располагают равномерно, т.е.  $x_{i+1} - x_i = \text{const} = h$  (хотя это и необязательно).

Необходимо найти четыре коэффициента при условии прохождения каждого полинома через две точки  $(x_i, y_i)$  и  $(x_{i+1}, y_{i+1})$ , следствием чего являются следующие очевидные уравнения:

$$\begin{aligned} y_i &= d_i, & i &= 0, 1, 2, \dots, n-1, \\ y_{i+1} &= a_i h^3 + b_i h^2 + c_i h + d_i, & i &= 0, 1, 2, \dots, n-1. \end{aligned}$$

Первое условие соответствует прохождению полинома через начальную точку, второе — через конечную точку. Найти все коэффициенты из этих уравнений нельзя, так как условий меньше, чем искомым параметров. Поэтому указанные условия дополняют условиями гладкости функции (т.е. непрерывности первой производной) и гладкости первой производной (т.е. непрерывности второй производной) в узлах интерполяции. Математически эти условия записываются как равенства соответственно первой и второй производных в конце  $i$ -го и в начале  $(i+1)$ -го участков.

Так как  $y' = 3a_i(x - x_i)^2 + 2b_i(x - x_i) + c_i$  и  $y'' = 6a_i(x - x_i) + 2b_i$ , то

$$3a_i h^2 + 2b_i h + c_i = c_{i+1}, \quad i = 0, 1, 2, \dots, n-2$$

( $y'(x_{i+1})$  в конце  $i$ -го участка равна  $y'(x_{i+1})$  в начале  $(i+1)$ -го),

$$6a_i h + 2b_i = 2b_{i+1}, \quad i = 0, 1, 2, \dots, n-2$$

( $y''(x_{i+1})$  в конце  $i$ -го участка равна  $y''(x_{i+1})$  в начале  $(i+1)$ -го).

Получилась система линейных уравнений (для всех участков), содержащая  $4n - 2$  уравнения с  $4n$  неизвестными (неизвестные  $a_1, a_2, \dots, a_n, b_1, \dots, b_n$  — коэффициенты сплайнов). Для решения системы добавляют два граничных условия одного из следующих видов (чаще применяют 1):

- 1)  $y''(x_0) = y''(x_n) = 0$ ,      3)  $y'(x_0) = y'(x_n) = 0$ ,
- 2)  $y'(x_0) = g_0$ ,  $y'(x_n) = g_n$ ,    4)  $y'(x_0) = y'(x_n)$ ,  $y''(x_0) = y''(x_n)$ .

Совместное решение  $4n$  уравнений позволяет найти все  $4n$  коэффициента.

Для восстановления производных можно продифференцировать на каждом участке соответствующий кубический полином. В случае необходимости определения производных в узлах существуют специальные приемы, сводящие определение производных к решению более простой системы уравнений относительно искомым производных второго или первого порядка. К важным достоинствам интерполяции кубическими сплайнами относится получение функции, имеющей минимальную возможную кривиз-

ну. К недостаткам сплайновой интерполяции относится необходимость получения сравнительно большого числа параметров.

Вычислительный пример рассматривать не будем, что связано с большим объемом вычислений, которые нецелесообразно проводить вручную.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

### *Метод Лагранжа*

1. Полиномом какой степени является интерполяционный полином Лагранжа при  $n+1$  узлах?
2. Может ли метод Лагранжа применяться для экстраполяции?
3. Что влияет на точность интерполяции в методе Лагранжа?
4. Можно ли добавлять новые узлы интерполяции при использовании метода Лагранжа?
5. Можно ли располагать узлы интерполяции произвольно при использовании метода?
6. К какому классу функций относится функция, задаваемая интерполяционной формулой Лагранжа?
7. Как повлияет дополнительная  $n+1$  точка исходных данных внутри отрезка  $[x_0, x_n]$  на точность интерполяции?
8. Как определить погрешность интерполяции в узле?
9. Как влияет количество узлов интерполяции на точность интерполяции?
10. Каким путем в общем случае можно повысить точность интерполяции?

### *Метод Ньютона*

1. Может ли метод Ньютона применяться для экстраполяции?
2. Можно ли располагать неравномерно узлы интерполяции при использовании основного метода Ньютона?
3. Каким путем можно повысить точность интерполяции при использовании метода Ньютона?
4. Конечную разность какого наивысшего порядка можно получить по  $n$  исходным точкам?

5. Как выражается конечная разность  $k$ -го порядка?
6. Можно ли конечную разность выразить только через исходные значения функции?
7. В чем заключается разница между первой и второй интерполяционными формулами Ньютона?
8. Какой прием можно использовать для оценки погрешности интерполяции таблично заданной функции?
9. Какой степени можно получить интерполяционный полином при трех заданных точках методом Ньютона?
10. Сколько существует интерполяционных полиномов степени  $n$ ?

### *Сплайновая интерполяция*

1. Каковы основные возможности сплайновой интерполяции?
2. Что называется кубическим сплайном?
3. Через сколько исходных точек проходит один кубический полином в кубическом сплайне?
4. Сколько коэффициентов, подлежащих определению, содержит кубический сплайн?
5. Какие условия используются для нахождения коэффициентов сплайнов?
6. Какую функцию называют гладкой?
7. Могут ли узлы сплайнов располагаться неравномерно?
8. Что относится к недостаткам сплайновой интерполяции?
9. Можно ли найти с помощью сплайновой интерполяции производные таблично заданной исходной функции в узлах?
10. Зачем нужны дополнительные граничные условия?

## **АППРОКСИМАЦИЯ**

---

Основная задача аппроксимации — построение приближенной (аппроксимирующей) функции, в целом наиболее близко проходящей около данных точек или около данной непрерывной функции. Такая задача возникает при наличии погрешности в исходных данных (в этом случае нецелесообразно проводить функ-

цию точно через все точки, как в интерполяции) или при желании получить упрощенное математическое описание сложной или неизвестной зависимости.

## КОНЦЕПЦИЯ АППРОКСИМАЦИИ

Близость исходной и аппроксимирующей функций определяется числовой мерой — *критерием аппроксимации* (близости). Наибольшее распространение получил квадратичный критерий, равный сумме квадратов отклонений расчетных значений от “экспериментальных” (т.е. заданных), — критерий близости в заданных точках:

$$R = \sum_{i=1}^n \beta_i (y_i - y_i^{\text{расчет.}})^2.$$

Здесь  $y_i$  — заданные табличные значения функции;  $y_i^{\text{расчет.}}$  — расчетные значения по аппроксимирующей функции;  $\beta_i$  — весовые коэффициенты, учитывающие относительную важность  $i$ -й точки (увеличение  $\beta_i$  приводит при стремлении уменьшить  $R$  к уменьшению прежде всего отклонения в  $i$ -й точке, так как это отклонение искусственно увеличено за счет относительно большого значения весового коэффициента).

Квадратичный критерий обладает рядом “хороших” свойств, таких, как дифференцируемость, обеспечение единственного решения задачи аппроксимации при полиномиальных аппроксимирующих функциях.

Другим распространенным критерием близости является следующий:

$$R = \max_i |y_i - y_i^{\text{расчет.}}|.$$

Этот критерий менее распространен в связи с аналитическими и вычислительными трудностями, связанными с отсутствием гладкости функции и ее дифференцируемости.

В обоих рассмотренных случаях в качестве значения функции  $y_i$  можно брать не только абсолютные, но и относительные значения, например,  $y_i/y_n$  и др.

Выделяют две основные задачи:

- 1) получение аппроксимирующей функции, описывающей имеющиеся данные, с погрешностью не хуже заданной;
- 2) получение аппроксимирующей функции заданной структуры с наилучшей возможной погрешностью.

Чаще всего первая задача сводится ко второй перебором различных аппроксимирующих функций и последующим выбором наилучшей.

## ОСНОВНЫЕ МЕТОДЫ

### МЕТОД НАИМЕНЬШИХ КВАДРАТОВ

Метод базируется на применении в качестве критерия близости суммы квадратов отклонений заданных и расчетных значений. При заданной структуре аппроксимирующей функции  $y_i^{\text{расчет}}(x)$  необходимо таким образом подобрать параметры этой функции, чтобы получить наименьшее значение критерия близости, т.е. наилучшую аппроксимацию. Рассмотрим путь нахождения этих параметров на примере полиномиальной функции одной переменной:

$$y_i^{\text{расчет}}(x) = a_k x^k + a_{k-1} x^{k-1} + \dots + a_0 = \sum_{j=0}^k a_j x^j.$$

Запишем выражение критерия аппроксимации при  $\beta_i = 1$  ( $i = 1, 2, \dots, n$ ) для полиномиального  $y_i^{\text{расчет}}(x)$ :

$$R = \sum_{i=1}^n \left( y_i - \sum_{j=0}^k a_j x_i^j \right)^2.$$

Искомые переменные  $a_j$  можно найти из необходимого условия минимума  $R$  по этим переменным, т.е.  $dR / da_p = 0$  (для  $p = 0, 1, 2, \dots, k$ ). Продифференцируем по  $a_p$  ( $p$  — текущий индекс):

$$dR / da_p = 2 \sum (y_i - \sum a_j x_i^j) (-x_i^p) = 0, \quad p = 0, 1, 2, \dots, k, \quad i = 1, 2, \dots, n.$$

Значения индексов суммирования для простоты опущены.

После очевидных преобразований (сокращение на два, раскрытие скобок, изменение порядка суммирования) получим

$$dR/da_p = \sum_i y_i(-x_i^p) + \sum_i \sum_j a_j x_i^j x_i^p = \sum_i y_i(-x_i^p) + \sum_j a_j \sum_i x_i^j x_i^p = 0, \\ p = 0, 1, 2, \dots, k.$$

Перепишем последние равенства

$$\sum_j a_j \sum_i x_i^j x_i^p = \sum_i y_i x_i^p, \quad p = 0, 1, 2, \dots, k, \quad i = 1, 2, \dots, n.$$

Получилась система  $n+1$  уравнений с таким же количеством неизвестных  $a_j$ , причем линейная относительно этих переменных. Эта система называется системой нормальных уравнений. Из ее решения находятся параметры  $a_j$  аппроксимирующей функции, обеспечивающие  $\min R$ , т.е. наилучшее возможное квадратичное приближение. Зная коэффициенты, можно (если нужно) вычислить и величину  $R$  (например, для сравнения различных аппроксимирующих функций). Следует помнить, что при изменении даже одного значения исходных данных (или пары значений  $x_i, y_i$ , или одного из них) все коэффициенты изменят в общем случае свои значения, так как они полностью определяются исходными данными. Поэтому при повторении аппроксимации с несколько изменившимися данными (например, вследствие погрешностей измерения, помех, влияния неучтенных факторов и т.п.) получится другая аппроксимирующая функция, отличающаяся коэффициентами. Обратим внимание на то, что коэффициенты  $a_j$  полинома находятся из решения системы уравнений, т.е. они связаны между собой. Это приводит к тому, что если какой-то коэффициент вследствие его малости захочется отбросить, придется пересчитывать заново оставшиеся. Можно рассчитать количественные оценки тесноты связи коэффициентов. Существует специальная теория планирования экспериментов, которая позволяет обосновать и рассчитать значения  $x_i$ , используемые для аппроксимации, чтобы получить заданные свойства коэффициентов (несвязанность, минимальная дисперсия коэффициентов и т.д.) или аппроксимирующей функции (равная точность описания реальной зависимости в различных направлениях, минимальная дисперсия предсказания значения функции и т.д.).

В случае постановки другой задачи — найти аппроксимирующую функцию, обеспечивающую погрешность не хуже заданной, — необходимо подбирать и структуру этой функции. Эта задача значительно сложнее предыдущей (найти параметры аппроксимирующей функции заданной структуры, обеспечивающей наилучшую возможную погрешность) и решается в основном путем перебора различных функций и сравнения получающихся мер близости. Для примера на рис. 3 приведены для визуального сравнения исходная и аппроксимирующие функции с различной степенью полинома, т.е. функции с различной структурой. Не следует забывать, что с повышением точности аппроксимации растет и сложность функции (при полиномиальных аппроксимирующих функциях), что делает ее менее удобной при использовании.

### *Пример.*

Необходимо найти аппроксимирующую функцию в виде линейного полинома  $y = a_0 + a_1x$  по имеющимся экспериментальным данным

$x$	-26	-22	-16	-11	-5	3	10	25	42
$y$	66,7	71,0	76,3	80,6	85,7	92,9	99,4	113,6	125,1

Система нормальных уравнений будет выглядеть следующим образом:

$$a_0n + a_1 \sum_{i=1}^n x_i = \sum_{i=1}^n y_i,$$

$$a_0 \sum_{i=1}^n x_i + a_1 \sum_{i=1}^n x_i^2 = \sum_{i=1}^n y_i x_i.$$

Используя имеющиеся данные, получим

$$n = 9, \quad \sum_i x_i = 0, \quad \sum_i x_i^2 = 4060, \quad \sum_i y_i = 811,3, \quad \sum_i y_i x_i = 3534,8.$$

Решим полученную систему линейных уравнений относительно  $a_0$  и  $a_1$ , получим  $a_0 = 90,1$ ,  $a_1 = 0,87$ .



*a*



*б*



*в*

**Рис. 3.** Влияние степени аппроксимирующего полинома  $M$  на точность аппроксимации:  
*a* —  $M=2$ , *б* —  $M=3$ , *в* —  $M=5$ .

Аппроксимирующая функция имеет вид

$$y = 90,1 + 0,87x.$$

Приведем расчетные значения функции:

$x$	-26	-22	-16	-11	-5	3	10	25	42
$y$	64,97	70,96	76,18	80,53	85,75	92,71	98,8	111,85	126,64

### **МЕТОД РАВНОМЕРНОГО ПРИБЛИЖЕНИЯ**

*Равномерным приближением* называется аппроксимация, в которой в качестве критерия близости используется модуль максимального отклонения расчетных и заданных значений. При этом решается задача минимизации этого критерия, т.е.

$$R = \max |y_i - y_i^{\text{расчет.}}| \rightarrow \min.$$

Принято считать, что наилучшее равномерное приближение обеспечивает несколько лучшую аппроксимацию, чем среднеквадратичное. Однако теоретические оценки показывают незначительность такого преимущества; оно более значительно для функций с непрерывными старшими производными, не слишком большими по абсолютной величине, при небольших объемах исходных данных. Однако существенный недостаток — отсутствие эффективных вычислительных алгоритмов (кроме прямого поиска  $\min R$  методами нелинейного программирования, причем необходимо применять методы, пригодные для недифференцируемых функций) — делает малоприменимым данный способ аппроксимации.

### **КОНТРОЛЬНЫЕ ВОПРОСЫ**

#### ***Метод наименьших квадратов***

1. Можно ли при аппроксимации полиномом таблично заданной функции обеспечить прохождение аппроксимирующей функции точно через все точки?
2. Назначение весовых коэффициентов в критерии близости исходной и аппроксимирующей функций.

3. Можно ли повысить точность, одновременно увеличив в несколько раз все весовые коэффициенты?
4. Всегда ли увеличение суммы квадратов отклонений соответствует худшей близости исходной и аппроксимирующей функций?
5. Можно ли с помощью МНК найти параметры неполиномиальной аппроксимирующей функции?
6. В чем отличие применения метода при использовании в качестве аппроксимирующей функции полинома и показательной функции?
7. В каком случае система нормальных уравнений получается линейной относительно искомым коэффициентов?
8. В каком случае не удастся получить искомые коэффициенты непосредственно из решения системы нормальных уравнений?
9. Можно ли обеспечить требование, чтобы аппроксимирующая функция практически точно проходила через отдельные выбранные точки?
10. В чем основное достоинство квадратичного критерия близости исходной и аппроксимирующей функций?

### *Метод равномерного приближения*

1. Что можно отнести к достоинствам критерия близости в методе равномерного приближения?
2. Является ли единственным аппроксимирующий полином, например, 3-й степени, получаемый методом равномерного приближения?
3. Каким образом можно определить наилучшую степень аппроксимирующего полинома?
4. Почему нельзя минимум критерия близости найти путем использования необходимых условий оптимальности, известных из математического анализа?
5. Какой вид экстремума критерия близости ищется при определении параметров:  $\min$  или  $\max$ ?
6. Почему метод равномерного приближения не получил широкого распространения?

7. В какой точке минимизируется в методе равномерного приближения разность между расчетным и заданным значением функции?
8. Можно ли при аппроксимации произвольно задавать степень аппроксимирующего полинома?
9. Как можно обеспечить немного отличающуюся относительную погрешность аппроксимации на разных участках, если функция имеет очень большой размах?
10. Может ли степень аппроксимирующего полинома быть выше числа узлов аппроксимации?

## ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ

Вычисление интегралов встречается при моделировании достаточно часто. Численные методы обычно применяются при взятии неберущихся интегралов от достаточно сложных функций, которые предварительно табулируются, или при интегрировании таблично заданных функций, что в экономических приложениях встречается значительно чаще.

### КОНЦЕПЦИЯ ЧИСЛЕННОГО ИНТЕГРИРОВАНИЯ

Все численные методы строятся на том, что подынтегральная функция приближенно заменяется более простой (горизонтальной или наклонной прямой, параболой 2-го, 3-го или более высокого порядка), от которой интеграл легко берется. В результате получаются формулы интегрирования, называемые *квadrатурами*, в виде взвешенной суммы ординат подынтегральной функции в отдельных точках:

$$\int_a^b f(x) dx \approx \sum_i \omega_i f(x_i).$$

Чем меньше интервалы, на которых производят замену, тем точнее вычисляется интеграл. Поэтому исходный отрезок  $[a, b]$

для повышения точности делят на несколько равных или неравных интервалов, на каждом из которых применяют формулу интегрирования, а затем складывают результаты.

Все методы различаются значениями ординат  $x_i$  и весов  $\omega_i$ .

В большинстве случаев погрешность численного интегрирования определяется путем двойного интегрирования: с исходным шагом (шаг определяется путем равномерного деления отрезка  $b - a$  на число отрезков  $n$ :  $h = (b - a) / n$ ) и с шагом, увеличенным в 2 раза. Разница вычисленных значений интегралов определяет погрешность.

Сравнение эффективности различных методов проводится по степени полинома, который данным методом интегрируется точно, без ошибки. Чем выше степень такого полинома, тем выше точность метода, тем он эффективнее.

## ОСНОВНЫЕ МЕТОДЫ

### ПРОСТЕЙШИЕ МЕТОДЫ

К простейшим методам можно отнести методы прямоугольников (левых и правых) и трапеций. В первом случае подынтегральная функция заменяется горизонтальной прямой ( $y = c_0$ ) со значением ординаты, т.е. значения функции соответственно слева или справа участка, во втором случае — наклонной прямой ( $y = c_1x + c_0$ ). Формулы интегрирования при разбиении отрезка  $[a, b]$  на  $n$  частей с равномерным шагом  $h$  соответственно приобретают вид:

- для одного участка интегрирования:

$$I = \int_a^b f(x) dx \approx f(a)(b-a) = f(x_0)h, \quad I = \int_a^b f(x) dx \approx f(b)(b-a) = f(x_1)h,$$

$$\omega_0 = h = \text{const};$$

$$I = \int_a^b f(x) dx \approx \frac{f(a) + f(b)}{2(b-a)} = f(x_0) \frac{h}{2} + f(x_1) \frac{h}{2},$$

$$\omega_0 = \omega_1 = \frac{h}{2} = \text{const},$$

где  $x_0 = a, x_1 = b, h = b - a$ ;

- для  $n$  участков интегрирования:

$$I = \int_a^b f(x) dx \approx \sum_{i=0}^{n-1} f(x_i)h = \sum_{i=0}^{n-1} \omega_i f(x_i),$$

$$I = \int_a^b f(x) dx \approx \sum_{i=1}^n f(x_i)h = \sum_{i=1}^n \omega_i f(x_i),$$

$$\omega_i = h = \text{const};$$

$$\begin{aligned} I &= \int_a^b f(x) dx \approx \sum_{i=0}^{n-1} (f(x_i) + f(x_{i+1})) \frac{h}{2} = \\ &= f(x_0) \frac{h}{2} + \sum_{i=1}^{n-1} f(x_i)h + f(x_n) \frac{h}{2} = \sum_{i=0}^n \omega_i f(x_i), \end{aligned}$$

$$\omega_0 = \omega_n = \frac{h}{2}, \quad \omega_1 = \omega_2 = \dots = \omega_{n-1} = h.$$

Нетрудно заметить, что в методе прямоугольников интеграл вычислится абсолютно точно только при  $f(x) = c$  (const), а в методе трапеций — при  $f(x)$  линейной или кусочно-линейной.

На рис. 4 для сравнения приведены примеры прямоугольников при различном числе участков. Наглядно видно, что площадь всех прямоугольников на правом рисунке меньше отличается от площади под кривой  $f(x)$ , чем на левом.

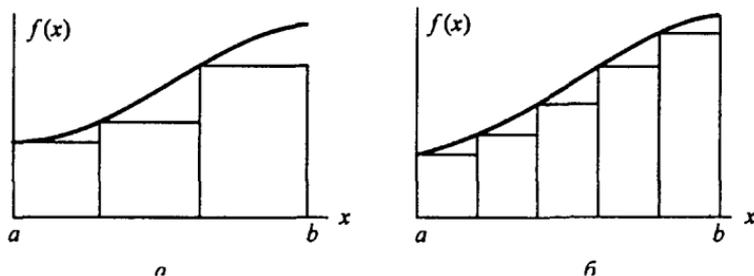


Рис. 4. Иллюстрация метода левых прямоугольников:

$a$  — с 3 участками разбиения отрезка интегрирования  $[a, b]$ ;  
 $b$  — с 6 участками разбиения отрезка интегрирования  $[a, b]$

Метод прямоугольников не находит практического применения в силу значительных погрешностей, что тоже видно из рис. 4.

На рис. 5 приведен пример вычисления интеграла методом трапеций. По сравнению с методом прямоугольников метод трапеций более точный, так как трапеция точнее заменяет соответствующую криволинейную трапецию, чем прямоугольник.

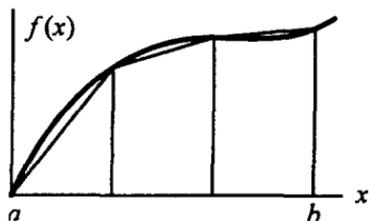


Рис. 5. Иллюстрация метода трапеций

Погрешность  $R$  вычисления интеграла методом трапеций при использовании двойного просчета на практике может быть определена из следующего соотношения:

$$R \leq \frac{|I_n - I_{n/2}|}{3},$$

где  $I_n$  и  $I_{n/2}$  — соответственно значения интеграла при числе разбиений  $n$  и  $n/2$ . Существуют и аналитические выражения для определения погрешности, но они требуют знания второй производной подынтегральной функции, поэтому имеют только теоретическое значение. С использованием двойного просчета можно организовать автоматический подбор шага интегрирования (т.е. числа разбиений  $n$ ) для обеспечения заданной погрешности интегрирования (последовательно удваивая шаг и контролируя погрешность).

### Пример.

Вычислить  $I = \int_0^1 \frac{dx}{1+x^2}$  для всего интервала и с делением интервала на четыре участка.

Аналитическое вычисление данного интеграла дает  $I = \arctg(1) - \arctg(0) = 0,7853981634$ . В нашем случае:

1)  $h = 1, x_0 = 0; x_1 = 1;$

2)  $h = 0,25$  ( $1/4$ ),  $x_0 = 0, x_1 = 0,25, x_2 = 0,5, x_3 = 0,75, x_4 = 1,$

$$f(x) = \frac{1}{1+x^2}.$$

Получим методом левых прямоугольников:

$$1) I \approx (b - a)f(0) = 1 \cdot \frac{1}{1+0} = 1,0;$$

$$2) I \approx \frac{1}{1+0} \cdot 0,25 + \frac{1}{1+0,25^2} \cdot 0,25 + \frac{1}{1+0,5^2} \cdot 0,25 + \\ + \frac{1}{1+0,75^2} \cdot 0,25 = 0,84529412.$$

Получим методом правых прямоугольников:

$$1) I \approx (b - a)f(1) = 1 \cdot \frac{1}{1+1} = 0,5;$$

$$2) I \approx \frac{1}{1+0,25^2} \cdot 0,25 + \frac{1}{1+0,5^2} \cdot 0,25 + \frac{1}{1+0,75^2} \cdot 0,25 + \\ + \frac{1}{1+1^2} \cdot 0,25 = 0,72029412.$$

Получим методом трапеций:

$$1) I \approx (b - a) \frac{f(0) + f(1)}{2} = 1 \cdot \left( \frac{1}{1+0} + \frac{1}{1+1} \right) \cdot \frac{1}{2} = 0,75;$$

$$2) I \approx \frac{1}{1+0} \cdot \frac{0,25}{2} + \frac{1}{1+0,25^2} \cdot 0,25 + \frac{1}{1+0,5^2} \cdot 0,25 + \\ + \frac{1}{1+0,75^2} \cdot 0,25 + \frac{1}{1+1^2} \cdot \frac{0,25}{2} = 0,78279412.$$

### **МЕТОД СИМПСОНА**

Этот метод базируется на замене подынтегральной функции квадратичной параболой, которая строится уже не по двум (как прямая в методе трапеций), а по трем точкам на каждом участке. По этим трем точкам (крайние точки участка и средняя точка) строится интерполяционная функция — полином второго порядка, который аналитически интегрируется. Получается следующая расчетная формула:

• для одного участка интегрирования:

$$I = \int_a^b f(x) dx \approx \frac{h}{3} (f(x_0) + 4f(x_1) + f(x_2)), \quad \omega_0 = \omega_2 = \frac{h}{3}, \quad \omega_1 = \frac{4h}{3},$$

где  $x_0 = a, x_1 = \frac{b+a}{2}, x_2 = b; h = \frac{b-a}{2}$ ;

• для  $n$  участков интегрирования:

$$I = \int_a^b f(x) dx \approx \frac{hf(x_0)}{3} + \frac{4h}{3} \sum_{i=1}^{n/2} f(x_{2i-1}) + \\ + \frac{2h}{3} \sum_{i=1}^{n/2} f(x_{2i}) + \frac{hf(x_n)}{3} = \sum_{i=0}^n \omega_i f(x_i)$$

$$\omega_0 = \omega_n = \frac{h}{3}, \quad \omega_2 = \omega_4 = \dots = \omega_{n-2} = \frac{2h}{3}, \quad \omega_1 = \omega_3 = \dots = \omega_{n-1} = \frac{4h}{3}.$$

В этой формуле все ординаты с нечетными номерами имеют коэффициент  $4h/3$ , а с четными —  $2h/3$  (кроме нулевого и последнего). При работе с этим методом обязательно разбивают весь интервал на четное число участков.

На рис. 6 приведен пример вычисления интеграла методом Симпсона. По сравнению с методами прямоугольников и трапеций он более точный, что наглядно видно из графика (подынтегральная функция почти совпадает с параболой).

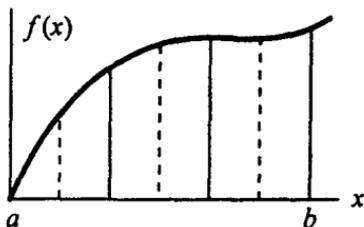


Рис. 6. Иллюстрация метода Симпсона

Метод Симпсона обеспечивает вычисление интеграла точно, без погрешности при полиноме третьего порядка. Следовательно, этот метод предпочтительнее предыдущих. Количественно оценить погрешность при использовании двойного просчета можно по соотношению

$$R \leq \frac{|I_n - I_{n/2}|}{15},$$

т.е. при увеличении числа разбиений в два раза погрешность падает в 15 раз.

Теоретические формулы оценки погрешности содержат производную четвертого порядка от подынтегральной функции, поэтому не имеют практического значения.

### Пример.

Рассмотрим вычисление интеграла из предыдущего раздела. В случае одного участка будем иметь  $x_0 = 0, x_1 = 0,5, x_2 = 1, h = 0,5$ .

$$1) I \approx \frac{0,5}{3} \left( \frac{1}{1+0} + \frac{4 \cdot 1}{1+0,5^2} + \frac{1}{1+1^2} \right) = 0,78333333;$$

$$2) I \approx \frac{1}{1+0} \cdot \frac{0,25}{3} + \frac{1}{1+0,25^2} \cdot \frac{4 \cdot 0,25}{3} + \frac{1}{1+0,5^2} \cdot \frac{2 \cdot 0,25}{3} + \\ + \frac{1}{1+0,75^2} \cdot \frac{4 \cdot 0,25}{3} + \frac{1}{1+1^2} \cdot \frac{0,25}{3} = 0,78539216.$$

## МЕТОД НЬЮТОНА — КОТЕСА

Данный метод является обобщением предыдущих, построен на аналогичных принципах и предполагает замену подынтегральной функции параболой  $k$ -го порядка (а не второго, как в методе Симпсона). Расчетная формула для одного (!) участка выглядит следующим образом:

$$I = \int_a^b f(x) dx \approx (b-a) \sum_{i=0}^k f(x_i) H_i = \sum_{i=0}^k \omega_i f(x_i), \quad \omega_i = (b-a) H_i,$$

где  $x_0 = a, x_k = b, x_i = a + i(b-a)/k, H_i$  — коэффициенты Ньютона — Котеса, а  $k$  — число использующихся ординат на участке (начиная с 0), которые применяются для аппроксимации подынтегральной функции. Естественно, что для замены  $f(x)$  параболой третьей степени потребуется уже четыре точки, а четвертой — пять точек. Коэффициенты  $H_i$  не зависят от функции  $f(x)$  и определены заранее. Некоторые из них приведены ниже.

$$\begin{aligned}
k=1; & H_0 = H_1 = 1/2; \\
k=2; & H_0 = H_2 = 1/6; \quad H_1 = 2/3; \\
k=3; & H_0 = H_3 = 1/8; \quad H_1 = H_2 = 3/8; \\
k=4; & H_0 = H_4 = 7/90; \quad H_1 = H_3 = 16/45; \quad H_2 = 2/15; \\
k=5; & H_0 = H_5 = 19/288; \quad H_1 = H_4 = 25/96; \quad H_2 = H_3 = 25/144. \\
\dots & \quad \dots \quad \dots \quad \dots
\end{aligned}$$

Все предыдущие формулы являются частным случаем формулы Ньютона — Котеса. В частности, при  $k=1$  получаем метод трапеций (для одного участка), при  $k=2$  — формулу Симпсона. Не следует брать  $k > 10$ , так как при больших значениях  $k$  алгоритм оказывается неустойчивым.

При разбиении всего интервала  $[a, b]$  на  $n$  участков формулу нужно применять для каждого участка, а результаты сложить.

### *Пример.*

Рассмотрим вычисление интеграла из предыдущего раздела. Выберем  $k=4$ .

$$\begin{aligned}
1) I \approx \frac{7}{90} f(0) + \frac{16}{45} f(0,25) + \frac{2}{15} f(0,5) + \frac{16}{45} f(0,75) + \\
+ \frac{7}{90} f(1) = 0,7855294;
\end{aligned}$$

2) в этом случае разобьем интервал на четыре участка ( $n=4$ ) с  $h=0,25$ , применим для каждого интервала формулу с  $k=4$ , сложим результаты и получим  $I \approx 0,78539817$ .

## **МЕТОДЫ ЧЕБЫШЕВА И ГАУССА**

Все предыдущие методы имели следующую особенность: значения  $x$  располагались равномерно, а весовые коэффициенты были разными (в общем случае, хотя некоторые из них были равны друг другу). В методе Чебышева приняты все весовые коэффициенты одинаковыми, а  $x_i$  — разными. Предварительно при использовании приведенных ниже формул метода следует преобразовать переменную интегрирования, приведя ее к диапазону  $[-1, 1]$  следующим образом:

$$x = \frac{a+b}{2} + \frac{b-a}{2} z.$$

Расчетные формулы получаются для различных значений  $k$  (число ординат, использующихся при расчетах на одном участке) исходя из обеспечения возможности интегрирования без ошибки полинома как можно более высокой степени. Оказывается, можно интегрировать полином без ошибки при  $k = 2, 3, 4, 5, 6, 7, 9$ . Для этого уже рассчитаны необходимые параметры. Часть из них приведена ниже.

$$\begin{array}{llll}
 k = 2; & -z_1 = z_2 = 0,577350; & & \\
 k = 3; & -z_1 = z_3 = 0,707107; & z_2 = 0; & \\
 k = 4; & -z_1 = z_4 = 0,794654; & -z_2 = z_3 = 0,187592; & \\
 k = 5; & -z_1 = z_5 = 0,832498; & -z_2 = z_4 = 0,374541; & z_3 = 0. \\
 \dots & \dots & \dots & \dots
 \end{array}$$

С учетом преобразования переменной формула интегрирования будет выглядеть так:

$$I = \int_a^b f(x) dx = \int_{-1}^1 f(z) dz \approx \frac{b-a}{k} \sum_{i=1}^k f(z_i).$$

В методе Гаусса в отличие от метода Чебышева все  $z_i$  — разные и все весовые коэффициенты  $\omega_i$ . Это позволило обеспечить интегрирование без ошибки уже полинома степени  $2k - 1$ , что и для любых других, не полиномиальных подынтегральных функций дает лучшие результаты (т.е. меньшую ошибку). Некоторые значения параметров формулы интегрирования приведены ниже.

$$\begin{array}{llll}
 k = 2; & -z_1 = z_2 = 0,577350; & \omega_1 = \omega_2 = 1; & \\
 k = 3; & -z_1 = z_3 = 0,774597; & \omega_1 = \omega_3 = 0,555555; & \\
 & z_2 = 0; & \omega_2 = 0,888889; & \\
 k = 4; & -z_1 = z_4 = 0,861136; & \omega_1 = \omega_4 = 0,347855; & \\
 & -z_2 = z_3 = 0,339981; & \omega_2 = \omega_3 = 0,652145. &
 \end{array}$$

С учетом преобразования переменной формула интегрирования методом Гаусса будет выглядеть так:

$$I = \int_a^b f(x) dx = \int_{-1}^1 f(z) dz \approx \frac{b-a}{2} \sum_{i=1}^k \omega_i f(z_i).$$

Следует отметить, что в этих методах вследствие неравномерного шага интегрирования нельзя оценить погрешность интегрирования двойным просчетом. Для этой цели применяются другие достаточно сложные алгоритмы.

### *Пример.*

Рассмотрим задачу, аналогичную ранее рассмотренным в этом разделе. Для сокращения решим ее только для всего интервала сразу, не разбивая на участки. Примем  $k = 4$ . Помним, что  $f(x) = 1/(1+x^2)$  и  $z(x) = (0+1)/2 + x(1-0)/2$ , соответственно будет  $x = x(z) = 2z - 1$ . Для метода Чебышева получим:

$$x_1 = x(-0,794654); \quad x_2 = x(-0,187592); \quad x_3 = x \cdot 0,187592;$$
$$x_4 = x \cdot 0,794654;$$

$$I \approx \frac{1-0}{4} (f(x_1) + f(x_2) + f(x_3) + f(x_4)) = 0,78530324.$$

Для метода Гаусса получим:

$$x_1 = x(-0,861136); \quad x_2 = x(-0,339981); \quad x_3 = x \cdot 0,339981;$$
$$x_4 = x \cdot 0,861136;$$

$$I \approx \frac{1-0}{2} (0,347855f(x_1) + 0,652145f(x_2) +$$
$$+ 0,652145f(x_3) + 0,347855f(x_4)) = 0,78540298.$$

Если сравнить полученные результаты с аналогичными для других методов при интегрировании всего участка сразу, то можно убедиться, что последние методы обладают более высокой точностью, хотя и требуют более сложных вычислений.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

### *Простейшие методы вычисления интегралов*

1. Как в методе прямоугольников уменьшить погрешность нахождения интеграла?
2. В каких случаях метод прямоугольников находит применение?
3. Как уменьшить в методе трапеций погрешность нахождения интеграла?

4. В каких случаях метод трапеций находит применение?
5. Можно ли получить методами прямоугольников и трапеций точное значение интеграла?

### ***Метод Симпсона***

1. Какой аппроксимирующей заменяется подынтегральная функция в методе Симпсона?
2. Если для построения аппроксимирующей функции средняя точка берется не в середине участка, то что изменится в алгоритме?
3. Обязательно ли участок интегрирования разбивать при реализации метода на более мелкие участки?
4. Дана подынтегральная функция  $f(x) = x + 7$ , с каким методом совпадет метод Симпсона?
5. Почему метод Симпсона использует аппроксимацию подынтегральной функции квадратичной параболой, а способен интегрировать без ошибки и кубические параболы?

### ***Метод Ньютона — Котеса***

1. Являются ли постоянными весовые коэффициенты в слагаемых в формуле Ньютона — Котеса?
2. Может ли результат, полученный методом Ньютона — Котеса, совпасть с результатом, полученным методом с более низкой точностью, например методом левых прямоугольников?
3. Может ли подынтегральная функция в методе Ньютона — Котеса аппроксимироваться полиномом второй степени?
4. Какой наивысшей степени полином может использоваться для замены подынтегральной функции в методе Ньютона — Котеса?
5. Могут ли точки при интегрировании располагаться неравномерно?

### ***Методы Чебышева и Гаусса***

1. Зачем осуществляют преобразование исходного интервала интегрирования к диапазону  $(-1, 1)$ ?

2. Почему нельзя для оценки погрешности пользоваться приемом двойного просчета?
3. Почему метод Гаусса дает более высокую точность вычисления интеграла, чем метод Чебышева?
4. Можно ли пользоваться автоматическим подбором шага при использовании метода Гаусса?
5. Для какого числа точек на отрезке интегрирования работают методы?

### *Общие вопросы численного интегрирования*

1. Дана подынтегральная функция  $f(x) = 1500x$ . Какой из методов будет наиболее эффективен?
2. В каких случаях можно пользоваться автоматическим подбором шага интегрирования?
3. Дана подынтегральная функция  $f(x) = x^2$ . Можно ли каким либо численным методом вычислить интеграл без ошибки?
4. Дана подынтегральная функция  $f(x) = 5x^3$ . Какой из методов даст наиболее точный результат?
5. Как изменяется погрешность нахождения интеграла при уменьшении числа разбиений  $n$ ?

---

# МЕТОДЫ РЕШЕНИЯ УРАВНЕНИЙ И ИХ СИСТЕМ

---

## МЕТОДЫ РЕШЕНИЯ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

---

Многие задачи исследования различных объектов с помощью математических моделей, применения их для прогноза или расчета приводят к необходимости решения нелинейных уравнений. Поэтому в данном пособии этому разделу уделено достаточное внимание.

### КОНЦЕПЦИЯ МЕТОДОВ

Как правило, процесс решения нелинейного уравнения общего вида  $f(x) = 0$  осуществляется в два этапа. На первом этапе отделяют корни, т.е. находят такие отрезки, внутри которых находится строго один корень. На втором этапе уточняют корень, т.е. находят его значение  $x^*$  с предварительно заданной точностью  $\varepsilon$ . В практических задачах решением называют любое значение  $x$ , отличающееся по модулю от точного значения  $x^*$  не более чем на величину  $\varepsilon$ .

Идеи аналитических методов первого этапа базируются на очевидном свойстве непрерывных функций: корни функции (точки пересечения  $f(x)$  с горизонтальной осью) обязательно лежат между соседними экстремумами функции (хотя обратное неверно: между каждой парой экстремумов необязательно находится корень).

Идеи методов второго этапа можно сгруппировать по трем основным направлениям. В первом — поиск корня с заданной погрешностью сводится к перебору всех возможных значений аргумента с проверкой наличия решения. Во втором — поиск корня нелинейной функции заменяется поиском корня той или иной более простой функции (линейной, параболической), близкой к ис-

ходной нелинейной; как правило, процесс поиска осуществляется итерационными процедурами (однотипными, последовательно повторяющимися). В третьем — нелинейное уравнение вида  $f(x)=0$  сводят к одной из форм вида  $g(x)=\varphi(x)$  и стремятся обеспечить равенство левой и правой частей тоже, как правило, с помощью итерационных процедур.

Условием окончания процесса решения уравнения (т.е. получения корня  $x^*$  с заданной погрешностью) может быть одно из двух возможных: 1)  $|f(x^*)| \leq \delta$ , 2)  $|x^k - x^{k+1}| \leq \varepsilon$ , где  $\delta, \varepsilon$  — предварительно заданные малые величины,  $k$  — номер итерации, т.е. или близость к нулю левой части уравнения, или близость друг к другу двух значений  $x$ , между которыми находится решение. Второе условие во многих случаях можно использовать, не зная точного значения корня, путем замены его другим, например  $|x^{k+1} - x^k| \leq \varepsilon'$ , при выполнении которого данное условие будет гарантированно выполняться. Условие окончания поиска выбирается исходя из неформальных соображений, и в некоторых случаях применение разных условий может привести к существенно разным результатам. При решении конкретных задач в математическом моделировании важными являются две цели решения:

1) обеспечение близости к нулю  $f(x)$  ( $f(x) \approx 0$ ) как меры выполнения тех или иных балансовых соотношений, тогда не очень важно, при каких именно (в пределах здравого смысла конкретной прикладной задачи) значениях  $x$  это равенство справедливо с заданной погрешностью;

2) обеспечение точности нахождения решения  $x^*$ , имеющего содержательное значение, при этом  $f(x) \approx 0$  является лишь индикатором правильности решения. Отсюда и выбирают условие окончания поиска решения.

Знание особенностей левой части нелинейного уравнения позволяет в ряде случаев, не производя отделения корней, определить число корней (причем отдельно действительных и комплексных), что невозможно в общем случае, а также предельные оценки корней, интервалы существования корней. Это прежде всего касается алгебраических уравнений с действительными коэффициентами (далее для простоты — алгебраических), часто

встречающихся в практике. Такие уравнения имеют вид

$$f(x) = a_0x^n + a_1x^{n-1} + a_2x^{n-2} + \dots + a_n = 0.$$

Кроме того, с учетом конкретного вида уравнения можно построить более эффективные алгоритмы.

## ОТДЕЛЕНИЕ КОРНЕЙ

Отделение корней может производиться графически (путем построения графика функции  $f(x)$ ) или аналитически. Для аналитического отделения корней находят все критические точки функции  $f(x)$ , т.е. точки, в которых производные равны нулю или не существуют. Это можно сделать численными методами или — в несложных случаях — аналитически. Для этого  $f(x)$  дифференцируют, приравнивают производную к нулю и решают полученное уравнение относительно  $x$ . Кроме того, определяют все точки, где по тем или иным причинам (например, знаменатель обращается в нуль, под логарифмом появляется нуль и т.п.) производная может не существовать. В этих (критических) точках или в непосредственной близости от них определяют знак функции  $f(x_i)$ , т.е. находят  $\text{sign } f(x_i)$ . Затем строят ряд знаков функции в критических точках, включая в рассмотрение и крайние точки числовой оси  $-\infty$  и  $+\infty$ . Анализируют этот ряд, и по числу смен знаков определяют количество корней (равно числу смен знаков  $\text{sign } f(x_i)$ ) и интервалы, где локализованы эти корни: на левой и на правой границах такого интервала функция  $f(x)$  должна иметь разные знаки. В случае необходимости можно дополнительно к критическим точкам использовать и произвольные точки, что позволяет сузить интервал локализации корня. Особенно это надо делать, когда одна из границ интервала находится в бесконечности, так как интервал хотя бы с одной границей в бесконечности не позволит уточнить корни.

### *Пример 1.*

Дано уравнение:  $5^x - 6x - 3 = 0$ .

Обозначим  $f(x) = 5^x - 6x - 3$ . Находим производную  $f'(x) = 5^x \ln 5 - 6$ . Вычислим корень производной:

$$5^x \ln 5 - 6 = 0; \quad 5^x = \frac{6}{\ln 5}; \quad x \lg 5 = \lg 6 - \lg(\ln 5);$$

$$x = \frac{\lg 6 - \lg(\ln 5)}{\lg 5} = \frac{0,7782 - 0,2065}{0,6990} = \frac{0,5717}{0,6990} \approx 0,82.$$

Составим таблицу знаков функции  $f(x)$ , полагая  $x$  равным: а) критическим значениям функции (корням производной) или близким к ним; б) граничным значениям (исходя из области допустимых значений неизвестного):

$x$	$-\infty$	1	$+\infty$
$\text{sign } f(x)$	+	-	+

Так как происходят две перемены знака функции, то уравнение имеет два действительных корня. Чтобы завершить операцию отделения корней, можно попытаться уменьшить промежутки, содержащие корни, так, чтобы их длина была бы меньше. Для этого возьмем несколько значений  $x$  и внесем их дополнительно в таблицу. Получим следующую таблицу:

$x$	$-\infty$	-1	0	1	2	$+\infty$
$\text{sign } f(x)$	+	+	-	-	+	+

Отсюда видно, что корни заключены в следующих промежутках:  $x_1 \in [-1, 0]$ ;  $x_2 \in [1, 2]$ .

### Пример 2.

Дано уравнение:  $x^4 - x^3 - 2x^2 + 3x - 3 = 0$ .

Обозначим  $f(x) = x^4 - x^3 - 2x^2 + 3x - 3$ , тогда имеем  $f'(x) = 4x^3 - 3x^2 - 4x + 3$ . Найдем корни производной:

$$4x^3 - 3x^2 - 4x + 3 = 0; \quad 4x(x^2 - 1) - 3(x^2 - 1) = 0; \quad (x^2 - 1)(4x - 3) = 0;$$

$$x_1 = -1; \quad x_2 = 1; \quad x_3 = 3/4.$$

Составим таблицу знаков функции  $f(x)$ :

$x$	$-\infty$	$-1$	$3/4$	$1$	$+\infty$
$\text{sign } f(x)$	$+$	$-$	$-$	$-$	$+$

Из таблицы видно, что уравнение имеет два действительных корня:  $x_1 \in ]-\infty, -1]$ ;  $x_2 \in [1, +\infty[$ .

Уменьшим промежутки, в которых находятся корни:

$x$	$-\infty$	$-2$	$-1$	$3/4$	$1$	$2$	$+\infty$
$\text{sign } f(x)$	$+$	$+$	$-$	$-$	$-$	$+$	$+$

Следовательно,  $x_1 \in [-2, -1]$ ;  $x_2 \in [1, 2]$ .

## УТОЧНЕНИЕ КОРНЕЙ

Рассмотрим методы уточнения корней и раскрывающие их основные идеи. Отметим очевидный момент: при прочих равных условиях тот метод уточнения корней будет более эффективен, в котором результат с той же погрешностью найден за меньшее число раз вычисления функции  $f(x)$ .

### МЕТОД СКАНИРОВАНИЯ

Метод предусматривает разделение всего интервала  $[a, b]$ , где отделен корень, на маленькие отрезки, равные заданной погрешности  $\epsilon$ , с последующим вычислением (или определением экспериментально) значений функции  $f(x)$  на концах этих отрезков (т.е. в точках, расстояние между которыми не превышает величины  $\epsilon$ ). Анализируя значения функции, нетрудно выбрать отрезок, где функция меняет знак (или точно равна нулю, что маловероятно). В качестве решения можно взять любую точку — левую ( $x_i$ ) или правую ( $x_{i+1}$ ) границу выделенного отрезка, хотя предпочтительнее взять середину этого отрезка  $x^* = (x_i + x_{i+1})/2$ . В любом случае погрешность решения не будет превышать заданную погрешность  $\epsilon$ , даже при условии, что мы не знаем точного значения решения.

Иногда весь отрезок разбивают на маленькие отрезки величиной  $2\epsilon$ , а затем искомое значение корня берут в середине отрезка, где функция меняет знак. Это непринципиальная разница с основ-

ным вариантом, результаты вариантов полностью совпадут и по значению корня, и по затратам на поиск, если в первом сразу взять погрешность вдвое больше необходимой.

Для повышения эффективности метода можно уточнение производить в несколько этапов. На первом этапе задать большое значение  $\epsilon$ , найти отрезок, где функция меняет знак (грубо найти корень), затем найденный отрезок еще раз разделить с более мелким шагом, более точно найти корень и т.д. еще несколько этапов (обычно 3...5), после чего удастся найти корень с заданной погрешностью в целом за меньшее число раз вычисления  $f(x)$ . Метод очевиден и не требует практического пояснения.

### **МЕТОД ДЕЛЕНИЯ ОТРЕЗКА ПОПОЛАМ**

Этот метод можно рассматривать как развитие метода сканирования: величина отрезков, на которые делится весь интервал при многоэтапном применении метода сканирования, становится равной половине исходного отрезка  $[a, b]$ . В этом случае сначала исходный отрезок делится на две равные части (пополам) и путем сравнения знаков функции на концах каждой из двух половинок (например, по знаку произведения значений функций на концах) определяют ту половинку, в которой содержится решение (знаки функции на концах должны быть разные). Затем найденную половинку опять делят на две равные части, снова выбирают одну из двух половинок, содержащую корень, и т.д. Условием окончания служит заданная малость отрезка, где содержится корень, т.е. аналогично методу сканирования. Существуют и более эффективные алгоритмы, например, выбор точки не в середине отрезка, а ближе к тому краю, в котором значение функции меньше. Но этот вариант не будет работать при немонотонной функции.

Метод половинного деления, как и метод сканирования, очевиден и не требует практического пояснения.

### **МЕТОД ХОРД**

В этом методе нелинейная функция  $f(x)$  на отделенном интервале  $[a, b]$  заменяется линейной, в качестве которой берется хорда — прямая, стягивающая концы нелинейной функции. Эта

хорда определяется как прямая, проходящая через точки с координатами  $(a, f(a))$  и  $(b, f(b))$ . Имея уравнение хорды  $y = cx + d$ , можно легко найти точку ее пересечения с горизонтальной осью, подставив в уравнение  $y = 0$  и найдя из него  $x$ . Естественно, в полученной таким путем точке  $x_1$  не будет решения, ее принимают за новую границу отрезка, где содержится корень. Через эту точку с координатами  $(x_1, f(x_1))$  и соответствующую границу предыдущего интервала опять проводят хорду, находят  $x_2$  и т.д. несколько раз, получая последовательность  $x_3, x_4, x_5, \dots$ , сходящуюся к корню. Метод применим только для монотонных функций.

Алгоритм метода зависит от свойств функции  $f(x)$ . Если  $f(b)f''(b) > 0$ , то строящаяся на каждом этапе хорда имеет правый фиксированный ("закрепленный") конец, и алгоритм выглядит следующим образом:

$$x_{i+1} = x_i - \frac{f(x_i)}{f(b) - f(x_i)}(b - x_i);$$

при этом последовательность  $x_1, x_2, \dots$  будет приближаться к корню слева.

Если  $f(a)f''(a) > 0$ , то строящаяся на каждом этапе хорда имеет левый фиксированный ("закрепленный") конец, и алгоритм выглядит следующим образом:

$$x_{i+1} = a + \frac{f(a)}{f(a) - f(x_i)}(x_i - a);$$

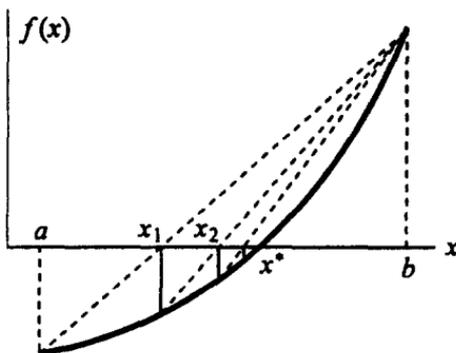


Рис. 7. Иллюстрация метода хорд

при этом последовательность  $x_1, x_2, \dots$  будет приближаться к корню справа.

На рис. 7 приведен один из вариантов применения метода хорд. В рассматриваемом случае "закрепленным" является правый конец. Приведено пять шагов (пять хорд), при этом к решению приближаемся слева.

Теоретически доказано, что если первые производные на концах интервала при монотонной и выпуклой функции  $f(x)$  не различаются более чем в 2 раза, то справедливо соотношение  $|x^* - x_i| < |x_i - x_{i-1}|$  и условием прекращения пополнения последовательности может быть  $|x_{i+1} - x_i| \leq \varepsilon$ , а в качестве корня принято  $x_{i+1}$  (можно также окончить процесс и при достижении  $f(x_i) \leq \delta$ , о чем указывалось в концепции методов). На практике указанные условия можно применять и без предварительной проверки производных, отклонение погрешности результата при пологих функциях не будет существенным.

### *Пример.*

Имеем уравнение  $x^3 - 0,2x^2 + 0,5x + 1,5 = 0$ . Уточнить корень с погрешностью  $\varepsilon < 0,001$ .

Запишем  $f(x) = x^3 - 0,2x^2 + 0,5x + 1,5$ .

Проведя процедуру отделения корней, получим, что корень находится в промежутке  $[-1, 0]$ , т.е.  $a = -1$ ,  $b = 0$ .

Находим вторую производную  $f''(x) = 6x - 0,4$ ; в промежутке  $[-1, 0]$  выполняется неравенство  $f''(x) < 0$ , поэтому для вычислений применяем формулу

$$x_{i+1} = a + \frac{f(a)}{f(a) - f(x_i)}(x_i - a),$$

где  $x_0 = b = 0$ ;  $f(a) = f(-1) = -1 - 0,2 - 0,5 + 1,5 = -0,2$ . Все вычисления сведены в табл. 2.

Таблица 2

$i$	$x_i$	$f(x_i)$	$x_i - a$
0	0	1,5	1
1	-0,882	0,2173	0,118
2	-0,943	0,0121	0,057
3	-0,946	0,0014	0,054
4	-0,946		

Ответ:  $x \approx -0,946$

## МЕТОД НЬЮТОНА (КАСАТЕЛЬНЫХ)

Идея, на которой основан метод, аналогична той, которая реализована в методе хорд, только в качестве прямой берется касательная, проводимая в текущей точке последовательности. Уравнение касательной находится по координате одной точки и углу наклона (значение производной). В качестве начальной точки в зависимости от свойств функции берется или левая точка  $x_0 = a$  (если  $f(a)f''(a) > 0$ ), или правая точка  $x_0 = b$  (если  $f(b)f''(b) > 0$ ). Алгоритм записывается следующим образом:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}.$$

Алгоритм работоспособен при выпуклых и монотонных функциях  $f(x)$ . Главным теоретическим достоинством метода является квадратичная скорость сходимости, что во многих случаях может привести к сокращению числа вычислений функции при получении решения с заданной погрешностью. В ряде случаев можно применять упрощенный алгоритм, связанный с сокращением числа раз вычисления производных — вместо вычисления производной в каждой очередной точке  $f'(x_i)$  использовать значение производной в начальной точке  $f'(x_0)$ . Следует обратить внимание на следующую особенность метода: последователь-

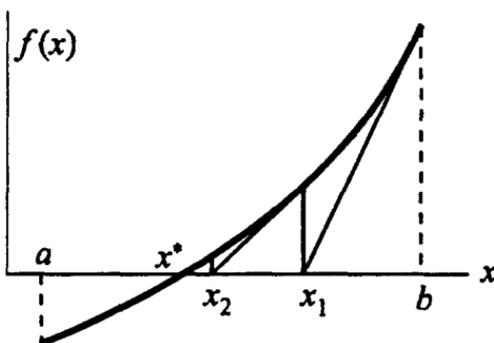


Рис. 8. Иллюстрация метода Ньютона

ность  $x_1, x_2, x_3, \dots$  приближается к корню с другой стороны, чем при использовании метода хорд при прочих равных условиях.

На рис. 8 приведен один из вариантов применения метода Ньютона. В рассматриваемом случае процесс начинается с правого конца. К решению приближаемся справа. Условия окончания поиска аналогичны методу хорд.

### Пример.

Имеем уравнение  $\operatorname{tg}(0,55x + 0,1) - x^2 = 0$ . Уточнить корень с погрешностью  $\varepsilon < 0,001$ . Запишем  $f(x) = \operatorname{tg}(0,55x + 0,1) - x^2$ .

Проведя процедуру отделения корней, получим, что корень находится в промежутке  $[0,6, 0,8]$ , т.е.  $a = 0,6, b = 0,8$ .

Так как  $f(0,6) > 0, f(0,8) < 0$  и  $f''(x) < 0$ , то за начальное приближение примем  $x_0 = 0,8$ , а вычисления будем проводить по формуле

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_0)}.$$

$$\begin{aligned} \text{Предварительно найдем } f'(x_0) &= \frac{0,55}{\cos^2(0,44 + 0,1)} - 2 \cdot 0,8 = \\ &= \frac{0,55}{0,85772^2} - 1,6 = \frac{0,55}{0,7356} - 1,6 = 0,7477 - 1,6 = -0,8523. \end{aligned}$$

Составим таблицу (табл. 3):

Таблица 3

$i$	$x_i$	$f(x_i)$
0	0,8	-0,0406
1	0,7524	-0,0018
2	0,7503	-0,0000

Ответ:  $x \approx 0,750$ .

## КОМБИНИРОВАННЫЙ МЕТОД

Данный метод, так же как и предыдущие, базируется на замене нелинейной функции  $f(x)$  линейной, но с учетом стремления к корню метода хорд и метода Ньютона с разных сторон; для повышения эффективности использует оба алгоритма одновременно. Один шаг делается методом хорд, а следующий с другой стороны — методом Ньютона. При этом интервал, где содержится корень, сокращается с обеих сторон, что обуславливает другое условие окончания поиска. Поиск можно прекратить, как только разница между правым и левым концами интервала станет меньше предварительно заданной погрешности  $\varepsilon$ .

Алгоритмы используемых методов следует выбирать с учетом упомянутых выше особенностей функций.

**Пример.**

Имеем уравнение  $x^3 - 2x^2 - 4x + 7 = 0$ . Уточнить корень с погрешностью  $\epsilon < 0,001$ .

Проведя процедуру отделения корней, получим, что уравнение имеет три действительных корня:  $x_1 \in [-2, -1]$ ;  $x_2 \in [1, 2]$ ;  $x_3 \in [2, 3]$ .

Для примера рассмотрим уточнение корня  $x_1$ . Учитывая, что  $f(-2) < 0$ ;  $f(-1) < 0$ ;  $f'(x) = 6x - 4$  и при  $-2 \leq x \leq -1$   $f'(x) < 0$ , для расчетов примем следующие формулы:

$$x_{n(i+1)} = x_{li} - \frac{f(x_{li})}{f'(x_{li})}; \quad x_{n(i+1)} = x_{li} - \frac{f(x_{li})}{f(x_{ni}) - f(x_{li})}(x_{ni} - x_{li}),$$

где  $x_{li}$  и  $x_{ni}$  — соответственно значение корня по недостатку (слева) и избытку (справа);  $x_{l0} = -2$ ,  $x_{n0} = -1$ .

Все промежуточные результаты вычислений сведем в табл. 4.

Таблица 4

$i$	$x_{li}$	$x_{ni}$	$x_{ni} - x_{li}$	$f(x_{li})$	$f(x_{ni})$
0	-2	-1	1	-1	8
1	-1,9400	-1,8900	0,0500	-0,0686	0,6645
2	-1,9355	-1,9353	0,0002	-0,0011	0,0020

Ответ:  $x_1 \approx -1,935$ .

**МЕТОД ПАРАБОЛИЧЕСКОЙ АППРОКСИМАЦИИ**

В этом методе функция  $f(x)$  заменяется не линейной, а параболической функцией, что является более точной заменой. Следовательно, метод может обеспечить более быструю сходимость к решению. На первом этапе параболу обычно строят по трем точкам: крайним и средней точкам интервала  $(a, b)$ , где отделен корень, т.е.  $(a, f(a))$ ,  $((a + b)/2, f((a + b)/2))$ ,  $(b, f(b))$ . По

полученному уравнению параболы  $y = c_2x^2 + c_1x + c_0$  находят приближенный корень (приближенный — потому что парабола приближенно заменяет  $f(x)$ ), для чего решают уравнение  $c_2x^2 + c_1x + c_0 = 0$ . На втором этапе строят параболу по трем точкам: найденному приближенному корню и двум предыдущим точкам (слева и справа от этой точки), лежащим по разные стороны оси  $x$ . Такой вариант выбора точек на практике быстрее приводит к решению по сравнению с вариантом, когда для построения параболы берутся последовательно три последние точки. Такая процедура повторяется многократно до тех пор, пока величина отрезка, внутри которого находится корень, не будет меньше  $\epsilon$  — предварительно заданной погрешности.

На рис. 9 приведен один шаг уточнения корня методом параболической аппроксимации. Начальная парабола проведена через точки  $a$ ,  $x_1$ ,  $b$  (здесь  $x_1$  является серединой отрезка  $[a, b]$ );  $x_2$  — пересечение параболы с осью. Следующая парабола должна проводиться через точки  $x_1$ ,  $x_2$ ,  $b$ .

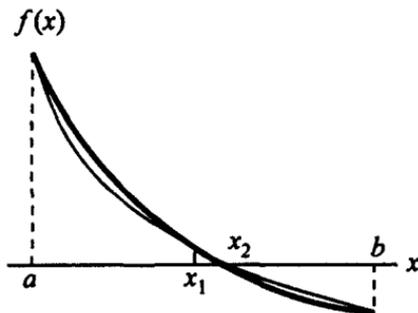


Рис. 9. Иллюстрация первого этапа метода параболической аппроксимации

### Пример.

Имеем уравнение  $x2^x - 1 = 0$ . Сделать две итерации методом параболической аппроксимации.

Запишем  $f(x) = x2^x - 1$ . Проведя процедуру отделения корней, получим, что корень находится в промежутке  $[0, 1]$ , т.е.  $a = 0$ ,  $b = 1$ .

Выбираем среднюю точку интервала  $x = 1/2$  и вычисляем значения функции в этих точках:

$$x = 0, f(0) = -1; \quad x = 0,5, f(0,5) \approx -0,293; \quad x = 1, f(1) = 1.$$

Через эти три точки и будем проводить параболу  $y = c_2x^2 + c_1x + c_0$ . Для того чтобы парабола проходила через заданные

точки, коэффициенты  $c_i$  должны удовлетворять следующим уравнениям:

- $0^2 \cdot c_2 + 0 \cdot c_1 + c_0 = -1$  (условие прохождения через первую точку);
- $\left(\frac{1}{2}\right)^2 c_2 + \frac{1}{2} c_1 + c_0 = -0,293$  (условие прохождения через вторую точку);
- $1^2 \cdot c_2 + 1 \cdot c_1 + c_0 = 1$  (условие прохождения через третью точку).

Решая эту систему, находим коэффициенты:

$$c_0 = -1; \quad c_1 = -0,828; \quad c_2 = 1,172.$$

Для нахождения приближенного значения корня решим уравнение  $1,172x^2 + 0,828x - 1 = 0$ , что на участке  $(0, 1)$  дает корень  $x = 0,636$ . Найдем значение функции в этой точке:

$$f(0,636) = 0,636 \cdot 2^{0,636} - 1 = -0,011647.$$

Полученное значение функции отлично от нуля, так как парабола приблизительно описывает исходную нелинейную функцию.

Используя три точки:  $(0,5, -0,293)$ ;  $(0,636, -0,011647)$ ;  $(1, 1)$ , построим новую параболу. Коэффициенты этой параболы будут:

$$c_0 = -1; \quad c_1 = 0,828; \quad c_2 = 1,172.$$

### **МЕТОД ПРОСТОЙ ИТЕРАЦИИ**

Рассматриваемый метод реализует третий подход из представленных в концепции. Предварительно исходное уравнение  $f(x) = 0$  преобразуют к виду  $\varphi(x) = x$ , что является частным случаем более общей структуры  $g(x) = f(x)$ . Затем выбирают начальное значение  $x_0$  и подставляют его в левую часть уравнения, но  $\varphi(x_0) \neq x_0$ , поскольку  $x_0$  взято произвольно и не является корнем уравнения. Полученное  $\varphi(x_0) = x_1$  рассматривают как очередное приближение к корню. Его снова подставляют в левую часть уравнения  $\varphi(x_1)$  и получают следующее значение  $x_2$  ( $x_2 = \varphi(x_1)$ ) и т.д., в общем случае  $x_{i+1} = \varphi(x_i)$ . Получающаяся таким образом

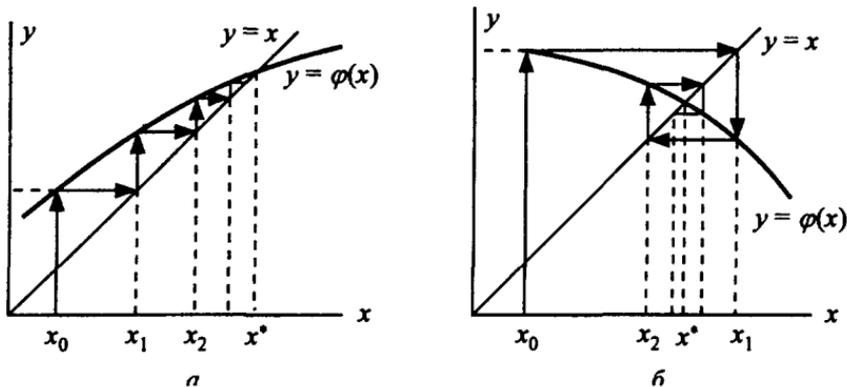


Рис. 10. Иллюстрация метода итераций для различных ситуаций:  
 $a — 0 < \varphi'(x) < 1$ ;  $b — -1 < \varphi'(x) < 0$

последовательность  $x_0, x_1, x_2, x_3, x_4, \dots$  при определенных условиях может сходиться к корню  $x^*$  (рис. 10).

Таким условием является  $|\varphi'(x)| \leq 1$  на  $[a, b]$ , причем чем ближе модуль к нулю, тем выше окажется скорость сходимости к решению. В противном случае последовательность расходится от искомого решения (“метод не сходится”).

На рис. 11 приведен один из возможных случаев, когда итерационный процесс не сходится. Видно, что последовательность  $x_0, x_1, x_2, \dots$  удаляется от корня  $x^*$ . Это всегда будет иметь место в том случае, если тангенс угла наклона  $\varphi(x)$  в окрестности корня по модулю больше единицы.

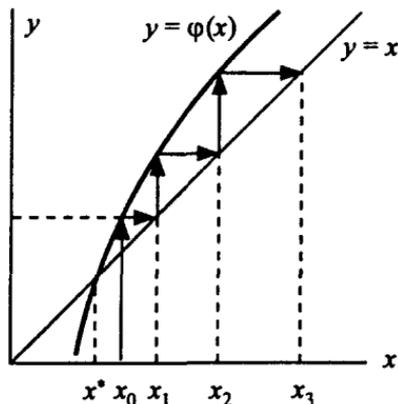


Рис. 11. Иллюстрация несходящегося итерационного процесса

Существуют различные способы преобразования уравнения  $f(x) = 0$  к виду  $\varphi(x) = x$ ; одни могут привести к выполнению усло-

вия сходимости всегда, другие — в отдельных случаях. Самый простой способ следующий:

$$f(x) + x = 0 + x, \quad f(x) + x = \varphi(x) \Rightarrow \varphi(x) = x,$$

но он не всегда приводит к успеху. Существует другой способ, в соответствии с которым  $\varphi(x) = x - f(x)/k$ , причем  $k$  следует выбирать так, чтобы  $|k| > Q/2$ , где  $Q = \max_{[a, b]} |f'(x)|$  и знак  $k$  совпадал бы со знаком  $f'(x)$  на  $[a, b]$ .

Погрешность решения можно оценить из соотношения

$$|x^* - x_i| \leq \frac{q}{1-q} |x_i - x_{i+1}|, \quad \text{где } q = \max_{[a, b]} \varphi'(x).$$

Вследствие этого для окончания вычислений в методе итераций применяют соотношение  $\frac{q}{1-q} |x_i - x_{i+1}| \leq \varepsilon$ , где  $\varepsilon$  — заданная погрешность решения.

Часто используют упрощенное условие окончания поиска  $|x_i - x_{i+1}| \leq \varepsilon$ , не вычисляя максимальное значение производной, но в этом случае погрешность решения может не соответствовать заданной (т.е. быть больше или меньше).

### *Пример.*

Имеем уравнение  $2x + \lg(2x + 3) = 1$ . Необходимо уточнить корень с погрешностью  $\varepsilon < 0,001$ .

Запишем  $f(x) = 2x + \lg(2x + 3) - 1$ . Проведя процедуру отделения корней, получим, что корень находится в промежутке  $[0, 0,5]$ , т.е.  $a = 0, b = 0,5$ . Приведем уравнение к виду, удобному для итераций  $\varphi(x) = x$ . Функцию  $\varphi(x)$  будем искать из соотношения  $\varphi(x) = x - f(x)/k$ , считая для повышения сходимости, что  $|k| \geq Q/2$ , где  $Q = \max |f'(x)|$ ; число  $k$  имеет тот же знак, что и  $f'(x)$  в промежутке  $[0, 0,5]$ .

Находим

$$f'(x) = 2 + \frac{0,8686}{2x + 3}; \quad Q = \max_{[0, 0,5]} f'(x) = 2 + \frac{0,8686}{2 \cdot 0 + 3} \approx 2,2895; \quad f'(x) > 0$$

при  $0 \leq x \leq 0,5$ .

Примем  $k = 2$ , тогда  $\varphi(x) = x - f(x)/2 = 0,5 - 0,5 \lg(2x + 3)$ .

За начальное приближение возьмем  $x_0 = 0$ , все остальные приближения будем определять из равенства  $x_{i+1} = 0,5 - 0,5 \lg(2x_i + 3)$ , результаты сведем в табл. 5.

Таблица 5

$i$	$x_i$	$\varphi(x_i) = x_{i+1}$
0	0	0,2614
1	0,2614	0,2266
2	0,2266	0,2309
3	0,2309	0,2303
4	0,2303	0,2304
5	0,2304	

Ответ:  $x \approx 0,230$ .

## ОПРЕДЕЛЕНИЕ ЧИСЛА КОРНЕЙ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ

Для уравнения  $P_n(x) = a_0x^n + a_1x^{n-1} + a_2x^{n-2} + \dots + a_n = 0$  общее количество действительных корней и комплексных корней (которые всегда в рассматриваемом случае при действительных коэффициентах уравнения будут попарно сопряженными) всегда равно наивысшей степени полинома в левой части уравнения, т.е. в нашем случае —  $n$ .

Существует правило Декарта, в соответствии с которым можно оценить отдельно количество положительных и отрицательных действительных корней: количество действительных положительных корней либо равно числу перемен знака в последовательности коэффициентов уравнения, либо на четное число меньше; количество действительных отрицательных корней либо равно числу перемен знака в последовательности коэффициентов уравнения  $P(-x) = 0$ , либо на четное число меньше (равные нулю коэффициенты в обоих случаях не учитываются). Отсюда следует, что правило Декарта не точно определяет количество корней, а приближенно. Существуют и более строгие методы (например, правило Штурма), которые точно определяют количество действительных корней на любом промежутке числовой оси (а не только положительных и отрицательных).

**Пример.**

Имеем уравнение  $P_6(x) = x^6 - 17x^4 + 12x^3 + 7x^2 - x + 1 = 0$ .

Это уравнение имеет шесть корней ( $n = 6$ ). Знаки коэффициентов образуют следующую последовательность: +, -, +, +, -, +. Знак меняется 4 раза, следовательно, положительных корней будет либо 4, либо 2, либо 0. Для отрицательных: последовательность знаков в  $P_6(-x)$  будет: +, -, -, +, +, +, т.е. отрицательных корней будет либо 2, либо 0, так как только два раза меняется знак в последовательности.

## ПРЕДЕЛЬНЫЕ ОЦЕНКИ И ОБЛАСТЬ СУЩЕСТВОВАНИЯ КОРНЕЙ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ

Во многих случаях достаточно знать предельное (максимально возможное) значение корня. Это необходимо в различных приложениях (например, при оценке устойчивости систем управления), а также для более эффективного отделения корней нелинейного уравнения. Эту проблему можно решить, не находя все корни, а воспользовавшись специальными методами, которые кратко рассмотрены ниже.

### МЕТОД ЛАГРАНЖА

Метод сводится к определению верхней границы положительных корней по формуле

$$R = 1 + \sqrt[m]{B/a_0},$$

где  $m$  — номер первого по порядку отрицательного коэффициента в полиноме левой части уравнения  $P_n(x)$ ,  $B$  — наибольшая из абсолютных величин отрицательных коэффициентов  $P_n(x)$ ; при этом предполагается, что  $a_0 > 0$ .

**Пример.**

Определить предельное значение положительных корней уравнения  $-8x^4 + 8x^2 + 32x - 1 = 0$ .

В этом уравнении  $a_0 < 0$ , поэтому сначала умножим обе части уравнения на  $-1$ . Получим  $8x^4 - 8x^2 - 32x + 1 = 0$ . Здесь  $a_0 > 0$ ,  $a_1 = 0$ ,  $a_2 = -8 < 0$ ,  $a_3 = -32$ ,  $a_4 = 1$ . Следовательно,  $m = 2$ ,  $B = 32$ .

Применяя формулу, получим  $R = 1 + \sqrt{32/8} = 3$ .

## МЕТОД НЬЮТОНА

Этот метод не дает конструктивного пути для отыскания искомого значения, т.е. не дает более эффективных алгоритмов помимо простого перебора или проверки наугад взятых точек, и формулируется следующим образом: если при каком-то значении  $x = \beta$   $P_n(x) > 0$  и все производные  $P'(x) > 0$ ,  $P''(x) > 0$ , ...,  $P^{(n)}(x) > 0$ , то значение  $\beta$  является предельным значением положительных корней.

### Пример.

Рассмотрим применение метода для того же уравнения, что и в предыдущем случае.

$$P(x) = 8x^4 - 8x^2 - 32x + 1; \quad P'(x) = 32x^3 - 16x - 32;$$
$$P''(x) = 96x^2 - 16; \quad P'''(x) = 192x; \quad P^{(4)}(x) = 192.$$

Из третьей производной видно, что нужно искать предельное значение при  $x > 0$ . Из второй производной получим:  $96x^2 - 16 > 0$ , т.е.  $x^2 > 16/96$ , или  $x > 4/9,8$ . Далее из неравенства  $P'(x) > 0$  можно попытаться найти значения  $x$ , удовлетворяющие ему и не противоречащие ранее найденным, и т.д. Однако неравенства с полиномами высокого порядка решаются довольно сложно и не всегда. Поэтому чаще используют простой подбор такого значения  $\beta$ , при котором справедливы все неравенства.

## МЕТОД КОЛЬЦА

Метод позволяет находить область существования всех корней алгебраического уравнения, в том числе и комплексных. Действительные корни находятся в интервалах  $(-R, r)$  и  $(r, R)$  (соот-

ветственно отрицательные и положительные). Величины  $r$  и  $R$  вычисляются по формулам

$$R = 1 + \frac{A}{|a_0|}; \quad r = \frac{1}{1 + B/|a_n|},$$

где  $A = \max\{|a_1|, |a_2|, \dots, |a_n|\}$ ;  $B = \max\{|a_0|, |a_1|, \dots, |a_{n-1}|\}$ .

В случае нахождения области существования всех корней, а не только действительных, строится кольцо с радиусами  $r$  и  $R$ , внутри которого находятся и действительные корни. Метод дает диапазон существования корней приближенно с определенным запасом.

### *Пример.*

Определить границы корней уравнения  $5x^3 - 20x + 3 = 0$ . Здесь  $|a_0| = 5$ ,  $A = 20$ ,  $|a_n| = 3$ ,  $B = 20$ .

Используя формулы, получим

$$R = 1 + \frac{20}{5} = 5; \quad r = \frac{1}{1 + 20/3} \approx 0,013.$$

Отсюда следует, что положительные корни находятся в интервале  $(0,013, 5)$ , а отрицательные — в интервале  $(-5, -0,013)$ . Следует отметить, что указание границ корней не означает, что такие корни обязательно есть. В данном случае мы указали области для положительных и отрицательных корней, а если в уравнении окажутся комплексные корни, то действительный будет только один, так как уравнение третьей степени.

## **МЕТОД ПРЕДЕЛЬНЫХ ЗНАЧЕНИЙ**

Существует метод, позволяющий несколько точнее находить область существования корней. Для этого вычисляют верхние границы положительных корней  $R_i$ :

для полинома  $P_n(x) — R_1$ ,

для полинома  $P_n(-x) — R_2$ ,

для полинома  $x^n P_n(x) — R_3$ ,

для полинома  $x^n P_n(-1/x) — R_4$ .

Если действительные корни существуют, то они лежат в интервалах  $(-R_2, -1/R_4)$  и  $(1/R_3, R_1)$ .

## УТОЧНЕНИЕ КОРНЕЙ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ

Алгебраические уравнения можно, конечно, решать любым подходящим методом из ранее приведенных, но конкретная и заранее известная структура левой части уравнения позволяет строить более эффективные алгоритмы. Идея уточнения корней алгебраического уравнения методом понижения порядка базируется на возможности выделения в полиноме  $P_n(x)$  множителя, содержащего корень. Для действительного корня таким множителем является  $x - x_1$ , где  $x_1$  — корень, а для пары комплексных корней  $\alpha \pm j\omega$  — множитель  $x^2 + bx + c$ , где  $b = -2\alpha$ ,  $c = \alpha^2 + \omega^2$ . После такого выделения можно снизить порядок исходного полинома на единицу (или на два) путем деления  $P_{n-1} = P_n / (x - x_1)$  (для действительного корня) или  $P_{n-2} = P_n / (x^2 + bx + c)$  (для комплексной пары корней), далее для нового полинома опять выделить следующий корень  $x_2$  (или новую пару корней) и т.д. В рассматриваемом алгоритме можно выделить три этапа:

- 1) нахождение приближенного значения корня;
- 2) проверка, является ли найденное значение корнем;
- 3) получение нового полинома степени на единицу меньше, чем предыдущая.

### УТОЧНЕНИЕ ДЕЙСТВИТЕЛЬНОГО КОРНЯ

При выделении действительного корня процесс его получения носит итерационный характер и реализуется по формуле

$$x_{i+1} = -x_i \frac{a_n}{a_0 x_i^n + a_1 x_i^{n-1} + \dots + a_{n-1} x_i} = - \frac{a_n}{a_0 x_i^{n-1} + a_1 x_i^{n-2} + \dots + a_{n-1}}$$

Здесь через  $x_i$  и  $x_{i+1}$  обозначены соответственно  $i$ -е и  $(i+1)$ -е приближения выделяемого корня. Если начальное приближение достаточно близко к искомому корню, то итерационный процесс сойдется очень быстро. К сожалению, сходимость метода зависит от

условия сходимости

$$\left| 1 + x_1 \frac{P'(x_1)}{P(0)} \right| < 1,$$

которое практически невозможно оценить заранее ( $x_1$  — искомый корень). Кроме того, метод может не сходиться к решению и вследствие отсутствия действительного корня.

По существу рассматриваемый метод является методом итераций для одного из способов приведения к необходимому виду

$$x = \varphi(x) = x + \frac{x P(x)}{P(0) - P(x)}.$$

Значение приближения  $x_i$  на каждой итерации нужно проверять, не является ли оно корнем. Для этого достаточно найти остаток  $r$  от деления  $P_n(x)$  на  $x - x_i$ . Если остаток равен нулю (или очень близок к нулю), то текущее значение  $x_i$  является корнем, если нет, то итерационный процесс следует повторять. Воспользовавшись теоремой Безу, можно находить остаток, не производя собственно операцию деления. В соответствии с этой теоремой остаток равен значению полинома при  $x = x_i$ , т.е.  $r = P_n(x_i)$ . Такой подход существенно сокращает вычисления.

После итерационного выделения одного корня (что контролируется по остатку  $r$ ) следует понизить порядок полинома, т.е. найти новый полином  $P_{n-1}$ , содержащий остальные корни:

$$P_{n-1}(x) = \frac{P_n(x)}{x - x_1}.$$

Коэффициенты  $b_k$  нового полинома  $P_{n-1}$  находятся по схеме Горнера (не проводя операцию деления), в соответствии с которой

$$b_0 = a_0; \quad b_1 = b_0 x_1 + a_1; \quad b_2 = b_1 x_1 + a_2;$$

$$b_3 = b_2 x_1 + a_3; \quad \dots \quad b_k = b_{k-1} x_1 + a_k; \quad \dots \quad b_{n-1} = b_{n-2} x_1 + a_{n-1},$$

где  $a_k$  — коэффициенты полинома  $P_n(x)$ .

**Пример.**

Найти один из корней уравнения  $x^4 + 3x^3 - 2x^2 + 3x - 1 = 0$ .

Примем в качестве начального значения  $x = 0,75$ . Результаты итерационного вычислительного процесса сведем в табл. 6.

Таблица 6

$i$	$x_i$	$x_{i+1}$
0	0,75000000	0,27705628
1	0,27705628	0,37072257
2	0,37072257	0,36740246
3	0,36740246	0,36768187
4	0,36768187	0,36765881
5	0,36765881	0,36766072
6	0,36766072	0,36766056

Из таблицы видна высокая скорость сходимости метода. Примем за корень  $x_1 = 0,367660$ . Проиллюстрируем и несходящийся итерационный процесс на примере решения уравнения  $x^4 - x^3 - 2x^2 + 3x - 1 = 0$  (один из корней равен 1,73). В качестве начальной точки возьмем точку, близкую к решению  $x_0 = 1,75$ . Результаты вычислений приведены ниже в табл. 7.

Таблица 7

$i$	$x_i$	$x_{i+1}$
0	1,75000000	1,66956522
1	1,66956522	1,96431777
2	1,96431777	1,07441109
3	1,07441109	3,20145152
4	3,20145152	0,15657280

В этом случае имеем расходящийся итерационный процесс. Решения здесь не получить, надо использовать другие методы.

Читатель может самостоятельно проанализировать выполнение условий сходимости по приведенным соотношениям для обоих примеров.

## УТОЧНЕНИЕ КОМПЛЕКСНОЙ ПАРЫ КОРНЕЙ (МЕТОД ХИЧКОКА)

Комплексные корни алгебраических уравнений с действительными коэффициентами всегда сопряженные, поэтому для их нахождения достаточно выделить квадратичный множитель вида  $x^2 + px + q$ , а затем уже найти его комплексные корни. В нахождении такого множителя и заключается поиск комплексных корней многими методами, в том числе и рассматриваемым.

Если разделим  $P_n(x)$  на  $x^2 + px + q$ , то получим тождество

$$P_n(x) = (x^2 + px + q)L_{n-2}(x) + xP(p, q) + Q(p, q),$$

где  $L_{n-2}(x)$  — частное от деления  $P_n(x)$  на квадратичный трехчлен, а  $P(p, q)$  и  $Q(p, q)$  — многочлены от  $p$  и  $q$ . Для того чтобы  $x^2 + px + q$  был делителем  $P_n(x)$ , необходимо и достаточно, чтобы  $P(p, q) = 0$  и  $Q(p, q) = 0$ . Если решить систему уравнений

$$P(p, q) = 0, \quad Q(p, q) = 0,$$

то мы получим необходимые значения  $p$  и  $q$ .

Процесс нахождения  $p$  и  $q$  носит итерационный характер:

$$p^{i+1} = p^i + h^i, \quad \text{где } h^i = \frac{\Delta p^i}{\Delta^i}, \quad q^{i+1} = q^i + t^i, \quad \text{где } t^i = \frac{\Delta q^i}{\Delta^i},$$

а  $\Delta p^i$ ,  $\Delta q^i$ ,  $\Delta^i$  в свою очередь вычисляются через определители так:

$$\Delta p^i = \begin{vmatrix} P'_q(p^i, q^i) & P(p^i, q^i) \\ Q'_q(p^i, q^i) & Q(p^i, q^i) \end{vmatrix},$$

$$\Delta q^i = \begin{vmatrix} P(p^i, q^i) & P'_p(p^i, q^i) \\ Q(p^i, q^i) & Q'_p(p^i, q^i) \end{vmatrix},$$

$$\Delta^i = \begin{vmatrix} P'_p(p^i, q^i) & P'_q(p^i, q^i) \\ Q'_p(p^i, q^i) & Q'_q(p^i, q^i) \end{vmatrix}.$$

Значения производных  $P'_p(p^i, q^i)$  (для компактности использованы стандартные обозначения  $P'_p(p^i, q^i) \equiv dP/dp$ ),  $P'_q(p^i, q^i)$ ,

$Q'_p(p^i, q^i), Q'_q(p^i, q^i)$  находятся по формулам:

$$\begin{aligned} P'_p(p^i, q^i) &= p^i R^i - S^i; & P'_q(p^i, q^i) &= -R^i; \\ Q'_p(p^i, q^i) &= q^i R^i; & Q'_q(p^i, q^i) &= -S^i. \end{aligned}$$

Величины  $P(p^i, q^i)$  и  $Q(p^i, q^i)$  вычисляются из разложения

$$P_n(x) = (x^2 + p^i x + q^i) L_{n-2}(x) + xP(p^i, q^i) + Q(p^i, q^i),$$

а  $R^i$  и  $S^i$  вычисляются из разложения

$$L_{n-2}(x) = (x^2 + p^i x + q^i) G_{n-4}(x) + xR(p^i, q^i) + S(p^i, q^i).$$

Метод требует сравнительно много вычислений и не всегда может сходиться. Корнями выделенного квадратичного трехчлена могут оказаться и действительные корни.

После выделения квадратичного сомножителя находят полином более низкой степени  $L(x)$ , и к нему применяют аналогичный алгоритм для выделения следующей пары комплексных корней.

### Пример.

Найти комплексные корни уравнения

$$x^4 + 4x^3 + 4,8x^2 + 16x - 1 = 0.$$

Приведем только основные промежуточные результаты вычислений (табл. 8).

Таблица 8

$i$	$P^i / q^i$	$R^i / S^i$	$P^i / Q^i$	$P'_p / Q'_p$	$P'_q / Q'_q$	$\Delta^i$	$\Delta p^i / \Delta q^i$	$h^i / r^i$
0	3 -1,75	-2 5,3	7,1 5,21	-11,3 3,5	2 -5,3	52,8 9	48,05 83,75	0,9 1,6
1	3,9 -0,15	3,8 4,71	-1,769 -0,316	-19,53 0,57	3,8 -4,71	89,8 2	-9,53 -7,18	-0,11 -0,08
2	3,79 -0,23	-3,58 4,46	0,001 0,0262	-18,03 0,8234	3,58 4,4641	77,5 5	-0,0889 -0,4715	-0,0011 -0,0061
3	3,789 -0,236	-3,5778 4,4723	-0,0093 0,0002	-18,028 0,8447	3,5778 -4,4724	77,6	-0,00352 0,00246	-0,00005 0,00003
4	3,788 -0,236							

Итак, выделен квадратичный трехчлен  $x^2 + 3,788x - 0,236$ . Его корнями являются  $x_1 \approx -3,8502, x_2 \approx 0,0613$ .

После деления исходного полинома на выделенный трехчлен получим следующий полином:  $L_2(x) = x^2 + 0,21115x + 4,23605$ . Приравняв его к нулю и решив, получим еще одну пару корней  $x_{3,4} \approx -0,1056 \pm 2,055j$ .

## РЕШЕНИЕ СИСТЕМ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

### КОНЦЕПЦИЯ МЕТОДОВ

Решение системы нелинейных уравнений  $f(x) = 0$ , где  $x$  — векторная величина, а  $f(x)$  — векторная функция, значительно сложнее, чем решение одного уравнения. Очень сложно отделить корни, поэтому на этом этапе обычно выбирают начальное приближение ближе к потенциально возможному решению. Для получения всех возможных решений чаще всего на практике перебирают различные начальные условия поиска. В основном используют два метода решения систем: метод Ньютона — Рафсона и метод итераций. Реже используются поисковые методы оптимизации, которыми ищут минимум суммы квадратов невязок  $f(x) = \varepsilon$  всех уравнений системы, подбирая переменные  $x_i$ :

$$R = \sum_i \varepsilon_i^2 \rightarrow \min_{x_i}$$

который будет иметь место только при всех  $\varepsilon_i = 0$ , т.е. соответствующие значения  $x_i$  будут являться решением системы.

### МЕТОД НЬЮТОНА — РАФСОНА

Метод базируется на разложении функций  $f_j(x_1, x_2, \dots, x_n)$  в ряд Тейлора с отбрасыванием всех нелинейных членов разложения (т.е. функции линеаризуют). Затем ставится задача перехода по всем переменным из текущей точки  $x^i$  в следующую  $x^{i+1}$ , которую считают решением, т.е. находят  $h^i = x^{i+1} - x^i$ , поэтому пола-

гают  $f(x^{i+1})=0$ . Получается система линейных уравнений относительно  $h_j^i$  ( $j = 1, 2, \dots, n$ ):

$$\frac{df_k(x^i)}{dx_1} h_1^i + \frac{df_k(x^i)}{dx_2} h_2^i + \dots + \frac{df_k(x^i)}{dx_n} h_n^i = -f_k(x^i), \quad k=1, 2, \dots, n$$

(здесь и далее через  $d$  обозначены частные производные), которую решают. С найденными шагами  $h_j$  переходят в новую точку по всем переменным  $x_j^{i+1} = x_j^i + h_j^i$  ( $j = 1, 2, \dots, n$ ). Однако вследствие линейного приближения новая точка  $x^{j+1}$  не является решением, т.е.  $f(x^{i+1}) \neq 0$ , ее просто считают следующим приближением и повторяют всю процедуру многократно, делая процесс поиска корней итерационным. Сходимость метода довольно высокая, особенно если удачно выбрано начальное приближение.

Окончание поиска решения осуществляется по условию малости шагов по всем переменным одновременно, например при выполнении условия  $|h_j^i| \leq \varepsilon$  для  $j = 1, 2, \dots, n$ , где  $\varepsilon$  — заданная погрешность. Так же как и при решении одного уравнения, можно воспользоваться для окончания поиска условием малости отклонений функций  $f(x_1, \dots, x_n)$  от нуля, например условием  $\max\{f_j(x^i)\} \leq \varepsilon$ , или потребовать выполнения обоих условий сразу.

Недостатком метода является необходимость на каждом шаге вычислять частные производные по всем переменным от всех функций и решать систему линейных уравнений. Существует модификация, не требующая на каждом шаге вычисления производных (используются производные, полученные на первом шаге), но сходимость такого метода существенно ниже.

### **Пример.**

Решить систему уравнений с погрешностью не хуже 0,001.

$$f_1(x_1, x_2) = \sin(2x_1 - x_2) - 1,2x_1 - 0,4 = 0,$$

$$f_2(x_1, x_2) = 0,8x_1^2 + 1,5x_2^2 - 1 = 0.$$

Найдем выражения для производных:

$$\frac{df_1}{dx_1} = 2\cos(2x_1 - x_2) - 1,2, \quad \frac{df_1}{dx_2} = -\cos(2x_1 - x_2), \quad \frac{df_2}{dx_1} = 1,6x_1, \quad \frac{df_2}{dx_2} = 3x_2.$$

Новые точки находим через предыдущие:

$$x_1^{i+1} = x_1^i + h_1^i, \quad x_2^{i+1} = x_2^i + h_2^i.$$

Величины  $h$  будем находить из решения системы уравнений через определители.

$$\frac{df_1(x_1, x_2)}{dx_1} h_1 + \frac{df_1(x_1, x_2)}{dx_2} h_2 = -(\sin(2x_1 - x_2) - 1, 2x_1 - 0, 4),$$

$$\frac{df_2(x_1, x_2)}{dx_1} h_1 + \frac{df_2(x_1, x_2)}{dx_2} h_2 = -(0, 8x_1^2 + 1, 5x_2^2 - 1).$$

В качестве начальной точки возьмем  $x_1^0 = 0,4$ ,  $x_2^0 = -0,75$ , результаты вычислений сведем в табл. 9.

Таблица 9

$i$	$x_1$	$x_2$	$df_1/dx_1$	$df_1/dx_2$	$df_2/dx_1$	$df_2/dx_2$	$h_1$	$h_2$
0	0,4000	-0,7500	-1,158	-0,021	0,640	-2,250	0,1030	0,0170
1	0,5030	-0,7331	-1,536	0,168	0,805	-2,199	-0,0168	-0,0003
2	0,4914	-0,7334	-1,490	0,145	0,786	-2,200	-0,0001	-0,0001
3	0,4913	-0,7335						

Ответ:  $x_1 \approx 0,491$ ,  $x_2 \approx -0,734$ .

### МЕТОД ИТЕРАЦИЙ

В этом методе, как и при решении одного уравнения, предварительно все уравнения приводят к специальному виду  $x = \varphi(x)$ :

$$\begin{aligned} x_1 &= f_1(x_1, x_2, \dots, x_n), \\ x_2 &= f_2(x_1, x_2, \dots, x_n), \\ &\dots \dots \dots \\ x_n &= f_n(x_1, x_2, \dots, x_n). \end{aligned}$$

Сущность метода аналогична соответствующему методу при решении одного уравнения. Берется произвольное начальное зна-

чение  $x^0 (x_1^0, x_2^0, \dots, x_n^0)$  и подставляется в уравнение. Из каждого уравнения системы находятся новые значения  $(x_1^1, x_2^1, \dots, x_n^1)$ , т.е.  $x^1$ , которое затем опять подставляется в уравнение и находится  $x^2$ , затем  $x^3$  и т.д. При выполнении условий

$$\sum_{i=1}^n M_{ki} < 1 \quad \text{или} \quad \sum_{k=1}^n M_{ki} < 1, \quad \text{где} \quad M_{ki} = \max \left| \frac{d\varphi_k}{dx_i} \right|,$$

последовательность  $x^0, x^1, x^2, x^3, \dots$  сходится к решению. Для обеспечения сходимости метода можно использовать следующий способ преобразования исходной системы к виду, удобному для итераций:

$$x_{i+1} = x_i + \sum_j \alpha_{ij} f_j(x) = \varphi_j(x),$$

где  $\alpha_{ij}$  находятся из решения вспомогательной системы уравнений

$$\frac{d\varphi_i(x)}{dx_j} = 0, \quad i, j = 1, 2, 3, \dots, n.$$

Недостатком такого подхода является необходимость большого объема вычислений, так как преобразования следует делать на каждом шаге итераций. Несколько меньшая сходимость может быть получена, если преобразование от  $f(x) = 0$  к  $x = \varphi(x)$  делать через несколько шагов или один раз.

### **Пример.**

Решить систему уравнений

$$\sin(x_1 - 0,6) - x_2 = 1,6,$$

$$3x_1 - \cos x_2 = 0,9$$

с погрешностью не хуже 0,001.

В качестве начальных условий примем  $x_1 = 0,15, x_2 = -2$ . Преобразуем систему к виду, удобному для применения метода, путем выражения переменных из каждого уравнения (этот прием

далеко не всегда применим, но в данном случае возможен):

$$x_1 = \frac{1}{3} \cos x_2 + 0,3 = \varphi_1(x_1, x_2),$$

$$x_2 = \sin(x_1 - 0,6) - 0,6 = \varphi_2(x_1, x_2).$$

Проверим выполнение условий сходимости

$$\frac{d\varphi_1}{dx_1} = 0, \quad \frac{d\varphi_1}{dx_2} = -\frac{1}{3} \sin x_2, \quad \frac{d\varphi_2}{dx_1} = \cos(x_1 - 0,6), \quad \frac{d\varphi_2}{dx_2} = 0.$$

Нетрудно заметить, что условия сходимости будут выполняться всегда:

$$\left| \frac{d\varphi_1}{dx_1} \right| + \left| \frac{d\varphi_2}{dx_1} \right| = |\cos(x_1 - 0,6)| < 1,$$

$$\left| \frac{d\varphi_1}{dx_2} \right| + \left| \frac{d\varphi_2}{dx_2} \right| = \left| -\frac{1}{3} \sin x_2 \right| < 1.$$

Все промежуточные вычисления для компактности записи сведем в табл. 10.

Таблица 10

$i$	0	1	2	3	4	5	6	7
$x_1$	0,15	0,1616	0,1508	0,1539	0,1510	0,1519	0,1510	0,151
$x_2$	-2,00	-2,035	-2,0245	-2,0342	-2,0313	-2,0341	-2,0333	-2,034

Ответ:  $x_1 \approx 0,151, x_2 \approx -2,034$ .

## КОНТРОЛЬНЫЕ ВОПРОСЫ

### ОТДЕЛЕНИЕ КОРНЕЙ

1. Что дает отделение корней?
2. Можно ли аналитически отделить корень функции с разрывами?

3. Можно ли произвольно задавать значения на отрезке по оси  $x$  для отделения корней?
4. Что при отделении корней называют критическими точками?
5. Сколько корней может быть у функции, если у нее существует лишь одна критическая точка?
6. Какие основные проблемы могут встретиться при аналитическом отделении корней?

### **УТОЧНЕНИЕ КОРНЕЙ**

#### ***Метод деления отрезка пополам***

1. В чем заключается геометрический смысл метода половинного деления?
2. Всегда ли позволяет метод половинного деления вычислить отделенный корень уравнения с заданной погрешностью?
3. Как выбираются концы отрезка следующего интервала в методе половинного деления?
4. Какими свойствами должна обладать функция  $f(x)$ , чтобы методом половинного деления можно было гарантированно решить уравнение  $f(x) = 0$ ?
5. Что необходимо для нахождения хотя бы одного действительного корня уравнения  $f(x) = 0$  методом половинного деления?
6. Можно ли найти корень методом половинного деления, если он находится на границе интервала?

#### ***Метод хорд***

1. Какие корни позволяет определить метод хорд?
2. В чем заключается геометрический смысл метода хорд?
3. Всегда ли метод хорд позволяет вычислить отделенный корень с заданной погрешностью?
4. Как выбираются концы отрезка интервала в методе хорд?
5. Какими свойствами должна обладать функция  $f(x)$  для того, чтобы методом хорд можно решить уравнение  $f(x) = 0$ ?
6. Какой конец хорды неподвижен при реализации метода?

### *Метод Ньютона*

1. В чем заключается геометрическая интерпретация метода Ньютона?
2. Исходя из чего выбирается в методе Ньютона первое приближение  $x_0$ ?
3. Как выбираются концы “закрепленного” отрезка интервала в методе Ньютона при  $f \cdot f'' < 0$  на концах интервала?
4. Как выбираются концы “закрепленного” отрезка интервала в методе Ньютона при  $f \cdot f'' > 0$  на концах интервала?
5. Что необходимо для того, чтобы уравнение  $f(x) = 0$  решалось методом Ньютона?
6. В каких случаях применение метода Ньютона не рекомендуется?

### *Метод параболической аппроксимации*

1. В чем заключается геометрический смысл метода параболической аппроксимации?
2. Последовательность каких процессов представляет собой метод параболической аппроксимации?
3. Как выбираются концы отрезка интервала в методе параболической аппроксимации на втором и последующих шагах?
4. В каких ситуациях метод параболической аппроксимации не найдет корень?
5. Можно ли утверждать, что в методе парабол последовательные приближения могут лежать по одну сторону от корня?
6. Может ли метод параболической аппроксимации найти корень, если на начальном участке находится несколько корней?

### *Метод простой итерации*

1. Какой функцией заменяется левая часть уравнения  $f(x) = 0$  в методе итераций?
2. Что называется сходимостью метода итераций?
3. С какой стороны может осуществляться приближение к корню в процессе итераций — слева или справа?

4. Если на заданном отрезке имеется два корня, то что можно сказать о сходимости метода итераций на этом отрезке?
5. Что означает несходимость процесса итераций?
6. Есть ли отличие условий окончания поиска при “монотонном” и при “колебательном” приближении к корню?

### *Алгебраические уравнения*

1. Почему для решения алгебраических уравнений целесообразно придумывать какие-то специальные методы?
2. Всегда ли перед применением метода понижения порядка требуется отделять корень?
3. В каком случае можно точно определить количество корней, используя правило Декарта?
4. Каким образом можно определить число отрицательных корней уравнения  $P(x) = 0$ , если полином является полным?
5. Как найти общее число корней алгебраического уравнения?
6. Почему сумма положительных и отрицательных корней не обязательно равна наивысшей степени полинома в левой части уравнения?

### *Системы нелинейных уравнений*

1. Как проводится отделение корней при решении систем нелинейных уравнений?
2. Почему после одного шага по методу Ньютона — Рафсона мы не попадаем в решение, хотя рассчитывали из условия попадания в решение?
3. От чего зависит скорость сходимости метода Ньютона — Рафсона?
4. Можно ли обеспечить сходимость метода итераций при решении систем нелинейных уравнений?
5. Каким образом можно повысить точность решения системы нелинейных уравнений?
6. Оказывает ли влияние на результат решения выбор начального приближения в методе итераций?



или чаще — в матричном виде:  $ax = B$ , где  $a$  — квадратная матрица размером  $n \times n$ ,  $B$  и  $x$  — векторы размером  $n$  ( $n$  — размерность системы).

## ТОЧНЫЕ МЕТОДЫ

Точные методы обычно рассматриваются в курсах линейной алгебры, поэтому здесь упомянем коротко основные особенности самых распространенных методов для сопоставления с итерационными, которые обычно изучаются в курсах прикладной математики.

**Метод Гаусса** сводится к двум этапам. На первом осуществляется приведение исходной системы уравнений с помощью преобразований к эквивалентной системе с верхней треугольной матрицей (т.е. приведение системы к треугольному виду). Преобразования сводятся к умножению всех членов уравнения на постоянное число, сложению уравнений, выражению отдельных переменных через другие и т.п. Это прямой ход. На втором этапе, т.е. в обратном ходе (снизу вверх), находятся последовательно все переменные системы. В отдельных случаях, в частности при умножении всех членов уравнения на очень большое число (или при делении на очень маленькое), появляются большие вычислительные ошибки, которые обуславливают значительные погрешности результатов решения.

Более практичным является **метод оптимального исключения**, представляющий собой видоизменение метода Гаусса и требующий меньше памяти для решения. Здесь обратный ход соединен с прямым ходом за счет исключения всех уже выраженных переменных из вышестоящих уравнений.

Упомянем также **метод Крамера** (с использованием определителей), который требует очень больших вычислений уже при прямом решении систем из пяти — десяти уравнений, приведения матрицы  $A$  к форме произведения двух треугольных матриц, что позволяет свести решение заданной системы к последовательному решению двух систем с треугольными матрицами, что является задачей более простой. Поэтому вычисления определителей для матриц высокого порядка осуществляются обычно прибли-

женными методами, и метод Крамера перестает быть в полном смысле точным.

Во всех методах этой группы может появляться накапливающаяся вычислительная ошибка (не алгоритмическая!). Для ее контроля (а не для управления ею!) применяют специальные приемы. Например, в методе Гаусса к каждой строке добавляют еще один член, который равен сумме всех коэффициентов строки. С этим членом делают те же операции, что и с коэффициентами уравнения. На каждом шаге проверяют равенство суммы коэффициентов и “контрольного” добавленного члена: разница говорит о появлении накопившейся вычислительной погрешности. Можно оценивать не только абсолютную, но и относительную ошибку в равенстве суммы коэффициентов и “контрольного” добавленного члена. Таким образом, точные методы могут давать результат с погрешностью, которой трудно управлять и которая в ряде случаев может оказаться значительной, например при высоких порядках системы.

В заключение отметим, что системы с плохо обусловленными матрицами коэффициентов нецелесообразно решать указанными методами вследствие возможности появления очень больших ошибок.

## **ПРИБЛИЖЕННЫЕ МЕТОДЫ**

Эти методы дают возможность найти решение системы как предел бесконечного вычислительного процесса, позволяющего по уже найденным приближениям к решению построить следующее, более точное приближение. Важной чертой таких методов является их самоисправляемость и простота реализации. Если в точных методах ошибка в вычислениях, когда она не компенсируется случайно другими ошибками, неизбежно ведет к ошибкам в результате, то в случае сходящегося итерационного процесса ошибка в каком-то приближении исправляется в последующих вычислениях, и такое исправление требует только нескольких лишних шагов единообразных вычислений.

Условия и скорость сходимости каждого итерационного процесса существенно зависят от свойств уравнений, т.е. от свойств матрицы системы и от выбора начальных приближений.

**Метод простой итерации.** В этом методе исходную систему уравнений  $Ax = B$  предварительно приводят к виду  $x = Cx + D$ , где  $\det C \neq 0$ . Зная какое-либо приближение  $x^0 = (x_1^0, x_2^0, \dots, x_n^0)$ , можно получить  $x^1$  (здесь верхним индексом обозначен номер итерации) из соотношения  $x^1 = Cx^0 + D$ , затем аналогично  $x^2 = Cx^1 + D$ ,  $x^3 = Cx^2 + D$  и т.д. В общем случае алгоритм записывается следующим образом:

$$x^i = Cx^{i-1} + D.$$

Последовательность  $x^0, x^1, x^2, x^3, \dots, x^i, \dots$  будет сходиться к решению  $x^*$ , если для матрицы  $C$  выполняется одно из неравенств:

$$\sum_j |c_{ij}| < 1 \quad (i = 1, 2, \dots, n),$$

$$\sum_i |c_{ij}| < 1 \quad (j = 1, 2, \dots, n).$$

Очень часто в качестве начального приближения берут вектор  $D$ , т.е.  $x^0 = D$ . Тогда нетрудно оценить погрешность после  $i$ -го приближения

$$\|x^* - x^{i-1}\| \leq \frac{\|C\|^i \|D\|}{1 - \|C\|},$$

где  $\| \cdot \|$  — любая из некоторого набора стандартных норм матрицы, например,

$$\|C\|_1 = \max_j \sum_{i=1}^n |c_{ij}| < 1 \quad \text{или} \quad \|C\|_2 = \max_i \sum_{j=1}^n |c_{ij}| < 1.$$

Последнее соотношение позволяет организовать решение системы линейных уравнений с заранее заданной погрешностью  $\varepsilon$ , используя условие

$$\varepsilon \leq \frac{\|C\|^i \|D\|}{1 - \|C\|},$$

и даже определять необходимое количество итераций  $i$  для достижения заданной погрешности.

**Пример.**

Рассмотрим систему уравнений:

$$4,5x_1 - 1,8x_2 + 3,6x_3 = -1,7; \quad (1)$$

$$3,1x_1 + 2,3x_2 - 1,2x_3 = 3,6; \quad (2)$$

$$1,8x_1 + 2,5x_2 + 4,6x_3 = 2,2. \quad (3)$$

Для выполнения условий сходимости приведем систему к виду, в котором элементы главной диагонали превосходили бы остальные элементы строк. Преобразования носят неформальный характер, требуют определенных навыков. В данном случае новые уравнения с нужными свойствами получим следующим путем:

первое уравнение — путем сложения (1) + (2),

второе уравнение — путем сложений и умножений  $2 \cdot (3) + (2) - (1)$ ,

третье уравнение — путем вычитания (3) - (2).

При этом получим следующую эквивалентную систему:

$$\begin{aligned} 7,6x_1 + 0,5x_2 + 2,4x_3 &= 1,9; \\ 2,2x_1 + 9,1x_2 + 4,4x_3 &= 9,7; \\ -1,3x_1 + 0,2x_2 + 5,8x_3 &= -1,4. \end{aligned}$$

Представим диагональные члены в следующем виде:

$$\begin{aligned} 7,6x_1 &= 10x_1 - 3,4x_1; \\ 9,1x_2 &= 10x_2 - 0,9x_2; \\ 5,8x_3 &= 10x_3 - 4,2x_3. \end{aligned}$$

Система преобразуется к виду

$$\begin{aligned} 10x_1 &= 2,4x_1 - 0,5x_2 - 2,4x_3 + 1,9; \\ 10x_2 &= -2,2x_1 + 0,9x_2 - 4,4x_3 + 9,7; \\ 10x_3 &= 1,3x_1 - 0,2x_2 + 4,2x_3 - 1,4. \end{aligned}$$

Разделив все на 10, получим:

$$\begin{aligned}x_1 &= 0,24x_1 - 0,05x_2 - 0,24x_3 + 0,19; \\x_2 &= -0,22x_1 + 0,09x_2 - 0,44x_3 + 0,97; \\x_3 &= 0,13x_1 - 0,02x_2 + 0,42x_3 - 0,14; \\ \sum_j |c_{1j}| &= 0,24 + 0,05 + 0,24 = 0,53; \\ \sum_j |c_{2j}| &= 0,22 + 0,09 + 0,42 = 0,77; \\ \sum_j |c_{3j}| &= 0,13 + 0,02 + 0,42 = 0,57.\end{aligned}$$

$\|C\| = \max \{0,53, 0,77, 0,57\} = 0,77 < 1$ , следовательно, итерационный процесс будет сходиться.

Рассчитаем число итераций  $k$ , например, для получения решения с погрешностью не хуже 0,001. Его можно найти из выражения  $0,77^k \cdot 0,97 / (1 - 0,77) \leq 0,001$ . Здесь 0,77 — норма матрицы  $C$ , 0,97 — норма вектора правых частей  $D$ .

Минимальное число итераций будет при знаке равенства в соотношении, полученное значение  $k$  необходимо округлить в большую сторону. Опуская очевидные преобразования, получим

$$0,77^k = 0,001 \frac{1 - 0,77}{0,97} = 0,0002371,$$

откуда  $k = \ln(0,0002371) / \ln(0,77) \approx 5,5$ .

Следовательно,  $k = 6$ ; за 6 итераций можно получить решение с погрешностью не хуже 0,001.

Алгоритм решения очевиден, приведем несколько первых итераций (табл. 11).

Таблица 11

№	$x_1$	$x_2$	$x_3$
0	0,22073	1,07710	-0,19356
1	0,23555	1,10352	-0,21412
2	0,24274	1,11171	-0,22138
3	0,24580	1,11406	-0,22366
...	...	...	...

Решением является:  $x_1 = 0,247$ ,  $x_2 = 1,1145$ ,  $x_3 = -0,2243$ .



### *Пример.*

Рассмотрим предыдущий пример. Приведем только таблицу значений на нескольких первых итерациях, так как все остальное будет то же самое (табл.12).

Т а б л и ц а 1 2

№	$x_1$	$x_2$	$x_3$
1	0,2207	1,0703	-0,1915
2	0,2354	1,0988	-0,2118
3	0,2424	1,1088	-0,2196
4	0,2454	1,1124	-0,2226
...	...	...	...

Как нетрудно заметить, в данном случае последовательность медленнее сходится к решению.

## **КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. В чем основное отличие точных и приближенных методов решения систем линейных уравнений?
2. Каким методом лучше всего решать систему уравнений невысокого порядка, например третьего?
3. В каких случаях предпочтительны итерационные методы решения систем линейных уравнений?
4. От чего зависит скорость сходимости метода итераций?
5. Можно ли получить решение системы высокой размерности с погрешностью не хуже заданной?
6. Каким образом в методе Гаусса можно контролировать накопление вычислительных ошибок?
7. К точным или приближенным методам относится метод Крамера?
8. При каком условии будет сходиться метод итераций?
9. Можно ли заранее оценить число итераций для получения решения с заданной погрешностью?
10. Как влияет вычислительная ошибка на точность решения системы уравнений методом итераций?

## ОСНОВЫ РЕШЕНИЯ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

Дифференциальные уравнения очень часто встречаются при построении моделей динамики объектов исследования. Они описывают, как правило, изменение параметров объекта во времени (хотя могут быть и другие случаи). Результатом решения дифференциальных уравнений являются функции, а не числа, как при решении конечных уравнений, вследствие чего методы решения их более трудоемки. Особенно это касается дифференциальных уравнений в частных производных. В данном пособии такие методы не рассматриваются. Владение методами решения дифференциальных уравнений обязательно при моделировании и очень важно для получения правильного результата.

### КОНЦЕПЦИЯ РЕШЕНИЯ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

При использовании численных методов решение дифференциальных уравнений  $dy/dx = f(x, y)$  или  $y' = f(x, y)$  представляется в табличном виде, т.е. получается совокупность значений  $y_i$  и  $x_i$ . Решение носит шаговый характер, т.е. по одной или по нескольким начальным точкам  $(x, y)$  за один шаг находят следующую точку, затем следующую и т.д. Разница между двумя соседними значениями аргумента  $h = x_{i+1} - x_i$  называется *шагом*.

Наибольшее распространение имеют задачи Коши, в которых заданы начальные условия: при  $x = x_0$   $y(x_0) = y_0$ . Имея их, легко начинать процесс решения, т.е. найти  $y_1$  при  $x_1$ ,  $y_2$  — при  $x_2$  и т.д. Задачи другого типа — краевые задачи (например, с конечными условиями или с условиями в промежуточной точке) — решаются специальными приемами, в том числе нередко сведением к другим эквивалентным задачам с начальными условиями.

Выделяют два класса методов решения: *одношаговые* и *многшаговые*. Первый класс методов требует для нахождения следующего значения функции только одной текущей точки, т.е.  $y_{i+1} = F[f(x_i, y_i)]$ , а второй — нескольких, например,  $y_{i+1} = F(y_{i-3}, y_{i-2}, y_{i-1}, y_i)$ . Поэтому методы второго класса не облада-

ют свойством “самостартования”, т.е. ими нельзя начать решение задачи Коши, это всегда делается одношаговыми методами. К недостаткам многошаговых методов относится также и невозможность изменения в процессе решения величины шага (так как они используют предыдущие точки с ранее применяемым шагом, а учет меняющегося шага очень сложен и громоздок), что бывает необходимо для повышения эффективности метода. Заметим, что величина шага существенно влияет на точность и скорость решения, поэтому изменение ее в процессе решения — увеличение при медленно изменяющемся решении и уменьшение при быстро изменяющемся — очень важно для эффективности решения. К достоинствам многошаговых методов относят в основном меньший объем памяти компьютера, требующейся для реализации, и возможность теоретической оценки погрешности решения. Представителем класса многошаговых методов являются методы прогноза и коррекции. К классу одношаговых методов относятся методы Эйлера, Рунге — Кутты и др.

Основная идея получения простейших вычислительных алгоритмов в одношаговых методах сводится к разложению искомого решения  $y(x)$  в ряд Тейлора в окрестности текущей точки и усечения его. Количество оставленных членов ряда определяет порядок и, следовательно, точность метода. По полученному разложению, зная значение  $y$  в точке разложения  $y_i$  и производную  $f(x_i, y_i)$ , находят значение функции  $y$  через шаг  $h$ :  $y_{i+1} = y_i + \Delta y_i$ . Если в разложении удерживается большее число членов, то необходимо рассчитывать  $f(x_i, y_i)$  в нескольких точках (таким способом избегают необходимости прямого вычисления высших производных, присутствующих в разложении в ряд Тейлора). Расчетные алгоритмы многошаговых методов базируются на построении интерполяционных или аппроксимирующих функций, от которых берется интеграл,

Численными методами решаются не только отдельные уравнения, но и системы уравнений (чаще всего первого порядка), причем большинство методов решения одного уравнения легко распространяются на решение систем. Дифференциальные уравнения высших порядков вида

$$y^{(n)} = f(x, y, y', y'', \dots, y^{(n-1)})$$

решаются в основном сведением к системе уравнений первого порядка путем замены переменных:  $y_1 = y'$ ,  $y_2 = y''$ ,  $y_3 = y'''$  и т.д. При этом дифференциальное уравнение  $n$ -го порядка заменяется системой из  $n$  уравнений:

$$\begin{aligned} y' &= y_1, \\ y_1' &= y_2, \\ y_2' &= y_3, \\ &\dots\dots\dots \\ y_{n-1}' &= f(x, y, y_1, y_2, \dots, y_{n-1}). \end{aligned}$$

## ОСНОВНЫЕ МЕТОДЫ

### МЕТОД ЭЙЛЕРА И МОДИФИЦИРОВАННЫЙ МЕТОД ЭЙЛЕРА

Метод Эйлера является одним из самых простых методов решения дифференциальных уравнений первого порядка  $y' = f(x, y)$ . Он в основном используется как учебный, в практических расчетах он дает значительную погрешность. Вычислительный алгоритм представляется следующим образом:

$$y_{i+1} = y(x_i + h) = y_i + \Delta y_i = y_i + hf(x_i, y_i),$$

где  $h$  — шаг по  $x$  (в общем случае может быть непостоянным). Запускается метод из начальных условий  $y(x_0) = y_0$ .

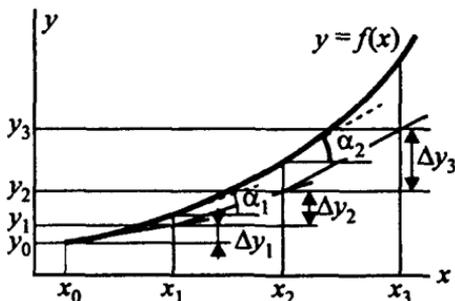


Рис. 12. Иллюстрация метода Эйлера

На рис. 12 приведена графическая интерпретация метода Эйлера. Величины  $\alpha_1$  и  $\alpha_2$  определяются из условий:  $\operatorname{tg} \alpha_1 = f(x_0, y_0)$  и  $\operatorname{tg} \alpha_2 = f(x_1, y_1)$ ,  $\Delta y_1 = \Delta x f(x_0, y_0)$ ,  $\Delta y_2 = \Delta x f(x_1, y_1)$ ; расчетные значения функции — по соответствующим соотношениям.

### Пример 1.

Решить дифференциальное уравнение вида  $y' = 2x^2 + 2y$  при начальных условиях  $x_0 = 0, y(x_0) = 1$  с шагом  $h = 0,1$  на интервале  $[0, 1]$ . Это уравнение имеет аналитическое решение  $y = 1,5e^{2x} - x^2 - x - 0,5$ . Для контроля решения численным методом приведем наряду с численным и точное решение.

Первый шаг:  $y_1 = y_0 + hf(x_0, y_0) = 1 + 0,1(2 \cdot 0 + 2 \cdot 1) = 1,2$ .

Второй шаг:  $y_2 = y_1 + hf(x_1, y_1) = 1,2 + 0,1(2 \cdot 0,1^2 + 2 \cdot 1,2) = 1,442$ .

Процесс вычислений по приведенной формуле не представляет трудностей, поэтому приведем остальные результаты (через точку) в табл. 13.

Таблица 13

x	0	0,2	0,4	0,6	0,8	1,0
Точное решение	1	1,4977	2,2783	3,5202	5,4895	8,5836
Приближенное решение	1	1,4420	2,1041	3,1183	4,6747	7,0472

Для повышения точности на практике используют модифицированный метод Эйлера второго порядка. Он имеет следующий вычислительный алгоритм:

$$y_{i+1} = y_i + 0,5h[f(x_i, y_i) + f(x_{i+1}, y_{i+1})].$$

Здесь в формуле используется значение  $f(x_{i+1}, y_{i+1})$  с еще пока неизвестным значением  $y_{i+1}$ . Это значение может быть найдено предварительно, например, по методу Эйлера, а затем использовано в алгоритме. Если же выражение  $f(x, y)$  несложное, то можно выразить  $y_{i+1}$  из уравнения в явном виде и найти его или решить его относительно  $y_{i+1}$  численными методами.

Точность вычислений обычно контролируют двойным подсчетом: сначала вычисляют решение уравнения на каком-то текущем шаге  $h$ , т.е. находясь в точке  $x_i$  и вычисляя значение  $y(x_i + h) = y_{i+1}^1$ , затем в эту же точку  $x_{i+1}$  приходят за два шага по  $h/2$ , получают  $y_{i+1}^2$ , сравнивают их: если для обоих вариантов различие

$|y_{i+1}^1 - y_{i+1}^2|$  в пределах желаемой погрешности, то решение принимают, а если нет, то опять делят шаг на два и т.д., до тех пор, пока не получится приемлемый результат. Однако следует помнить, что при очень маленьком шаге, получающемся в результате его последовательного деления, может значительной оказаться накапливающаяся вычислительная ошибка.

### Пример 2.

Условие задачи сформулировано в примере 1. Первый шаг (по методу Эйлера):

$$y_1 = y_0 + hf(x_0, y_0) = 1 + 0,1(2 \cdot 0 + 2 \cdot 1) = 1,2.$$

Первый шаг по модифицированному методу:

$$\begin{aligned} y_1 &= y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1)] = \\ &= 1 + \frac{0,1}{2}[(2 \cdot 0 + 2 \cdot 1) + (2 \cdot 0,1^2 + 2 \cdot 1,2)] = 1,221. \end{aligned}$$

Второй шаг (по методу Эйлера):

$$y_2 = y_1 + hf(x_1, y_1) = 1,221 + 0,1(2 \cdot 0,1^2 + 2 \cdot 1,221) = 1,473.$$

Второй шаг по модифицированному методу:

$$\begin{aligned} y_2 &= y_1 + \frac{h}{2}[f(x_1, y_1) + f(x_2, y_2)] = \\ &= 1,221 + \frac{0,1}{2}[(2 \cdot 0,1^2 + 2 \cdot 1,221) + (2 \cdot 0,2^2 + 2 \cdot 1,221)] = 1,4923. \end{aligned}$$

Результаты дальнейших шагов (через точку) представим в табл. 14.

Таблица 14

$x$	0	0,2	0,4	0,6	0,8	1,0
Точное решение	1	1,4977	2,2783	3,5202	5,4805	8,5836
Приближенное решение	1	1,4923	2,2466	3,4176	5,2288	8,0032

## МЕТОД РУНГЕ — КУТТА

Существует целая группа методов Рунге — Кутта (в последние годы начинает в литературе появляться “русский” вариант произношения фамилий авторов метода, в соответствии с которым название звучит как метод Рунге — Кутты), среди которых наибольшее распространение получил метод четвертого порядка. Следовательно, он более точен, чем метод Эйлера, который является методом первого порядка. Для расчета одного значения функции необходимо четыре раза вычислять правую часть дифференциального уравнения, а не два, как в модифицированном методе Эйлера второго порядка. Вычислительный алгоритм записывается следующим образом:

$$y_{i+1} = y_i + \frac{k_1 + 2k_2 + 2k_3 + k_4}{6},$$

$$\begin{aligned} \text{где } k_1 &= hf(x_i, y_i); & k_2 &= hf\left(x_i + \frac{h}{2}, y_i + \frac{k_1}{2}\right); \\ k_3 &= hf\left(x_i + \frac{h}{2}, y_i + \frac{k_2}{2}\right); & k_4 &= hf(x_i + h, y_i + k_3). \end{aligned}$$

Здесь также для контроля точности можно применять прием двойного просчета.

### Пример 3.

Условие задачи сформулировано в примере 1. На первом шаге по приведенным для  $k_1, k_2, k_3, k_4$  формулам получим:

$$k_1 = 0,1(2 \cdot 0 + 2 \cdot 1) = 0,2; \quad k_2 = 0,1(2 \cdot 0,05^2 + 2(1 + 0,2/2)) = 0,221;$$

$$k_3 = 0,1(2 \cdot 0,05^2 + 2(1 + 0,221/2)) = 0,223;$$

$$k_4 = 0,1(2 \cdot 0,1^2 + 2(1 + 0,223)) = 0,247;$$

$$y_1 = 1 + \frac{0,2 + 2 \cdot 0,221 + 2 \cdot 0,223 + 0,247}{6} = 1,2221.$$

На втором шаге получим:

$$k_1 = 0,246; \quad k_2 = 0,274; \quad k_3 = 0,276; \quad k_4 = 0,308,$$

соответственно будет

$$y_2 = 1,2221 + \frac{0,246 + 2 \cdot 0,274 + 2 \cdot 0,276 + 0,308}{6} = 1,4977.$$

Дальнейшие результаты (для компактности через точку) представлены в табл. 15.

Таблица 15

$x$	0	0,2	0,4	0,6	0,8	1,0
Точное решение	1	1,4977	2,2783	3,5202	5,4805	8,5836
Приближенное решение	1	1,4977	2,2783	3,5201	5,4894	8,5834

### МЕТОД МИЛНА

Метод Милна относится к многошаговым методам и представляет один из методов прогноза и коррекции. Решение в следующей точке находится в два этапа. На первом этапе осуществляется по специальной формуле прогноз значения функции, а затем на втором этапе — коррекция полученного значения. Если полученное значение  $y$  после коррекции существенно отличается от спрогнозированного, то проводят еще один этап коррекции. Если опять имеет место существенное отличие от предыдущего значения (т.е. от предыдущей коррекции), то проводят еще одну коррекцию и т.д. Однако очень часто ограничиваются одним этапом коррекции.

Метод Милна имеет следующие вычислительные формулы:

- этап прогноза:

$$y_{i+1}^n = y_{i-3} + 4 \frac{h}{3} (2f_{i-2} - f_{i-1} + 2f_i),$$

где для компактности записи использовано следующее обозначение  $f_i = f(x_i, y_i)$ ;

- этап коррекции:

$$y_{i+1} = y_{i-1} + \frac{h}{3} (f_{i+1} + 4f_i + y_{i+1}^n).$$

Абсолютная погрешность определяется по формуле  $\varepsilon \approx |y_{i+1} - y_{i+1}^n|/29$ .

Метод требует несколько меньшего количества вычислений (например, достаточно только два раза вычислить  $f(x, y)$ , остальные запомнены с предыдущих этапов), но требует дополнительного “расхода” памяти. Кроме этого, как уже указывалось выше, невозможно “запустить” метод: для этого необходимо предварительно получить одношаговыми методами первые три точки.

#### **Пример 4.**

Условие задачи сформулировано в примере 1. Первые три точки получаем методом Рунге — Кутта, поэтому они полностью совпадают с результатами предыдущего метода. С вычислительной точки зрения приведенные алгоритмы не представляют трудностей, поэтому в качестве результатов примера приведем в табл. 16 только итоговые значения как для прогноза, так и для коррекции.

Таблица 16

$x$	0	0,2	0,4	0,6	0,8	1,0
Точное решение	1	1,4977	2,2783	3,5202	5,4805	8,5836
Прогноз	1	1,4977	2,2783	3,5201	5,4895	8,5835
Коррекция		—	2,2781	3,5298	5,4890	8,5828

Из сравнения результатов вычислений различными методами видна разница в рассмотренных методах.

## **КОНТРОЛЬНЫЕ ВОПРОСЫ**

### **Метод Эйлера**

1. Что является решением дифференциального уравнения?
2. Необходим ли поиск начальных условий в методе Эйлера?
3. К какой группе относится модифицированный метод Эйлера?
4. Почему точность метода Эйлера пропорциональна  $h$ , а модифицированного —  $h^2$ ?

5. Метод Эйлера относится к одношаговым методам. В чем основное отличие одно- и многошаговых методов?
6. Можно ли методом Эйлера решать системы дифференциальных уравнений?
7. Можно ли использовать метод Эйлера для решения задач, не относящихся к задачам Коши?
8. Обязательно ли необходимо задание начальных условий при решении дифференциального уравнения методом Эйлера?
9. В чем заключается отличие явных и неявных вычислительных схем в модифицированном методе Эйлера?
10. Можно ли оценить погрешность решения дифференциального уравнения, не зная точного решения?

### *Метод Рунге — Кутта*

1. Сколько раз необходимо на каждом шаге вычислять правую часть уравнения при использовании метода четвертого порядка?
2. Как можно оценить погрешность решения дифференциального уравнения при использовании метода Рунге — Кутта?
3. Можно ли задавать погрешность решения при автоматическом подборе шага в относительных величинах?
4. Сколько предыдущих значений функции нужно иметь, чтобы сосчитать одно следующее значение?
5. К какой группе методов (аналитические или численные) относится имеющий аналитическое выражение от искомого значения функции метод Рунге — Кутта?
6. Как записывается рекуррентная формула метода четвертого порядка?
7. Что можно отнести к недостаткам метода, например, самого распространенного четвертого порядка?
8. Как зависит погрешность метода от величины шага решения?
9. Возможно ли применение переменного шага в методе Рунге — Кутта?
10. Каким образом можно организовать автоматический подбор шага решения уравнения?

### *Многошаговые методы*

1. Каковы достоинства многошаговых методов?
2. За сколько этапов реализуется метод Милна?
3. Что делается на этапе прогноза?
4. Что делается на этапе коррекции?
5. Являются ли многошаговые методы итерационными?
6. Почему на каждом шаге многошаговые методы могут использовать меньшее количество раз вычисления правых частей уравнения?
7. Можно ли оценить погрешность метода Милна?
8. Почему для запуска многошаговых методов используют одношаговые методы?
9. Можно ли методами прогноза и коррекции решать системы дифференциальных уравнений?
10. Какой важный для практического применения метода показатель определяется порядком метода?

---

## МЕТОДЫ ОПТИМИЗАЦИИ

---

Методы оптимизации — поиска экстремума функции (в практических задачах — критериев оптимальности) при наличии ограничений или без ограничений очень широко используются на практике. Это прежде всего оптимальное проектирование (выбор наилучших номинальных технологических режимов, элементов конструкций, структуры технологических цепочек, условий экономической деятельности, повышение доходности и т.д.), оптимальное управление, построение нелинейных математических моделей объектов управления (минимизации невязок различной структуры модели и реального объекта) и многие другие аспекты решения экономических и социальных проблем (например, управление запасами, трудовыми ресурсами, транспортными потоками и т.д. и т.п.).

Существует достаточно большое количество численных методов оптимизации. В данном пособии рассмотрены основные из них, классифицирующиеся следующим образом:

1. По размерности решаемой задачи: одномерные и многомерные.

2. По способу формирования шага многомерные методы делятся на следующие виды:

2.1. Градиентные.

- по способу вычисления градиента: с парной пробой и с центральной пробой;
- по алгоритму коррекции шага;
- по алгоритму вычисления новой точки: одношаговые и многошаговые.

2.2. Безградиентные: с поочередным изменением переменных и с одновременным изменением переменных.

2.3. Случайного поиска: с чисто случайной стратегией и со смешанной стратегией.

3. По наличию активных ограничений.

3.1. Без ограничений (безусловные).

### 3.2. С ограничениями (условные):

- с ограничениями типа равенств;
- с ограничениями типа неравенств;
- смешанные.

В пособии представлены пять тем:

1. Одномерная оптимизация.
2. Многомерная безусловная градиентная оптимизация.
3. Многомерная безусловная безградиентная оптимизация.
4. Многомерная безусловная случайная оптимизация.
5. Многомерная условная оптимизация.

Методы одномерной оптимизации являются базой для некоторых “многомерных” методов. В многомерной градиентной оптимизации строится улучшающая последовательность в зависимости от скорости изменения критерия по различным направлениям. При этом под улучшающей последовательностью понимается такая последовательность  $x_0, x_1, \dots, x_i, \dots$ , в каждой точке которой значение критерия оптимальности лучше, чем в предыдущей. В безградиентных методах величина и направление шага к оптимуму при построении улучшающей последовательности формируется однозначно по определенным детерминированным функциям в зависимости от свойств критерия оптимальности в окрестности текущей точки без использования производных (т.е. градиента). Случайные методы используются в задачах высокой размерности. Многомерная условная оптимизация учитывает активные ограничения, выраженные в виде равенств и неравенств. В каждом из рассмотренных направлений имеется большое число методов, обладающих своими достоинствами и недостатками, которые зависят прежде всего от свойств тех функций, экстремум которых ищется. Одним из сравнительных показателей качества метода является количество значений функции, которое нужно вычислить для решения задачи с заданной погрешностью. Чем это число меньше, тем при прочих равных условиях эффективнее метод.

Эти темы охватывают широкий спектр методов и являются достаточным минимумом, необходимым для дальнейшего успешного решения различных задач оптимизации, возникающих при математическом моделировании реальных технологических и экономических объектов и управлении ими.

В теоретических и математических задачах принято рассматривать задачи оптимизации как задачи поиска минимума функ-

ции. Даже методы имеют общее название — методы спуска. Однако при решении реальных практических задач очень часто встречаются задачи и на максимум (например, максимизация дохода, объема выпуска продукции и т.д.). Конечно, легко перейти от одного вида экстремума к другому путем смены знака у критерия оптимальности, но это делают в прикладных нематематических задачах не всегда, чтобы не терять содержательную нить задачи. Поэтому в данной книге не делается упор на поиск именно минимума, тем более что практически все методы могут искать и минимум, и максимум при незначительных изменениях в алгоритмах. Это поможет не потерять контакт со многими нематематическими изданиями, полезными в профессиональной деятельности.

## **ОДНОМЕРНАЯ ОПТИМИЗАЦИЯ**

---

### **КОНЦЕПЦИЯ МЕТОДОВ**

В данном разделе рассматриваются методы решения одномерных задач оптимизации вида

$$R(x) \rightarrow \max / a \leq x \leq b,$$

где  $x$  — скаляр,  $a$  и  $b$  — соответственно минимальное и максимальное возможные значения переменной  $x$ .

В основном рассматриваются алгоритмы, связанные с построением улучшающей последовательности. Решением задачи называется  $x^*$ , при котором  $R(x^*) \geq R(x)$  для любого значения  $a \leq x \leq b$ . При практическом решении задач не будем различать два значения  $x_i$  и  $x_{i+1}$ , если  $|x_i - x_{i+1}| \leq \varepsilon$ , где  $\varepsilon$  — задаваемая погрешность решения.

### **ОСНОВНЫЕ МЕТОДЫ**

#### **МЕТОД СКАНИРОВАНИЯ**

Метод заключается в последовательном переборе всех значений  $a \leq x \leq b$  с шагом  $\varepsilon$  (погрешность решения) с вычислением критерия оптимальности  $R$  в каждой точке. Путем выбора наибольшего из всех вычисленных значений  $R$  и находится решение задачи  $x^*$ .

Достоинство метода в том, что можно найти глобальный максимум критерия, если  $R(x)$  — многоэкстремальная функция. К недостаткам данного метода относится значительное число повторных вычислений  $R(x)$ , что в случае сложной функции  $R(x)$  требует существенных затрат времени.

На практике можно реализовать одну из основных модификаций метода — последовательное уточнение решения, или сканирование с переменным шагом (рис. 13).

На первом этапе сканирования осуществляют с крупным шагом, затем отрезок, внутри которого получено наибольшее значение  $R(x)$ , разбивается на более мелкие отрезки, ищется новый отрезок, внутри которого находится уточненное значение максимума. Он (новый отрезок) опять делится на более мелкие и т.д., до тех пор, пока величина

отрезка, содержащего максимальное значение  $R(x)$ , не будет меньше заданной погрешности. Главный недостаток этого варианта метода — возможность пропуска “острого” глобального максимума  $R(x)$ .

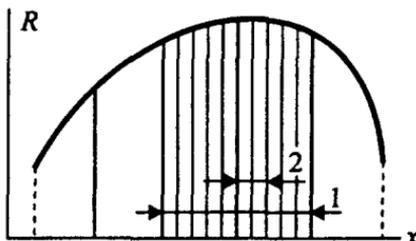


Рис. 13. Иллюстрация модифицированного метода сканирования: 1 — интервал, включающий в себя искомый максимум функции после первого этапа сканирования (исходный участок разбит на 5 участков); 2 — то же, после второго этапа

### Пример.

Дана функция  $R(x) = D \sin(Ax^B + C)$ , где коэффициенты имеют следующие значения:  $A = 1,0$ ,  $B = 1,0$ ,  $C = 1,0$ ,  $D = 1,0$ . Найти максимум на интервале:  $[-1, 2]$ . Ошибка задается по  $x$ :  $\varepsilon = 0,05$ .

Результаты расчетов. Разобьем весь интервал на четыре подынтервала (крупный шаг), координаты  $x$  будут следующими:  $x_1 = -0,25$ ,  $x_2 = 0,5$ ,  $x_3 = 1,25$ .

Соответственно значения критерия равны:  $R_1 = 0,68163$ ,  $R_2 = 0,9974$ ,  $R_3 = 0,77807$ . Следовательно, в качестве нового отрезка выбираем отрезок  $[-0,25, 1,25]$ , так как внутри него находится

максимальное значение:  $x_0 = 0,5$ ,  $R_0 = 0,99749499$ , 0 — номер итерации (после первого этапа). Разбиваем его снова на четыре части, имеем значения:  $x_1 = 0,125$ ,  $x_2 = 0,5$ ,  $x_3 = 0,875$ . Вычислив  $R(x)$  в этих точках, получим, что новый интервал, внутри которого лежит экстремум, равен  $[0,125, 0,875]$ .

Далее в табл. 17 приводятся только координаты середин отрезков, при которых критерий имеет наибольшее значение, номер итерации и значение критерия.

Таблица 17

№	$x$	$R$	№	$x$	$R$
1	0,50000000	0,99749499	4	0,57031250	0,99999988
2	0,50000000	0,99749499	5	0,59375000	0,99973658
3	0,59375000	0,99973658			

Всего проведено  $6 \cdot 3 = 18$  вычислений критерия оптимальности.

### МЕТОД ДЕЛЕНИЯ ПОПОЛАМ

Метод основан на делении текущего отрезка  $[a, b]$ , где содержится искомый экстремум, на две равные части с последующим выбором одной из половин, в которой локализуется максимум в качестве следующего текущего отрезка. Экстремум локализуется путем сравнения двух значений критерия оптимальности в точках, отстоящих от середины отрезка на  $\varepsilon / 2$ , где  $\varepsilon$  — погрешность решения задачи оптимизации.

Если  $R(x + \varepsilon/2) > R(x - \varepsilon/2)$ , то максимум располагается на правой половине текущего отрезка  $[a, b]$ , в противном случае — на левой.

Процесс поиска завершается при достижении отрезком  $[a, b]$  величины заданной погрешности  $\varepsilon$ .

К недостаткам метода относится его работоспособность только для одноэкстремальных функций  $R(x)$  (т.е. таких, которые содержат один экстремум того типа, который мы ищем в задаче),

так как в других случаях при сравнении двух критериев в соседних точках невозможно правильно выбрать следующий интервал, где находится максимум.

На рис. 14 приведены три этапа метода половинного деления. Сплошными вертикальными линиями отмечены середины отрезков, а пунктирными — вычисляемые значения критерия оптимальности слева и справа на  $\epsilon/2$  от середин.

Существует и другой вариант алгоритма, заключающийся в следующем. После нахождения середины отрезка (например, точка  $c_1$ ) в одной из половинок (допустим, в левой) находят среднюю точку (точка  $c_2$ ) и, сравнивая значения функции в этих точках, определяют, в какой из половинок находится экстремум. Если  $R(c_1) < R(c_2)$ , то в качестве следующего отрезка выбираем отрезок  $[a, c_1]$ , если же  $R(c_1) > R(c_2)$ , то берут новую точку в середине правой половины (точка  $c_3$ ) и в ней вычисляют функцию. В зависимости от сравнения значений функции в точках  $c_3$  и  $c_1$  выбирают новый отрезок  $[c_1, b]$  или  $[c_2, c_3]$  и т.д.

Второй вариант метода не имеет с точки зрения эффективности принципиального отличия от первого, так как эффективность принято оценивать по наихудшему варианту (т.е. по двум вычислениям  $f(x)$  на каждом шаге). В первом варианте метода есть одна особенность, которая его делает очень эффективным при экспериментальном отыскании экстремума (например, при автоматической настройке технических систем или при практическом поиске наилучших условий деятельности экономического объекта). Малые отклонения от текущей точки обеспечивают в процессе поиска отсутствие “шараханий”, сопровождающихся резкими отклонениями состояния системы.

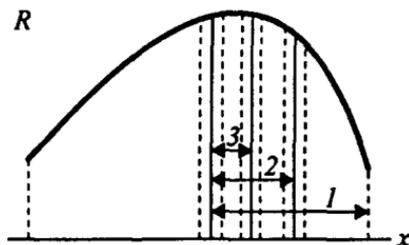


Рис. 14. Иллюстрация метода половинного деления: 1 — интервал, включающий в себя искомый максимум функции после первого этапа (первого деления пополам); 2, 3 — то же соответственно после второго и третьего этапов

### Пример.

Дана функция  $R(x) = D \sin(Ax^B + C)$ , где коэффициенты имеют следующие значения:  $A = 1,0$ ,  $B = 1,0$ ,  $C = 1,0$ ,  $D = 1,0$ . Найти максимум на интервале:  $[-1, 2]$ . Ошибка задается по  $x$ :  $\varepsilon = 0,05$ .

Результаты расчетов. Середина отрезка  $x_0 = 0,5000$ , значение критерия  $R_0 = 0,9975$ , значение  $R(0,5 - \varepsilon/2) = R(0,475) = 0,97922273$ , значение  $R(0,5 + \varepsilon/2) = R(0,525) = 0,9989513$ . Следовательно, искомый максимум лежит в правой половине отрезка, т.е. теперь отрезком является  $[0,5, 2]$ .

Далее приводятся только координаты середин отрезков с номером итерации, значения критерия в них и указывается новый отрезок (правый или левый).

$x_1 = 1,25000000$	$R_1 = 0,77807320$	левый
$x_2 = 0,87500000$	$R_2 = 0,95408578$	левый
$x_3 = 0,68750000$	$R_3 = 0,99319785$	левый
$x_4 = 0,59375000$	$R_4 = 0,99973658$	левый
$x_5 = 0,54687500$	$R_5 = 0,99971390$	

$|x_4 - x_5| < \varepsilon$ , поэтому в качестве решения можно принять любое из этих значений или середину между ними.

Всего восемь раз ( $4 \cdot 2 = 8$ ) вычислялся критерий оптимальности (не считая вычислений непосредственно в середине отрезка, которые не используются в алгоритме метода).

## МЕТОД ЗОЛОТОГО СЕЧЕНИЯ

Метод основан на делении текущего отрезка  $[a, b]$ , где содержится искомый экстремум, на две неравные части, подчиняющиеся правилу золотого сечения, для определения следующего отрезка, содержащего максимум.

Золотое сечение определяется по правилу: отношение всего отрезка к большей его части равно отношению большей части отрезка к меньшей. Ему удовлетворяют две точки  $c$  и  $d$ , расположенные симметрично относительно середины отрезка.

$$\frac{ab}{cb} = \frac{cb}{ac}; \quad \frac{ab}{ad} = \frac{ad}{db}.$$

Путем сравнения  $R(c)$  и  $R(d)$  определяют следующий отрезок, где содержится максимум. Если  $R(d) > R(c)$ , то в качестве следующего отрезка выбирается отрезок  $[c, b]$ , в противном случае — отрезок  $[a, d]$ .

Новый отрезок снова делится на неравные части по правилу золотого сечения. Следует отметить, что точка  $d$  является и точкой золотого сечения отрезка  $[c, b]$ , т.е.

$$\frac{db}{cd} = \frac{cd}{cb}.$$

Поэтому на каждой следующей итерации (кроме “запуска” метода на исходном отрезке) нужно вычислять только одно значение критерия оптимальности.

Существуют аналитические формулы для расчета новой точки на отрезке, где находится максимальное значение  $R(x)$ , которую нетрудно получить:

$$c = a + (b - a) \frac{\sqrt{5} - 1}{2}; \quad d = b - (b - a) \frac{\sqrt{5} - 1}{2}.$$

Условие окончания поиска — величина отрезка, содержащего максимум, меньше заданной погрешности.

Метод обеспечивает более быструю сходимость к решению, чем многие другие методы, и применим, очевидно, только для одноэкстремальных функций<sup>1</sup>.

На рис. 15 приведены два этапа поиска максимума функции методом золотого сечения.

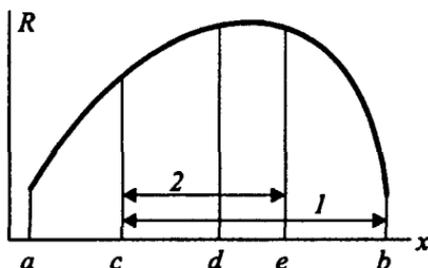


Рис. 15. Иллюстрация метода золотого сечения: 1 — интервал, включающий в себя искомый максимум функции после первого этапа (первого золотого сечения в точках  $c$  и  $d$ ); 2 — то же, после второго этапа (новая точка  $e$  и старая точка  $d$ )

<sup>1</sup> В практических задачах под одноэкстремальной функцией понимают функцию, содержащую один экстремум того типа, который ищется в задаче.

### **Пример.**

Дана функция  $R(x) = \sin(Ax^B + C)$ , где коэффициенты имеют следующие значения:  $A = 1,0, B = 1,0, C = 1,0, D = 1,0$ . Найти максимум на интервале:  $[-1, 2]$ . Ошибка задается по  $x$ :  $\varepsilon = 0,05$ .

Результаты расчетов. Для “запуска” метода найдем две симметричные точки золотого сечения для отрезка  $[-1, 2]$ :

$$x_1 = 0,145898, \quad x_2 = 0,85410197.$$

Значения критериев в этих точках соответственно  $R(x_1) = 0,911080, R(x_2) = 0,960136$ . Следовательно, новым отрезком является  $[0,145898, 2]$ , внутри которого находится максимальное из найденных значений  $R$ . Точка золотого сечения для нового отрезка будет  $x_3 = 0,58359214$ , а  $R(x_3) = 0,99991813$ . Далее приведены только координаты лучших точек при очередном шаге, номер шага и значения критерия в этих точках.

$x_3 = 0,58359214$	$R_3 = 0,99991813$	$x_4 = 0,58359214$	$R_4 = 0,99991813$
$x_5 = 0,58359214$	$R_5 = 0,99991813$	$x_6 = 0,58359214$	$R_6 = 0,99991813$
$x_7 = 0,58359214$	$R_7 = 0,99991813$	$x_8 = 0,55920028$	$R_8 = 0,99993277$
$x_9 = 0,55920028$	$R_9 = 0,99993277$		

Всего было проведено 10 вычислений критерия оптимальности.

### **МЕТОД ПАРАБОЛИЧЕСКОЙ АППРОКСИМАЦИИ**

Метод заключается в замене нелинейной функции  $R(x)$  квадратичной параболой  $R_2(x)$ , построенной по трем точкам, принадлежащим  $R(x)$ , с последующим нахождением  $\max$  параболической функции, используя аналитические условия оптимальности:  $dR/dx = 0$ .

На первом этапе в качестве исходных трех точек используются  $x_1 = a, x_2 = b$  и  $x_3 = (a+b)/2$ . В этих точках вычисляется  $R(x)$  и по полученным точкам  $R(x_1), R(x_2), R(x_3)$  строится парабола  $R_2 = C_2x^2 + C_1x + C_0$ , коэффициенты которой находятся из решения соответствующей системы уравнений:

$$R_2(x_1) = R(x_1), \quad R_2(x_2) = R(x_2), \quad R_2(x_3) = R(x_3).$$

Условие оптимальности приводит к уравнению  $x_4 = -C_1 / (2C_2)$ , где  $x_4$  — точка максимума параболы  $R_2(x)$ . Далее выбирается новый отрезок, внутри которого находится точка  $x_4$ , и, используя  $x_3$ ,  $x_4$ , строится новая парабола, по которой уточняется положение максимума  $R(x)$  и т.д. до тех пор, пока величина отрезка, внутри которого находится максимум, не будет меньше заданной погрешности  $\varepsilon$ . Таким образом, метод имеет итерационный характер. Можно строить параболу на каждом шаге и по трем последним точкам, но только в том случае, если точно известно, что функция гладкая и одноэкстремальная. В противном случае первый вариант даст лучший результат.

К достоинству метода относится высокая скорость сходимости к оптимуму, хотя метод может не всегда сходиться к нему.

На рис. 16 приведены два случая применения метода параболической аппроксимации: а) рассмотрена ситуация, когда метод параболической аппроксимации сходится к решению, уже на третьем этапе парабола, построенная по точкам  $x_3$ ,  $x_4$ ,  $x_5$ , практически совпадает с исходной функцией; б) парабола не имеет максимума уже на втором этапе.

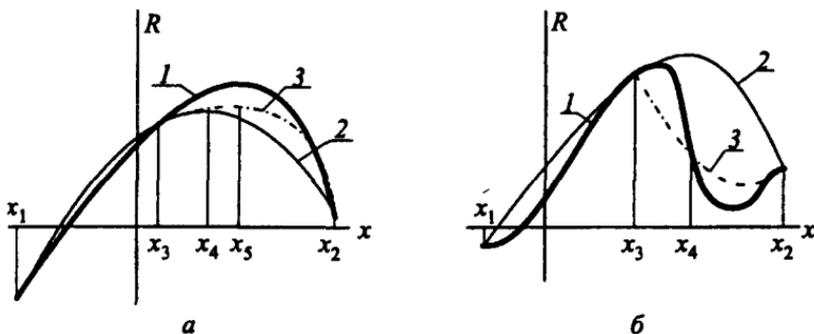


Рис. 16. Иллюстрация метода параболической аппроксимации: а — решение найти можно; б — решение найти нельзя; 1 — функция, экстремум которой ищется; 2 — аппроксимирующая парабола первого этапа, построенная по точкам  $x_1$ ,  $x_2$ ,  $x_3$ ; 3 — аппроксимирующая парабола второго этапа, построенная по точкам  $x_2$ ,  $x_3$ ,  $x_4$ ;  $x_3$  — середина исходного интервала;  $x_4$  — точка максимума первой параболы;  $x_5$  — точка максимума второй параболы

### **Пример.**

Дана функция  $R(x) = D \sin(Ax^B + C)$ , где коэффициенты имеют следующие значения:  $A = 1,0$ ,  $B = 1,0$ ,  $C = 1,0$ ,  $D = 1,0$ . Найти максимум на интервале:  $[-1, 2]$ . Ошибка задается по  $x$ :  $\varepsilon = 0,05$ .

Результаты расчетов. Первая аппроксимирующая парабола строится по точкам:  $x_1 = -1$ ,  $R(-1) = 0$ ;  $x_2 = 0,5$ ,  $R(0,5) = 0,9975$ ;  $x_3 = 2,0$ ,  $R(2,0) = 0,141120$ . Запишем систему уравнений для нахождения коэффициентов параболы:

$$\begin{aligned}1 \cdot C_2 + (-1)C_1 + C_0 &= 0, \\ 0,25C_2 + 0,5C_1 + C_0 &= 0,9975, \\ 4C_2 + 2C_1 + C_0 &= 0,14112.\end{aligned}$$

Решением этой системы является  $C_2 = -0,41197$ ,  $C_1 = 0,459012$ ,  $C_0 = 0,87089$ .

Находим  $x$ , при котором парабола имеет максимум:

$$x = -\frac{C_1}{2C_2} = -\frac{-0,41197}{2 \cdot 0,459012} = 0,55709139,$$

при этом  $R = 0,99990609$ . По этой точке, а также по второй и третьей исходным точкам, лежащим по обе стороны от точки максимума параболы, аналогично строится вторая парабола, максимум которой оказывается в точке  $x = 0,57823785$ , а  $R = 0,99997231$ . Разница между двумя точками максимума менее заданной погрешности, следовательно, можно заканчивать поиск.

В этом методе всего четыре раза вычислялся критерий оптимальности.

## **КОНТРОЛЬНЫЕ ВОПРОСЫ**

### **Метод сканирования**

1. Экстремум каких функций  $R(x)$  можно найти методом сканирования?
2. Основное достоинство метода сканирования.
3. Способ "размещения" точек вычисления критерия оптимальности на оси  $x$ .

4. Основная задача модернизации “базового” метода.
5. Основные достоинства модернизированного метода?
6. Каким образом повысить точность нахождения решения?
7. Условие отыскания оптимального решения.
8. Как найти самое большое значение  $R(x)$ ?
9. Трудно ли метод поддается алгоритмизации, т.е. сложно ли составить алгоритм для решения задачи на ЭВМ?
10. Как влияет вид функции  $R(x)$  на процесс нахождения решения?

### ***Метод деления пополам***

1. Для каких функций  $R(x)$  пригоден метод половинного деления?
2. Каково основное достоинство метода половинного деления?
3. Каков способ “размещения” точек вычисления критерия оптимальности на оси  $x$ ?
4. Каким образом повысить точность нахождения решения  $x^*$ ?
5. Условие отыскания оптимального решения.
6. Как влияет вид функции  $R(x)$  на процесс нахождения решения?
7. Всегда ли метод гарантированно дает решение?
8. Каким образом определяется следующий отрезок, на котором находится экстремум?
9. Сколько раз нужно вычислить  $R(x)$  на отрезке  $[a, b]$ , если хотим найти решение с погрешностью 1% от длины  $[a, b]$ ?
10. Может ли сокращение исходного отрезка  $[a, b]$  обеспечить уменьшение затрат на поиск решения с погрешностью 1% от  $[a, b]$ ?

### ***Метод золотого сечения***

1. Может ли сокращение исходного отрезка  $[a, b]$  обеспечить уменьшение затрат на поиск решения с погрешностью 1% от  $[a, b]$ ?
2. Всегда ли метод гарантированно дает решение?
3. Как влияет вид функции  $R(x)$  на процесс нахождения решения?

4. Каким образом определяется следующий отрезок, на котором находится экстремум?
5. Основное достоинство метода золотого сечения.
6. Каким образом повысить точность нахождения решения?
7. Как влияет вид функции  $R(x)$  на процесс нахождения решения?
8. Что делится по правилу золотого сечения?
9. Если отрезок  $[a, b]$  содержит внутреннюю точку  $c$ , то какое условие называется золотым сечением?
10. Сколько раз нужно вычислить  $R(x)$  на каждом шаге?

### *Метод параболической аппроксимации*

1. Экстремум каких функций  $R(x)$  можно найти методом параболической аппроксимации?
2. Основное достоинство метода параболической аппроксимации.
3. Условие окончания поиска.
4. Каким образом находится аппроксимирующая парабола?
5. К чему может привести увеличение степени аппроксимирующего полинома (с 2 до 3 или 4)?
6. Каким образом повысить точность нахождения решения?
7. Всегда ли метод гарантированно дает решение?
8. Способ формирования точек для построения аппроксимирующей параболы на текущем шаге.
9. Возможно ли нахождение решения задачи оптимизации за один шаг?
10. Как влияет вид функции  $R(x)$  на процесс нахождения решения?

## **МНОГОМЕРНАЯ БЕЗУСЛОВНАЯ ГРАДИЕНТНАЯ ОПТИМИЗАЦИЯ**

---

### **КОНЦЕПЦИЯ МЕТОДОВ**

В данном разделе рассматриваются методы построения улучшающих последовательностей при отыскании экстремума функции  $R(x)$  без активных ограничений. Активными принято назы-

вать такие ограничения, на границе которых находится решение. Если известно, что решение лежит строго внутри допустимой области, например в случае ограничений типа неравенств, то такие ограничения лучше выводить из задачи на этапе ее постановки. Кстати, следует отметить, что ограничения типа равенств всегда активные.

Величина шага  $\Delta x$  в рекуррентном соотношении

$$x^{i+1} = x^i + \Delta x^i$$

вычисляется с использованием градиента целевой функции  $R(x)$ , т.е.

$$\Delta x^i = f(\text{grad } R(x^i)),$$

при этом шаг может определяться с использованием градиента в одной (текущей) или в двух (текущей и предыдущей) точках. Направление градиента, как известно, показывает направления наискорейшего возрастания функции, а его модуль — скорость этого возрастания.

В отличие от других рассмотренных выше вычислительных методов поисковые методы оптимизации содержат неформально (т.е. субъективно) задаваемые параметры, которые существенно влияют на эффективность поиска, вследствие чего один и тот же метод может дать совершенно различные траектории поиска. Поэтому для всех методов, рассматриваемых далее, на рис. 17 приводится

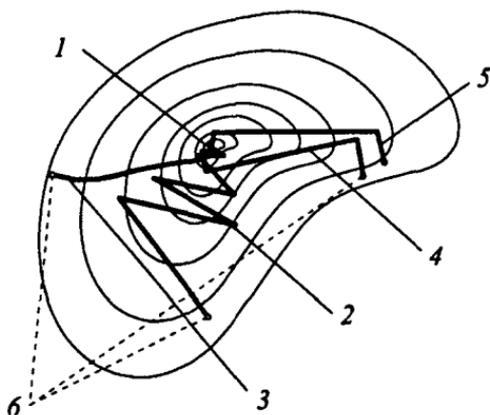


Рис. 17. Иллюстрация траекторий поиска минимума функции градиентными методами: 1 — оптимум; 2 — траектория метода градиента; 3 — траектория метода тяжелого шарика; 4 — траектория метода наискорейшего спуска; 5 — траектория метода сопряженных градиентов; 6 — начальные точки траекторий

лишь одна из возможных траекторий. Кроме того, для всех приведенных траекторий выбраны различные начальные условия, с тем чтобы не загромождать построения. На этом и последующих рисунках зависимость  $R(x_1, x_2)$  приведена в виде линий уровня на плоскости в координатах  $x_1 - x_2$ .

## ОСНОВНЫЕ МЕТОДЫ

### МЕТОД ГРАДИЕНТА

Метод градиента в чистом виде формирует шаг по переменным как функцию от градиента  $R(x)$  в текущей точке поиска. Простейший алгоритм поиска  $\min R(x)$  записывается в векторной форме следующим образом:

$$x^{i+1} = x^i - h \operatorname{grad} R(x^i),$$

или в скалярном виде:

$$x_j^{i+1} = x_j^i - h \frac{dR}{dx_j^i}, \quad j = 1, \dots, n.$$

Величина рабочего шага в направлении градиента  $h \operatorname{grad} R(x)$  зависит от величины градиента, который заранее учесть трудно, и от коэффициента пропорциональности шага  $h$ , с помощью которого можно управлять эффективностью метода.

Поиск каждой новой точки состоит из двух этапов:

- 1) оценка градиента  $R(x)$  путем вычисления частных производных от  $R(x)$  по каждой переменной  $x_j$ ;
- 2) рабочий шаг по всем переменным одновременно.

Величина  $h$  сильно влияет на эффективность метода. Большей эффективностью обладает вариант метода, когда шаг по каждой переменной определяется направляющими косинусами градиента

$$x_j^{i+1} = x_j^i - h \cos \varphi_j,$$

где  $\cos \varphi_j = \frac{(dR / dx_j)}{|\operatorname{grad} R(x)|}$ .

В этом случае величина рабочего шага не зависит от величины модуля градиента, и ею легче управлять изменением  $h$ . В районе оптимума может возникать значительное “рыскание”, поэтому используют различные алгоритмы коррекции  $h$ .

Наибольшее распространение получили следующие алгоритмы:

1.  $h^i = \text{const} = h$  (без коррекции);
2.  $h^i = h^{i-1}/2$ , если  $R(x^i) < R(x^{i-1})$ ;  $h^i = h^{i-1}$ , если  $R(x^i) > R(x^{i-1})$ ;
3.  $h^i = h^{i-1}$ , если  $\alpha_1 \leq \alpha \leq \alpha_2$ ;  $h^i = 2h^{i-1}$ , если  $\alpha_1 > \alpha$ ;  $h^i = \frac{h^{i-1}}{3}$ , если  $\alpha_2 < \alpha$ ,

где  $\alpha$  — угол между градиентами на предыдущем и текущем шаге;  $\alpha_1$  и  $\alpha_2$  — заданные пороговые значения выбираются субъективно (например,  $\alpha_1 = \pi/6$ ,  $\alpha_2 = \pi/3$ ).

Вдали от оптимума направление градиента меняется мало, поэтому шаг можно увеличить (второе выражение), вблизи от оптимума направление резко меняется (угол между градиентами  $R(x)$  большой), поэтому  $h$  сокращается (третье выражение).

Для оценки частных производных используются разностные методы:

1. Алгоритм с центральной пробой

$$\frac{dR}{dx_i} \approx \frac{R(x_1, \dots, x_i + g_i, \dots, x_n) - R(x_1, \dots, x_i, \dots, x_n)}{g_i}$$

2. Алгоритм с парными пробами

$$\frac{dR}{dx_i} \approx \frac{R(x_1, \dots, x_i + g_i, \dots, x_n) - R(x_1, \dots, x_i - g_i, \dots, x_n)}{g_i}$$

где  $g_i$  — пробный шаг по  $i$ -й переменной, выбираемый достаточно малым для разностной оценки производной.

Первый алгоритм требует меньших затрат по сравнению со вторым (обычно затраты выражаются количеством вычислений критерия оптимальности), но позволяет получить решение менее точно, чем второй, и эта погрешность зависит от величины пробного шага.

На рис. 17 приведена одна из возможных траекторий поиска минимума двумерной функции градиентным методом (наряду с другими ниже рассматриваемыми методами).

Условием окончания поиска может являться малость модуля градиента  $R(x)$ , т.е.  $|\text{grad } R(x)| < \varepsilon$ .

**Пример 1.**

1. Требуется найти минимум функции  $R(x_1, x_2) = Ax_1^3 + Bx_2^2 - Cx_1 - Dx_2$ , где  $A = 1, B = 2, C = 3, D = 4$ .
2. Интервал поиска:  $x_{1\text{нач}} = -2, x_{1\text{кон}} = 2, x_{2\text{нач}} = -2, x_{2\text{кон}} = 2$ .
3. Начальная точка:  $x_{10} = -0,5, x_{20} = -1$ .
4. Параметры поиска: коэффициент шага  $h = 0,1$ , пробный шаг  $g = 0,01$ , погрешность  $\varepsilon = 0,01$ .
5. Алгоритм метода: алгоритм 1 ( $x^{i+1} = x^i - h \text{grad}R(x^i)$ ).
6. Алгоритм коррекции шага: без коррекции коэффициента пропорциональности шага ( $h = \text{const}$ ).
7. Способ вычисления производной: вычисление  $\text{grad}R$  с парными пробами.

Результаты вычислений. В начальной точке вычисляем градиент функции:

$$R(x_1+g, x_2) = 7,352, \quad R(x_1-g, x_2) = 7,397, \quad \frac{dR}{dx_1} = -2,2499;$$

$$R(x_1, x_2+g) = 7,295, \quad R(x_1, x_2-g) = 7,455, \quad \frac{dR}{dx_2} = -8.$$

Значение критерия  $R = 7,3750$ . Делаем рабочий шаг, получаем  $x_1 = -0,275, x_2 = -0,2$ .

В новой точке опять вычисляем производные:

$$R(x_1+g, x_2) = 1,656, \quad R(x_1-g, x_2) = 1,712, \quad \frac{dR}{dx_1} = -2,7730;$$

$$R(x_1, x_2+g) = 1,636, \quad R(x_1, x_2-g) = 1,732, \quad \frac{dR}{dx_2} = -4,8.$$

Значение критерия  $R = 1,3750$ .

Делаем рабочий шаг, получаем  $x_1 = 0,002, x_2 = 0,280$ .

Далее аналогично осуществляем следующие шаги (табл. 18).

Таблица 18

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad } R $	$R$
3	0,302	0,568	-2,7258	-1,7280	3,2274	-2,5060
4	0,575	0,741	-2,0085	-1,0368	2,2603	-3,4002
5	0,776	0,844	-1,1947	-0,6221	1,3470	-3,8120
6	0,895	0,907	-0,5958	-0,3732	0,7031	-3,9508
7	0,955	0,944	-0,2652	-0,2239	0,3471	-3,9877
8	0,981	0,966	-0,1112	-0,1344	0,1744	-3,9967
9	0,992	0,980	-0,0453	-0,0806	0,0925	-3,9990
10	0,997	0,988	-0,0183	-0,0484	0,0517	-3,9997
11	0,999	0,993	-0,0073	-0,0290	0,0299	-3,9999
12	0,999	0,996	-0,0029	-0,0174	0,0177	-4,0000
13	1,000	0,997	-0,0012	-0,0104	0,0105	-4,0000
14	1,000	0,998	-0,0005	-0,0063	0,0063	-4,0000

В последней точке модуль градиента меньше заданной погрешности ( $0,0063 < 0,01$ ), поэтому поиск прекращается.

### Пример 2.

Отличается от предыдущего только величиной коэффициента пропорциональности шага  $h$ , теперь  $h = 0,4$ . Ниже, в табл. 19 приведены только первые 14 шагов (как и в предыдущем случае). Целесообразно сопоставить их путем построения траекторий поиска при обоих значениях  $h$  в координатах  $x_1 - x_2$ .

Таблица 19

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad } R $	$R$
1	-0,500	-1,000	-2,2499	-8,0000	8,3104	7,3750
2	0,400	2,200	-2,5200	4,8000	5,4213	-0,2559
3	1,408	0,280	2,9471	-2,8800	4,1207	-2,3960
4	0,229	1,432	-2,8424	1,7280	3,3265	-2,3020
5	1,366	0,741	2,5986	-1,0368	2,7978	-3,4145
6	0,327	1,156	-2,6798	0,6221	2,7511	-2,8967
7	1,399	0,907	2,8681	-0,3732	2,8923	-3,4427
8	0,251	1,056	-2,8104	0,2239	2,8193	-2,7319

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad } R $	$R$
9	1,375	0,966	2,6760	-0,1344	2,6794	-3,5218
10	0,305	1,020	-2,7207	0,0806	2,7219	-2,8860
11	1,393	0,988	2,8244	-0,0484	2,8248	-3,4747
12	0,264	1,007	-2,7914	0,0290	2,7916	-2,7724
13	1,380	0,996	2,7148	-0,0174	2,7148	-3,5114
14	0,294	1,003	-2,7401	0,0104	2,7401	-2,8573
...	...	...	...	...	...	...

В этом случае поиск носит явно колебательный характер, плохо приближаясь к решению.

### МЕТОД НАИСКОРЕЙШЕГО СПУСКА

Основным недостатком градиентного метода является необходимость частого вычисления производных от  $R(x)$ . Этого недостатка лишен метод наискорейшего спуска, который заключается в следующем.

В текущей точке вычисляется  $\text{grad}R(x)$ , и затем в направлении градиента ищется  $\min R(x)$ . Практически это может быть осуществлено любым методом одномерной оптимизации (поиск по одному направлению — направлению градиента), наиболее часто используется сканирование до первого локального минимума по направлению  $\text{grad}R(x)$ .

В результате вдали от оптимума эффективность метода повышается, мы быстрее попадаем в район оптимума, в окрестности которого эффективность метода снижается из-за частой смены направления поиска и приближается к эффективности метода градиента.

Метод, как и все градиентные методы, обладает невысокой эффективностью в овражных функциях. В ряде случаев можно повысить скорость выхода в район оптимума предъявлением невысоких требований к точности поиска  $\min$  по направлению (задается величиной  $h$  — шагом поиска по направлению).

Условием окончания может являться малость модуля градиента  $R(x) \quad |\text{grad}R(x)| < \varepsilon$ . Можно также использовать и малость приращений по переменным в результате шага, но только в том случае, если на данном шаге мы “проскочили” оптимум, иначе может оказаться, что малость шага обусловлена не близостью к оптимуму, а малостью коэффициента пропорциональности шага  $h$ .

В ряде случаев используют уменьшение шага поиска оптимума по направлению после каждой смены направления. Это позволяет с большей точностью каждый раз находить оптимум, но резко снижает эффективность поиска в овражных функциях. Метод используется для локализации “дна оврага” в специальных овражных методах. Условием окончания поиска в этом случае является достижение заданной малой величины шага.

Одна из возможных траекторий поиска минимума двумерной функции методом наискорейшего спуска приведена на рис. 17.

### *Пример.*

Для сравнения с методом градиента рассмотрим решение предыдущего примера при  $h = 0,1$ .

Результаты расчетов. Расчет производных детально рассмотрен выше, поэтому здесь не приводится. Ниже, в табл. 20 приводятся результаты движения по градиенту с постоянным шагом.

Таблица 20

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad}R $	$R$
1	-0,500	-1,000	-2,2499	-8,0000	8,3104	7,3750
1	-0,275	-0,200	-2,2499	-8,0000	8,3104	1,6842
1	-0,050	0,600	-2,2499	-8,0000	8,3104	-1,5301
1	0,175	1,400	-2,2499	-8,0000	8,3104	-2,1996

В следующей точке (0,400, 2,00) значение критерия ( $R = -0,256$ ) оказывается хуже, чем в последней ( $R = -2,1996$ ). Поэтому в найденной точке оптимума по направлению снова вычисляем градиент и по нему совершаем шаги, до тех пор, пока не найдем наилучшую точку (табл. 21).

Таблица 21

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad } R $	$R$
Второй поиск по градиенту						
2	0,175	1,400	-2,9081	1,6000	3,3192	-2,1996
2	0,466	1,240	-2,9081	1,6000	3,3192	-3,1811
2	0,757	1,080	-2,9081	1,6000	3,3192	-3,8239
2	1,047	0,920	-2,9081	1,6000	3,3192	-3,9804
Третий поиск по градиенту						
3	1,047	0,920	0,2912	-0,3200	0,4326	-3,9804
3	1,018	0,952	0,2912	-0,3200	0,4326	-3,9944
3	0,989	0,984	0,2912	-0,3200	0,4326	-3,9991
Четвертый поиск по градиенту						
4	0,989	0,984	-0,0646	-0,0640	0,0909	-3,9991
4	0,996	0,990	-0,0646	-0,0640	0,0909	-3,9998
4	1,002	0,997	-0,0646	-0,0640	0,0909	-4,0000
Пятый поиск по градиенту						
5	1,002	0,997	0,0126	-0,0128	0,0179	-4,0000
5	1,001	0,998	0,0126	-0,0128	0,0179	-4,0000
5	1,000	0,999	0,0126	-0,0128	0,0179	-4,0000

### МЕТОД СОПРЯЖЕННЫХ ГРАДИЕНТОВ

Градиентные методы, базирующиеся только на вычислении градиента  $R(x)$ , являются методами первого порядка, так как на интервале шага они заменяют нелинейную функцию  $R(x)$  линейной.

Более эффективными могут быть методы второго порядка, которые используют при вычислении не только первые, но и вторые производные от  $R(x)$  в текущей точке. Однако у этих методов есть свои труднорешаемые проблемы — вычисление вторых производных в точке, к тому же вдали от оптимума матрица вторых производных может быть плохо обусловлена.

Метод сопряженных градиентов является попыткой объединить достоинства методов первого и второго порядка с исключением их недостатков. На начальных этапах (вдали от оптимума)

метод ведет себя как метод первого порядка, а в окрестностях оптимума приближается к методам второго порядка.

Первый шаг аналогичен первому шагу метода наискорейшего спуска, второй и следующие шаги выбираются каждый раз в направлении, образуемом в виде линейной комбинации векторов градиента в данной точке и предшествующего направления.

Алгоритм метода можно записать следующим образом (в векторной форме):

$$\begin{aligned}x^1 &= x^0 - h \operatorname{grad} R(x^0), \\x^{i+1} &= x^i - h [\operatorname{grad} R(x^i) + \alpha \operatorname{grad} R(x^{i-1})].\end{aligned}$$

Величина  $\alpha$  может быть приближенно найдена из выражения

$$\alpha = \frac{|\operatorname{grad} R(x^i)|^2}{|\operatorname{grad} R(x^{i-1})|^2}.$$

Алгоритм работает следующим образом. Из начальной точки  $x^0$  ищут  $\min R(x)$  в направлении градиента (методом наискорейшего спуска), затем, начиная с найденной точки и далее, направление поиска  $\min$  определяется по второму выражению. Поиск минимума по направлению может осуществляться любым способом: можно использовать метод последовательного сканирования без коррекции шага сканирования при переходе минимума, поэтому точность достижения минимума по направлению зависит от величины шага  $h$ .

Для квадратичной функции  $R(x)$  решение может быть найдено за  $n$  шагов ( $n$  — размерность задачи). Для других функций поиск будет медленнее, а в ряде случаев может вообще не достигнуть оптимума вследствие сильного влияния вычислительных ошибок.

Одна из возможных траекторий поиска минимума двумерной функции методом сопряженных градиентов приведена на рис. 17.

### *Пример.*

Для сравнения рассмотрим решение предыдущего примера.

Первый шаг делаем по методу наискорейшего спуска (табл. 22).

Таблица 22

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad } R $	$R$
0	-0,500	-1,000	-2,2499	-8,0000	8,3104	7,3750
0	-0,275	-0,200	-2,2499	-8,0000	8,3104	1,6842
0	-0,050	0,600	-2,2499	-8,0000	8,3104	-1,5301
0	0,175	1,400	-2,2499	-8,0000	8,3104	-2,1996

Найдена наилучшая точка. Вычисляем производные в этой точке:  $dR/dx_1 = -2,908$ ,  $dR/dx_2 = 1,600$ ; вычисляем коэффициент  $\alpha$ , учитывающий влияние градиента в предыдущей точке:  $\alpha = 3,31920 \cdot 3,3192/8,3104^2 = 0,160$ . Делаем рабочий шаг в соответствии с алгоритмом метода, получаем  $x_1 = 0,502$ ,  $x_2 = 1,368$ . Далее все повторяется аналогично. Ниже, в табл. 23 приведены текущие координаты поиска следующих шагов.

Таблица 23

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad } R $	$R$	$\alpha$
1	0,155	1,303	1,0026	1,2114	1,5725	-3,7407	0,160
2	1,085	0,989	0,5323	-0,0450	0,5342	-3,9774	0,224
3	1,020	0,979	0,1230	-0,0829	0,1484	-3,9978	0,115
4	1,004	0,988	0,0234	-0,0484	0,0537	-3,9996	0,077
5	0,996	1,000	-0,0240	-0,0010	0,0240	-4,0000	0,131
6	1,000	1,002	-0,0001	0,0076	0,0076	-4,0000	0,200

### МЕТОД ТЯЖЕЛОГО ШАРИКА

Метод базируется на аналогии с движением “тяжелого” материального шарика по наклонной поверхности. Скорость шарика при движении вниз будет возрастать, и он будет стремиться занять нижнее положение, т.е. точку минимума. При выводе дифференциального уравнения движения шарика учитывается его масса и вязкость среды, которые влияют на характер его движения, т.е. поиска  $\min R(x)$ .

В дискретном варианте траектория поиска описывается следующим алгоритмом:

$$x^{i+1} = x^i - \alpha(x^i - x^{i-1}) - h \text{grad} R(x^i).$$

При  $\alpha = 0$  метод превращается в обыкновенный градиентный. При  $\alpha = 1$  поиск не затухает, следовательно, при  $0 < \alpha < 1$  можно получать различную эффективность метода, которая будет зависеть и от  $h$ .

Вдали от оптимума поиск будет ускоряться, а вблизи возможны колебания около точки  $\min R(x)$ .

К недостаткам метода относится необходимость задания сразу двух неформальных параметров, определяющих эффективность поиска. К достоинствам метода, помимо ускорения движения вдали от оптимума, относится возможность "проскока" мелких локальных "ямок" (минимумов) за счет "инерционности шарика", т.е. можно решать и задачу глобальной оптимизации для функции  $R(x)$  с одним явно выраженным минимумом и многими "мелкими".

Одна из возможных траекторий поиска минимума двумерной функции методом тяжелого шарика приведена на рис. 17.

### *Пример.*

Для сравнения методов рассмотрим решение предыдущего примера. Результаты вычислений при  $\alpha = 0,5$  и  $h = 0,1$  приведены ниже в кратком изложении, так как никаких принципиально новых элементов здесь нет, кроме формулы для вычисления рабочего шага. Обратим внимание на то, что первый шаг делается обычным методом градиента (результаты полностью совпадают с результатами метода градиента), так как мы еще не имеем предыдущей точки (табл. 24).

Таблица 24

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad } R $	$R$
0	-0,500	-1,000	-2,2499	-8,0000	8,3100	7,3750
1	-0,275	-0,200	-2,7730	-4,8000	5,5430	1,6842
2	-0,110	-0,120	-2,9635	-4,4800	5,3710	0,8381
3	0,104	0,288	-2,9676	-2,8480	-6,1130	-1,2962
4	0,294	-0,369	-2,7414	-2,5248	0,7269	-2,0585
5	0,473	-0,581	-2,3293	-1,6765	0,8699	-2,9613
6	0,616	-0,642	-1,8612	-1,4300	0,3472	-3,3588
7	0,731	0,755	-1,3988	-0,9812	0,7086	-3,6814

№	$x_1$	$x_2$	$dR/dx_1$	$dR/dx_2$	$ \text{grad } R $	$R$
8	0,813	0,797	-1,0160	-0,8131	0,3014	-3,8192
9	0,873	0,857	-0,7110	-0,5719	0,9125	-3,9131
10	0,914	0,884	-0,4913	-0,4638	0,6756	-3,9518
11	0,943	0,917	-0,3316	-0,3323	0,4695	-3,9767
12	0,962	0,934	-0,2240	-0,2651	0,3471	-3,9869
13	0,975	0,952	-0,1486	-0,1927	0,2433	-3,9935
14	0,983	0,962	-0,0994	-0,1518	0,1815	-3,9963
15	0,989	0,972	-0,0653	-0,1115	0,1293	-3,9981

Метод не оказался более эффективным по сравнению с другими. Это обусловлено проблемами с подбором параметров поиска  $\alpha$  и  $h$ .

Для наглядного сравнения методов целесообразно по приведенным результатам поиска различными методами самостоятельно построить траектории поиска в координатах  $x_1 - x_2$ , соединив отрезками прямых все точки для каждого метода.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

### *Метод градиента*

1. При каком из алгоритмов выбора направления поиска  $\text{max } R(x)$  метод будет более эффективен?
2. Как изменяется угол между двумя соседними направлениями поиска при приближении к оптимуму?
3. Что называется градиентом функции  $R(x_1, x_2)$ ?
4. Свойства градиента функции  $R(x)$ .
5. Как оценивается эффективность поиска градиентным методом?
6. Какой алгоритм коррекции шага предпочтительнее вблизи оптимума?
7. Почему в районе оптимума величина шага  $\Delta x$  убывает при использовании алгоритма  $x^j = x^{j-1} - h \text{grad } R(x)$ ?

8. В чем отличие двух алгоритмов градиентного метода:

$$x_i^j = x_i^{j-1} - h \text{grad}R(x^j), \quad x_i^j = x_i^{j-1} - h \cos \varphi_j,$$

где  $\cos \varphi_j$  — направляющие косинусы градиента.

9. Исходя из определения  $\text{grad}R(x)$  как вектора, указывающего направление возрастания функции, что лучше искать:  $\min$  или  $\max$ ?
10. Что дает вычисление производных по методу с парными пробами?

### *Метод наискорейшего спуска*

1. В чем основные отличия метода наискорейшего спуска от метода градиента?
2. По какому направлению осуществляется поиск из каждой текущей точки при поиске  $\min R(x)$ ?
3. Как вычисляется градиент  $R(x)$  в методе наискорейшего спуска?
4. Почему после нахождения  $\min R(x)$  по направлению необходимо еще раз искать  $\min R(x)$  по другому направлению?
5. Каковы условия окончания поиска?
6. Область наивысшей эффективности метода.
7. Какой метод вычисления шага при поиске  $\min R(x)$  по  $\text{grad}R(x)$  более предпочтителен?
8. Можно ли методом наискорейшего спуска найти  $\max R(x)$ ?
9. Можно ли применять алгоритм коррекции шага поиска, определяемый изменением угла между градиентами в текущей и предыдущей точках?
10. Какое влияние оказывают вычислительные погрешности при поиске  $\min R(x)$  в направлении градиента на точность получения решения?

### *Метод сопряженных градиентов*

1. Чем отличаются квадратичные методы оптимизации от линейных?
2. Какова сравнительная эффективность метода сопряженных градиентов и наискорейшего спуска вблизи от оптимума?

3. Как записывается алгоритм метода сопряженных градиентов?
4. Как влияют вычислительные погрешности на эффективность метода сопряженных градиентов?
5. Для каких функций  $R(x)$  метод сопряженных градиентов наиболее эффективен?
6. В чем недостатки использования методов второго порядка?
7. В чем отличие первого шага в методах наискорейшего спуска и сопряженных градиентов?
8. Какая процедура поиска осуществляется на каждом шаге?
9. Сравнительная эффективность метода градиента и метода сопряженных градиентов вдали от оптимума.
10. Возможно ли применение метода для недифференцируемых функций?

### *Метод тяжелого шарика*

1. Как влияет масса шарика на характер поиска, учитывая, что траектория поиска аналогична движению шарика в вязкой среде?
2. Может ли поиск ускориться?
3. Можно ли найти  $\max R(x)$ , а не  $\min R(x)$  методом тяжелого шарика?
4. В чем заключаются недостатки метода тяжелого шарика?
5. Является ли метод тяжелого шарика пригодным для одномерной оптимизации (т.е. когда у “шарика” нет объема, а следовательно, и массы)?
6. Можно ли найти глобальный минимум  $R(x)$  методом тяжелого шарика?
7. Зачем “помещают шарик в вязкую среду”?
8. Какой путь можно выбрать для затухания поиска в районе оптимума при использовании алгоритма  $x^{j+1} = x^j - \alpha(x^j - x^{j-1}) - h \text{grad} R(x)$ ?
9. Можно ли отнести метод тяжелого шарика к методам второго порядка?
10. В каких условиях предпочтительнее использовать метод тяжелого шарика?

# МНОГОМЕРНАЯ БЕЗГРАДИЕНТНАЯ ОПТИМИЗАЦИЯ

## КОНЦЕПЦИЯ МЕТОДОВ

В данном разделе рассматриваются численные методы оптимизации, у которых величина и направление шага к оптимуму формируются однозначно по определенным детерминированным функциям в зависимости от свойств критерия оптимальности в окрестности текущей точки без использования производных (т.е. градиента). Все алгоритмы имеют итерационный характер и выражаются формулой

$$x^{j+1} = x^j + f [R(x^j)]$$

Основная особенность рассматриваемой группы методов — отсутствие вычисления градиента критерия оптимальности. Ряд методов прямого поиска базируется на последовательном применении одномерного поиска по переменным или по другим задаваемым направлениям, что облегчает их алгоритмизацию и применение.

Как и для градиентных методов, на рис. 18 приводятся лишь по одной из возможных траекторий поиска каждым из ниже рассматриваемых методов. Кроме того, также для всех приведенных траекторий выбраны различные начальные условия, с тем чтобы не загромождать построения.

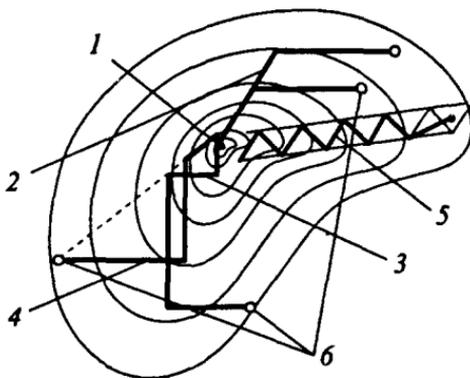


Рис. 18. Иллюстрация траекторий поиска минимума функции безградиентными детерминированными методами: 1 — оптимум; 2 — траектория метода параллельных касательных; 3 — траектория метода Гаусса — Зайделя; 4 — траектория метода Розенброка; 5 — траектория симплексного метода; 6 — начальные точки поиска

# ОСНОВНЫЕ МЕТОДЫ

## МЕТОД ГАУССА — ЗАЙДЕЛЯ

Метод Гаусса — Зайделя (в математической литературе используется и другое название — метод покоординатного спуска) заключается в последовательном поиске оптимума  $R(x)$  поочередно по каждой переменной. Причем после завершения перебора всех переменных (т.е. после завершения одного цикла) опять в общем случае приходится перебирать все переменные до тех пор, пока не придем к оптимуму.

В ряде случаев (для сепарабельных критериев оптимальности, т.е. таких  $R(x_1, x_2, \dots, x_i, \dots, x_n)$ , которые можно представить в виде

$$R(x) = f(x_1) + f(x_2) + \dots + f(x_i) + \dots + f(x_n),$$

удается получить решение всего за один цикл. В случае тесной нелинейной взаимосвязи переменных (например, при наличии произведения переменных и т.п.) для получения решения приходится делать очень много циклов.

Метод обладает низкой эффективностью в овражных функциях, может застревать в “ловушках”, особенно при сравнительно больших шагах  $h$  при поиске оптимума по каждой переменной, очень чувствителен и к выбору системы координат. Метод прост в реализации. На эффективность метода влияет порядок чередования переменных.

Одна из возможных траекторий поиска минимума двумерной функции методом Гаусса — Зайделя приведена на рис. 18. В качестве начальной изменяемой переменной в каждом цикле принята  $x_2$ .

Условием окончания поиска является малость изменения критерия оптимальности за один цикл или невозможность улучшения критерия оптимальности ни по одной из переменных.

Для нейтрализации недостатков разработаны модификации алгоритма, среди которых рассмотрим метод поиска с последствием.

При нахождении оптимума по каждой переменной прекращают поиск не в точке оптимума, а несколько пройдя ее. При этом

удается “выскочить из ловушек”, за меньшее число циклов выйти в район оптимума. В районе оптимума наблюдается зацикливание, и в этом случае последовательно уменьшают величину последствия.

В двумерных задачах метод Гаусса — Зайделя фактически сводится к методу наискорейшего спуска, так как в обоих методах траектория поиска представляет собой последовательность взаимноортогональных отрезков.

*Пример.* Требуется найти минимум функции

$$R(x_1, x_2) = Ax_1^2 + Bx_2^2 + Cx_1 \sin(Dx_1 x_2),$$

где  $A = 1, B = 2, C = 1,2, D = 2$ .

2. Интервал поиска:  $x_{1\text{нач}} = -2, x_{1\text{кон}} = 2, x_{2\text{нач}} = -2, x_{2\text{кон}} = 2$ .

3. Начальная точка:  $x_{10} = 1,4962, x_{20} = -1$ .

4. Параметры поиска: шаг  $h = 0,1$ , погрешность  $= 0,01$ .

**Результаты вычислений.** Из начальной точки  $(1,4962, -1,0000)$  с  $R = 3,9719$  ищем минимум критерия оптимальности по переменной  $x_1$ . Используем прием последовательного сканирования, т.е. “шагаем” до первого лучшего значения критерия, применяя алгоритм  $x_1^{i+1} = x_1^i \pm h$ . Нижний индекс — номер переменной, верхний — номер шага. Знак “+” или “-” выбирается в зависимости от направления изменения критерия: нужно взять такой знак, при котором критерий уменьшается. Наилучшей в этом направлении оказывается точка с координатами  $(0,6962, -1,0000)$  и  $R = 1,30374$ , полученная после семи шагов. Из этой точки ищем минимум критерия по переменной  $x_2$  тем же методом. Находим точку  $(0,6962, -0,3000)$  с  $R = 0,3258$  тоже после семи шагов. На этом заканчивается первый цикл поиска и начинается следующий, заключающийся опять в поиске  $\min R$  по переменной  $x_1$ , затем по  $x_2$ . Далее в табл. 25 приведены лишь координаты точек и значения критериев в концах циклов.

Таблица 25

№ цикла	$x_1$	$x_2$	$R$
2	-0,0038	0,0000	0,0000
3	-0,0038	0,0000	0,0000

После третьего цикла ни по одной из переменных (при изменении их с шагом  $h = 0,1$ ) не удастся получить меньшее значение  $R$ . Следовательно, последнюю точку считаем решением.

### МЕТОД РОЗЕНБРОКА

Метод Розенброка направлен на ликвидацию одного из недостатков метода Гаусса — Зайделя — высокую чувствительность эффективности к выбору системы координат. Метод сводится по сути к отысканию “удачной” системы координат путем поворота исходных осей координат. После этого цикл поиска осуществляется поочередно по всем переменным последовательно.

Первый цикл поиска полностью совпадает с методом Гаусса — Зайделя. Пусть исходная точка  $x^0$ , а точка, в которой завершился первый цикл, —  $x^1$ . Вектор смещения изображающей точки поиска будет равен  $S = x^1 - x^0$ . Чем больше одна составляющая вектора  $S$ , тем существеннее в данной текущей ситуации эта переменная (по ней произошло основное движение). Переход к новой системе координат должен учитывать это обстоятельство. Поэтому преобразовывать систему координат из  $x_1^i, x_2^i, \dots, x_n^i$  в систему  $x_1^{i+1}, x_2^{i+1}, \dots, x_n^{i+1}$  надо таким образом, чтобы наиболее существенная переменная была бы в направлении по вектору  $S$ . По этому направлению должна быть направлена та переменная, по которой поиск будет осуществляться в первую очередь (т.е.  $x_1^{i+1}$ ). Тогда новые оси (в общем случае на  $(i+1)$ -м шаге) определяются следующим образом из условия их ортогональности. Новый базис системы координат:

$$\xi_1^{i+1} = \sum_{j=1}^n S_j^i e_j^i, \quad \xi_2^{i+1} = \sum_{j=2}^n S_j^i e_j^i, \quad \xi_3^{i+1} = \sum_{j=3}^n S_j^i e_j^i \quad \text{и т.д.,}$$

где  $e_j^i$  — орты исходной системы координат. Для ортогонализации и нормирования базиса переходят к ортам новой системы координат стандартным образом:

$$e_1^{i+1} = \frac{\xi_1^{i+1}}{|\xi_1^{i+1}|}, \quad e_2^{i+1} = \frac{\xi_2^{i+1} - (\xi_2^{i+1}, e_1^{i+1}) e_1^{i+1}}{|\xi_2^{i+1} - (\xi_2^{i+1}, e_1^{i+1}) e_1^{i+1}|}$$

и т.д. Здесь в круглых скобках обозначено скалярное произведение векторов. Первая новая координата  $x_1^{i+1} = x_1^i e_1^{i+1}$  является самой важной, остальные могут быть получены и произвольно, соблюдая условия их ортогональности. Отмеченные преобразования (поворот осей) осуществляются после каждого цикла поиска.

В случае низкой размерности задачи ( $n = 2, 3$ ) можно непосредственно пользоваться формулами поворота осей координат, известными из математики. Например, для двухмерной задачи с учетом используемых обозначений будем иметь

$$x_1^{i+1} = x_1^i \cos \varphi + x_2^i \sin \varphi, \quad x_2^{i+1} = -x_1^i \sin \varphi + x_2^i \cos \varphi,$$

где  $\varphi$  — угол поворота, легко определяемый как отношение координат вектора  $S$ .

Особенно метод Розенброка эффективен для квадратичных функций, где оптимум может быть найден после одного преобразования осей. В общем случае метод Розенброка на каждом цикле обеспечивает такое изменение координатных осей, при котором направление первого спуска очередного цикла стремится к оптимальному, т.е. к антиградиентному.

Одна из возможных траекторий поиска минимума двухмерной функции методом Розенброка приведена на рис. 18.

### *Пример.*

Для сравнения рассмотрим решение примера предыдущего раздела.

Первый цикл полностью совпадает с методом Гаусса — Зайделя (что следует из сути метода). Поэтому из начальной точки  $x_{10} = 1,4962$ ,  $x_{20} = -1,0$  с  $R = 3,9719$  попадаем в точку  $x_1(0,6962, -0,3000)$  с  $R = 0,3258$ . В рассматриваемом двухмерном случае в следующем цикле будем искать минимум функции вдоль нового направления  $x_1^2$ , т.е. повернем оси координат.

Вектор  $S$  имеет следующие координаты:  $(-0,7, -0,7)$ , следовательно, угол поворота осей координат составит  $\pi/4$ , так как  $\text{tg}(S_1/S_2) = \text{tg}(0,7/0,7) = \text{tg}(1) = \pi/4$ . В данном случае это означает, что надо изменять обе переменные старой системы координат одновременно в одинаковом отношении — это и будет изменение

переменной в новой (повернутой) системе координат. Чтобы шаг по новой переменной был бы равен 0,01, нужно по исходным переменным совершать шаги  $h = 0,01 \cos(\pi/4) \approx 0,007$ . Шагаем по новой оси повернутой системы координат (фактически одновременно изменяем обе переменные в старой системе координат в необходимом соотношении). Лучшей оказывается точка с координатами (0,1864, 0,509), из нее продолжаем поиск по второй новой переменной (т.е. фактически изменяем одновременно обе переменные в старой системе координат в необходимом соотношении в соответствии с формулой поворота осей координат). Второй цикл поиска заканчивается в точке с координатами (0,0554, 0,0189), вектор  $S$  имеет координаты (0,6408, 0,3189). Следовательно, чтобы получить новую систему координат, мы должны повернуть предыдущую на угол  $\varphi = \arctg(0,6408/0,3189)$ . Можно новую систему координат выразить через исходную, учитывая предыдущий поворот осей и поворачивая ее на суммарный угол.

## СИМПЛЕКСНЫЙ МЕТОД

Симплексом в  $n$ -мерном пространстве называют фигуру, содержащую  $n+1$  вершину. На плоскости — это треугольник, в трехмерном пространстве — тетраэдр и т.д. Если все вершины симплекса равно удалены друг от друга, то такой симплекс называется *регулярным*. В литературе можно встретить и другое название метода — *метод деформируемого многогранника*. В организации алгоритма поиска используется важное свойство симплекса: против каждой вершины находится только одна грань. Суть метода заключается в следующем. В окрестности начальной точки  $x^0$  строим симплекс, затем находится самая “плохая” его вершина (т.е. та, в которой наихудшее значение критерия оптимальности) и на противоположной грани строится новый симплекс, отличающийся от исходного только одной вершиной. Эта вершина получается симметричным отражением выбрасываемой вершины относительно центра противоположной грани. Центр грани определяется геометрически, как среднее значение

по каждой проекции из всех вершин грани. Алгоритм получения новой вершины записывается следующим образом:

$$\tilde{x}^j = \frac{2}{n} (x^1 + x^2 + \dots + x^{n+1}) - \left(1 + \frac{2}{n}\right) x^j,$$

где  $x^1, x^2, \dots$  — вектора вершины симплекса (координаты вершин),  $x^j$  — вектор выбрасываемой вершины,  $\tilde{x}^j$  — новая вершина в новом симплексе. В скалярном представлении для каждой координаты будет аналогичная формула.

После построения нового симплекса требуется лишь одно вычисление критерия оптимальности: только в новой вершине, так как в остальных углах они вычислены. Далее все повторяется снова. При выходе в район оптимума процесс поиска “зацикливается”. Это имеет место тогда, когда приходится выбрасывать только что полученную вершину (эта ситуация возникает в том случае, если значение критерия в новой вершине оказывается самое плохое). В этом случае можно “сжимать” симплекс, откладывая новую вершину от грани на расстоянии вдвое меньше, чем необходимо.

$$\tilde{x}^j = \frac{3}{2n} (x^1 + x^2 + \dots + x^{n+1}) - \left(1 + \frac{3}{2n}\right) x^j.$$

Процесс сжатия происходит многократно, до тех пор пока размеры симплекса не будут меньше заданных или пока наибольшее расстояние между вершинами симплекса (длина ребра) не будет меньше заданной величины. Под размерами симплекса понимается расстояние от его центра до всех вершин (в этом случае расстояние от центра до всех вершин должно быть меньше заданного), а иногда и периметр симплекса, т.е. сумма всех его ребер.

Основным недостатком метода является невозможность ускорения поиска вдали от оптимума. Этот недостаток устранен в одной из модификаций метода, известной как метод Нелдера — Мида. В этом варианте симплексного метода предусмотрено растяжение симплекса в случае, если новая вершина лучше лучшей в старом симплексе, а также сжатия симплекса, если новая вершина оказывается хуже наихудшей в старом симплексе.

Формула растяжения имеет вид:

$$\hat{x}^j = \gamma x^j + (1 - \gamma)x^a,$$

где  $\gamma$  — коэффициент растяжения.

Формула сжатия имеет следующий вид:

$$\hat{x}^j = \beta x^j + (1 - \beta)x^a,$$

где  $\beta$  — коэффициент сжатия;  $x^a$  — вектор центра противоположащей грани,  $\tilde{x}^j$  — “отраженная” новая вершина по алгоритму (1),  $\hat{x}^j$  — новая вершина.

Имеется также операция уменьшения размера симплекса, называемая *редукцией*, которая вводится в случае заикливания. При этом уменьшаются все грани симплекса одновременно в два раза. Уменьшение размера симплекса в два раза осуществляется делением пополам расстояния от каждой точки симплекса до точки  $x_1$ , определяющей наименьшее значение функции, т.е. будет:  $x_i = x_i^c + 0,5(x_i - x_1)$ .

Более эффективной является модификация метода, в котором отражение при построении новой вершины осуществляется не относительно центра противоположащей грани, а относительно ее центра тяжести (при поиске  $\max R(x)$  или точки, симметричной центру тяжести на грани при поиске  $\min R(x)$ ). При этом под тяжестью вершин понимается значение критериев оптимальности в вершинах. В этом случае направление деформирования симплекса тяготеет в сторону вершин с лучшим значением критерия. Обе модификации могут совмещаться.

Одна из возможных траекторий поиска минимума двумерной функции базовым алгоритмом симплексного метода приведена на рис. 18.

### *Пример.*

Для сравнения рассмотрим решение примера предыдущего раздела, только параметр  $h$  примем равным 0,5 (он характеризует размер симплекса).

В окрестности заданной начальной точки (1,4962, -1,0000),  $R = 3,9719$  сформируем начальный симплекс, его координатами являются (табл. 26):

Таблица 26

№ вершины	$x_1$	$x_2$	$R$
1	1,7462	-0,8557	4,1938
2	1,2462	-0,8557	1,7519
3	1,4962	-1,2887	6,7368

Конечно, можно и по-другому построить начальный симплекс, например, заданная начальная точка может являться одной из вершин симплекса, а не лежать внутри симплекса, как в данном случае. Это не принципиально и на результат существенно не влияет.

Нетрудно заметить, что самой плохой вершиной является третья (самое большое значение критерия). Удалим эту вершину и на противоположной грани построим новый симплекс. Практически это означает, что нужно найти одну вершину нового симплекса. Применим базовый алгоритм для каждой переменной  $x_1$  и  $x_2$  отдельно.

Получим новую вершину:

№ вершины	$x_1$	$x_2$	$R$
4	1,4962	-0,4226	0,8839

Теперь симплекс образуют вершины 1, 2, 4. Самая плохая вершина — 4. По аналогичной формуле перейдем к следующей новой вершине, “отразив” удаляемую относительно центра грани.

Получим вершину:

№ вершины	$x_1$	$x_2$	$R$
5	0,9962	-0,4226	0,4578

Теперь симплекс образуют вершины 2, 4, 5. Самая плохая вершина — 2. Строим следующую вершину:

№ вершины	$x_1$	$x_2$	$R$
6	1,2462	0,0104	1,5919

Оказалось, что только что полученную вершину нужно будет удалять как самую плохую. Следовательно, необходимо сжимать симплекс, воспользовавшись второй формулой алгоритма.

№ вершины	$x_1$	$x_2$	$R$
7	1,2462	-0,6392	0,8750

Далее приведем без комментариев координаты вершин следующих симплексов, предоставив учащимся самостоятельно определять самую плохую вершину в каждом текущем симплексе (табл. 27).

Таблица 27

№ вершины	$x_1$	$x_2$	$R$	
8	0,7462	-0,6392	0,6435	отражение
9	0,4962	-0,4226	0,3610	отражение
10	0,7462	-0,2061	0,3707	отражение
11	0,2462	-0,2061	0,1157	отражение
...	...	...	...	...
15	0,4962	0,0104	0,2526	отражение
16	-0,0663	-0,1520	0,0490	сжатие
17	-0,3163	0,0645	0,1238	отражение
18	0,1056	-0,1385	0,0458	сжатие
..	...	...	...	...
24	0,0089	0,0704	0,0100	отражение
25	-0,1014	0,0814	0,0256	отражение
26	0,0546	0,0199	0,0039	сжатие

Симплекс быстро сжимается вокруг искомой оптимальной точки.

### МЕТОД ПАРАЛЛЕЛЬНЫХ КАСАТЕЛЬНЫХ

Метод параллельных касательных рассмотрим на примере двухмерной задачи  $R(x_1, x_2)$ . Он заключается в следующем. Из двух произвольных точек  $x^{10}, x^{20}$ , не лежащих на одной прямой заданного направления (например, вдоль одной переменной), проводят два спуска по направлению и находят две точки оптимумов  $x^{11}$  и  $x^{21}$ . Далее оптимум ищут на прямой, соединяющей эти точки. Во всех поисках по направлению могут применяться любые одномерные методы поиска. После отыскания оптимума  $x^3$  вдоль на-

правления  $x^{11} - x^{21}$  опять ищут оптимум из точки  $x^3$  в первоначально заданном направлении и находят точку  $x^{31}$ , затем опять в направлении  $x^{31} - x^{11}$  ищут одномерным методом оптимум и т.д.

В качестве исходного направления задается обычно направление одной из координатных осей (по  $x_1$  или по  $x_2$  и т.д.), хотя может задаваться любое направление.

Для квадратичных функций поиск заканчивается всего за три одномерных поиска, для неквадратичных — это итерационная процедура, сходящаяся к решению тем быстрее, чем ближе  $R(x_1, x_2)$  к квадратичной функции.

Для трехмерной задачи (в случае квадратичного критерия оптимальности) необходимо сначала найти за три одномерных поиска оптимум в одной плоскости (например,  $x_3 = \text{const}_1$ ), затем в другой, параллельной ей ( $x_3 = \text{const}_2$ ), далее потребуется один спуск вдоль направления точек оптимума в этих плоскостях.

В случае  $n$ -мерной квадратичной задачи общее число одномерных поисков будет определяться так:

$$N_n = 2N_{n-1} + 1.$$

Это число быстро растет с ростом размерности задачи. В целом метод успешно может применяться для задач невысокой размерности для функций, близких к квадратичным.

Одна из возможных траекторий поиска минимума двумерной функции методом параллельных касательных приведена на рис. 18.

### *Пример.*

Для сравнения рассмотрим пример предыдущего раздела. Одна начальная точка  $x_{10} (1,4962, -1,0)$  совпадает с предыдущими точками в методах этой группы, в качестве другой выберем точку  $x_{20} (1,0, 1,0)$  с  $R = 3,9719$ . Из первой точки ищем минимум по переменной  $x_1$ , как и в методе Гаусса — Зайделя, получаем в результате точку  $x^{11} (0,6962, -1)$  с  $R = 1,6625$ . Из второй точки аналогичным образом ищем минимум и получаем точку  $x^{21} (0, 1)$  с  $R = 2$ . В направлении через эти две точки опять ищем минимум критерия. Для этого можно получить уравнение прямой типа  $x_2 = k_1 x_1 + k_2$ , проходящей через две данные точки  $x^{11}$  и  $x^{21}$ , изменять с шагом

одну из переменных, например  $x_1$ , и вычислять соответствующее прямой значение  $x_2$ . Такое уравнение в нашем случае имеет вид

$$x_2 = 2,8727x_1 - 1.$$

Меняем  $x_1$  и находим наилучшую точку на этой прямой (здесь для более точного нахождения применен следующий прием: из найденной лучшей точки при поиске с начальным шагом  $h = 0,1$  еще раз ищем точку с  $h = 0,01$ , т.е. более точно, затем еще раз уточняют положение лучшей точки с еще более мелким шагом  $h$ ). Найденная таким образом точка имеет координаты  $x^3$  (0,3778, 0,0552) с  $R = 0,1239$ . Из этой точки опять ищем минимум по  $x_1$  с уменьшающимся шагом и получаем новую точку  $x^{31}$  (-0,0315, -0,0552) с  $R = 0,1239$ . Теперь аналогично предыдущему этапу ищем минимум по направлению от точки  $x^{11}$  (0,6962, -1) к точке  $x^3$  (-0,0315, -0,0552) и получаем точку  $x^4$  с координатами (-0,0325, -0,0556) и  $R = 0,0071$ . Эту точку не удастся улучшить поиском по  $x_1$ , поэтому принимаем ее за решение.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

### *Метод Гаусса — Зайделя*

1. Достаточно ли провести поиск оптимума поочередно по всем переменным последовательно?
2. Область наиболее предпочтительного использования метода Гаусса — Зайделя.
3. Можно ли найти оптимум за один цикл для квадратичной функции?
4. Можно ли найти решение за один шаг для сепарабельной функции?
5. Может ли оказывать влияние на результат поиска (значение оптимума) порядок чередования переменных при поиске?
6. Может ли оказывать влияние на эффективность поиска порядок чередования переменных?
7. Условие окончания поиска  $\min R(x)$ .
8. Зачем необходима модификация — поиск с последствием?
9. Основное достоинство метода.
10. Основной недостаток метода.

## ***Метод Розенброка***

1. Как можно “запустить” метод?
2. В каких случаях возможно “зацикливание” поиска?
3. Для каких функций  $R(x)$  метод наиболее предпочтителен?
4. Существует ли единственное, однозначно определяемое положение осей координат в начале каждого нового цикла?
5. Обязательно ли преобразовывать систему координат так, чтобы она становилась правой?
6. При оптимизации сепарабельной трехмерной не квадратичной функции  $R(x)$  сколько раз придется поворачивать оси координат?
7. Сколько одномерных поисков будет сделано, если ищется оптимум квадратичной функции?
8. При оптимизации сепарабельной трехмерной квадратичной функции  $R(x)$  сколько раз придется поворачивать оси координат?
9. Обязательно ли поворачивать оси координат при поиске экстремума  $n$ -мерной квадратичной сепарабельной функции  $R(x)$ ?
10. Будет ли метод эффективно работать при квадратичной сильно овражной функции  $R(x)$ ?

## ***Симплексный метод***

1. Что называется симплексом?
2. Как находится вершина нового (следующего) симплекса?
3. Признак зацикливания симплексного поиска.
4. Причина зацикливания поиска.
5. Условие растяжения симплекса в процессе поиска.
6. Алгоритм сжатия симплекса в процессе поиска в одной из модификаций.
7. Условие окончания поиска.
8. Как находится точка, относительно которой отражается новая вершина следующего симплекса?
9. Какая из модификаций метода эффективнее работает в овражных функциях?
10. Какой из алгоритмов соответствует базовому (исходному) симплексному методу?

### *Метод параллельных касательных*

1. Каким образом можно выбрать направление исходных одномерных поисков?
2. Для каких функций  $R(x)$  метод наиболее эффективен?
3. Основное достоинство метода.
4. Для задач какой размерности метод будет более эффективным?
5. При каких условиях метод является итерационным?
6. Обязательно ли оптимум в двумерной квадратичной задаче будет располагаться между двумя точками оптимума по параллельным направлениям?
7. Зависит ли эффективность метода для квадратичной сепарабельной функции  $R(x)$  от выбора начальных направлений поиска?
8. Зависят ли общие затраты на поиск от выбора начальной точки?
9. Зависит ли число одномерных поисков для квадратичной функции  $R(x)$  от погрешности нахождения оптимума по направлению?
10. Условие окончания поиска.

## **МНОГОМЕРНАЯ СЛУЧАЙНАЯ ОПТИМИЗАЦИЯ**

---

### **КОНЦЕПЦИЯ МЕТОДОВ**

В методах случайного поиска величина шага  $\Delta x$  при построении улучшающей последовательности  $x^{i+1} = x^i + \Delta x^i$  формируется случайным образом. Поэтому в одной и той же ситуации шаг  $\Delta x$  может быть различен в отличие от регулярных методов. “Методы случайного поиска являются прямым развитием известного метода проб и ошибок, когда решение ищется случайно, и при удаче принимается, а при неудаче отвергается, с тем чтобы немедленно снова обратиться к случайности как к источнику возможностей. Такое случайное поведение разумно опирается на уверенность, что случайность содержит в себе все возможности, в том числе и искомое решение во всех его вариантах” [10].

В данном разделе рассматриваются следующие методы: слепой поиск, метод случайных направлений, метод поиска с “наказанием случайностью”, блуждающий поиск, которые отличаются эффективностью и возможностями.

В целом случайные методы поиска предпочтительнее регулярных в задачах высокой размерности  $n \geq 10$  и вдали от оптимума. Поэтому здесь они рассматриваются сравнительно кратко, преимущественно в ознакомительном плане.

Методы этой группы позволяют в среднем быстрее выходить в район оптимума. Эффективны рассматриваемые методы и при поиске глобального оптимума.

Как и в предыдущих случаях, на рис. 19 приводятся лишь по одной из возможных траекторий поиска каждым из ниже рассматриваемых методов. Кроме того, случайные методы имеют ту особенность, что даже при одних и тех же неформально задаваемых параметрах они дадут различные траектории поиска. Здесь, так же как и в предыдущих случаях, приведенные траектории начинаются из различных начальных условий, с тем чтобы не загромождать построения.



Рис.19. Иллюстрация траекторий поиска оптимума функции методами случайного поиска: 1 — область оптимума; 2 — траектория метода случайных направлений; 3 — траектория метода с наказанием случайностью; 4 — траектория метода блуждающего поиска; 5 — начальные точки поиска

## ОСНОВНЫЕ МЕТОДЫ

### МЕТОД СЛЕПОГО ПОИСКА

Идея метода очень проста и наглядна. Случайным образом в допустимой области берется точка, и сравнивается значение критерия в ней с текущим наилучшим. Если новая случайно взятая точка хуже хранящейся в качестве текущей лучшей, то берут другую точку. Если же нашли точку, в которой критерий лучше, то ее запоминают в качестве текущей лучшей. Гарантируется, что при неограниченном возрастании числа попыток мы будем приближаться к глобальному оптимуму, т.е. найденное текущее наилучшее значение будет сколь угодно близко к точному решению.

На практике поиск прекращают, когда число неуспешных попыток превышает наперед заданное число  $Nz$ .

Данный метод можно применять для поиска начального приближения, задав сравнительно небольшое число попыток. Метод прост в алгоритмическом плане и не требует примера с конкретными значениями.

### МЕТОД СЛУЧАЙНЫХ НАПРАВЛЕНИЙ

Из текущей (или заданной начальной) точки делается шаг в случайном направлении  $x^{i+1} = x^i + h\alpha^i$ , где  $\alpha$  — случайный вектор с модулем, равным единице (случайно только его направление);  $h$  — коэффициент пропорциональности шага. Если  $R(x^{i+1}) < R(x^i)$  (при поиске минимума критерия оптимальности), то новая точка принимается за текущую и из нее делаются шаги в надежде найти лучшую точку. Если  $R(x^{i+1}) \geq R(x^i)$ , то делают новую попытку, т.е. новый шаг  $h\alpha^i$ .

Поиск заканчивают, когда за заданное число попыток  $Nz$  не удается найти точку с лучшим значением критерия оптимальности, чем имеющаяся текущая.

Существуют модификации метода, в одной из которых после серии неудачных попыток  $Nz$  уменьшается коэффициент  $h$ , что

позволяет “уточнить” положение оптимума. В этом случае условием окончания является малость значения шага (т.е.  $h \leq h_{\min}$ ).

Существует также модификация метода с обратным шагом. Отличительной ее особенностью является то, что при неудачном шаге  $h\alpha^i$  из точки  $x^i$  сразу производится шаг в обратном направлении  $-h\alpha^i$ . При достаточном удалении от оптимума такая стратегия поиска может оказаться весьма эффективной. Если и обратный шаг оказался неудачным, то можно сделать новый шаг из текущей точки или перейти к поиску с уменьшенным шагом. В последнем случае существует опасность замедления поиска вдали от оптимума, особенно в овраге.

Одна из возможных траекторий поиска минимума двумерной функции методом случайных направлений приведена на рис. 19.

### Пример.

1. Требуется найти минимум функции  $R(x_1, x_2) = Ax_1^3 + Bx_2^2 - Cx_1 - Dx_2$ , где  $A = 1, B = 2, C = 3, D = 4$ .

2. Интервал поиска:  $x_{1\text{нач}} = -2, x_{1\text{кон}} = 2, x_{2\text{нач}} = -2, x_{2\text{кон}} = 2$ .

3. Начальная точка:  $x_{10} = -0,5, x_{20} = -1$ .

4. Параметры поиска: коэффициент шага  $h = 1$ , число попыток в каждой точке  $Nz = 10$ .

Результаты вычислений представлены в табл. 28.

Таблица 28

№_итер	$x_1$	$x_2$	$R$ наилучшее	Попытка
1	-0,4282	-0,8868	7,3722	неудачная
2	-0,3375	-0,9057	7,3722	неудачная
3	-0,3375	-0,7358	7,3722	неудачная
4	-0,2469	-0,7358	7,3722	неудачная
5	-0,2166	-0,5660	7,3722	неудачная
6	-0,1965	-0,3962	7,3722	неудачная
7	-0,1159	-0,3019	7,3722	неудачная
8	-0,1864	-0,1887	1,7359	удачная
9	-0,2771	-0,1132	1,7359	неудачная
10	-0,1864	-0,1509	1,2884	удачная
11	-0,0957	-0,1887	1,2884	неудачная

№ попытк	$x_1$	$x_2$	$R$ наилучшее	Попытка
12	-0,0353	-0,0566	1,2884	неудачная
13	-0,0856	0,0943	1,2884	неудачная
14	-0,0151	0,2075	1,2884	неудачная
15	0,0756	0,2264	-0,6987	удачная
...	...	...	...	...
28	0,7607	1,0189	-3,6836	удачная
29	0,8514	0,9623	-3,8412	удачная
30	0,9421	0,9057	-3,8412	неудачная
31	0,9824	1,0755	-3,9723	удачная

Последнюю точку можно считать решением, так как за заданное число следующих попыток (10), которые для краткости не приведены, не удалось найти лучшую точку. Возможно, увеличив число таких попыток, можно найти лучшее решение.

### **МЕТОД ПОИСКА С "НАКАЗАНИЕМ СЛУЧАЙНОСТЬЮ"**

Метод является аналогом метода наискорейшего спуска, только направление локального поиска не градиентное, а случайное. Как и в предыдущем методе, из текущей точки делают случайные шаги до тех пор, пока не будет найдена точка с лучшим значением критерия оптимальности. Затем в этом направлении регулярным методом одномерного поиска ищут оптимум. В точке оптимума по направлению опять случайным образом ищут новое направление и т.д.

Условием окончания обычно является невозможность получения лучшей точки из текущей за предварительно заданное число попыток  $N_z$ .

Одна из возможных траекторий поиска минимума двумерной функции методом поиска с "наказанием случайностью" приведена на рис. 18.

#### **Пример.**

Для сравнения рассмотрим пример, аналогичный предыдущему.

Результаты вычислений представлены в табл. 29, где  $N_{\text{поп}}$  — число неудачных попыток в данной точке найти лучшее направление поиска.

Таблица 29

№ <sub>итер</sub>	$x_1$	$x_2$	$R$	$N_{\text{поп}}$
0	-0,4987	-1,0000	7,3722	1
1	-0,3779	1,0365	6,0485	3
2	-0,0641	0,5329	-1,0952	6
3	0,0814	0,3625	-1,4190	5
4	0,3661	1,2154	-1,9197	0
5	0,8722	1,4260	-3,1468	2
6	0,8983	0,9867	-3,8288	6
7	0,9988	0,9658	-3,9976	8
8	0,9988	0,9658	-3,9977	10

Последнюю точку можно считать решением, так как за заданное число попыток (10) не удалось найти лучшую точку. Возможно, увеличив число таких попыток, можно найти более хорошее решение.

### МЕТОД С "БЛУЖДАЮЩИМ" ПОИСКОМ

Данный метод является статистическим расширением градиентного метода и реализуется в соответствии с алгоритмом

$$x^{i+1} = x^i - h \text{grad } R(x^i) + h_1 \alpha^i = x^i - \Delta x^i,$$

где  $\alpha$  — случайный вектор с единичным модулем,  $h$  и  $h_1$  — коэффициенты, характеризующие вклад случайной составляющей ( $h_1$ ) и регулярной составляющей ( $h$ ) в величину шага.

Чаще в формуле для  $x^{i+1}$  используется не градиент  $R(x)$ , а соответствующие направляющие косинусы градиента (см. метод градиента в теме "Многомерная градиентная оптимизация"), что позволяет выдерживать заданное соотношение между регулярной и случайной составляющими шага.

Теоретически доказывается, что данный алгоритм наиболее вероятно приведет к глобальному экстремуму. В алгоритме могут использоваться алгоритмы коррекции шага  $h$ , свойственные градиентному методу, который включается после  $N_z$  неудачных попыток. Условием окончания является малость значения шага (т.е.  $h \leq h_{\min}$ ).

Стратегия поиска может предусматривать не постоянное, а периодическое добавление случайного вектора к градиентному шагу. Частота случайных “скачков” должна уменьшаться по мере приближения к оптимуму и увеличиваться вдали от него. Для этого существуют специальные алгоритмы самообучения, например:

$$N_s = N_s(M, R(x)),$$

где  $N_s$  — число шагов регулярным градиентным методом без случайной составляющей, т.е. период добавления случайной составляющей;

$M$  — заданное целое число (рекомендуется  $2 \leq M \leq 10$ , при этом в процессе поиска  $N_s$  будет изменяться в диапазоне  $M/2 \dots 10M$ ).

Обратно пропорционально частоте “скачков” меняется и доля случайной составляющей в шаге, т.е.  $h_1 = h_1 / N_s$ . Условием окончания поиска будет, как и в регулярном градиентном методе, близость градиента к нулю.

Одна из возможных траекторий поиска минимума двумерной функции методом блуждающего поиска приведена на рис. 19.

### *Пример.*

Для сравнения рассмотрим пример, полностью аналогичный примеру в предыдущем разделе. Результаты вычислений сведены в табл. 30.

Таблица 30

№ <sub>итер</sub>	$x_1$	$x_2$	$R$	$N$
1	-0,2275	-0,0375	0,8236	1
2	0,5266	0,6192	-3,1438	1
3	1,3784	1,1431	-3,4753	3
4	0,4002	0,9357	-3,1281	2
5	1,3950	1,0371	-3,4676	1

№ итер	$x_1$	$x_2$	$R$	$N$
6	0,3964	0,9849	-3,1263	4
7	1,3961	1,0088	-3,4671	1
8	0,3961	0,9965	-3,1262	1
9	1,3961	1,0021	-3,4671	1
10	0,3961	0,9992	-3,1262	3
11	1,3961	1,0005	-3,4671	2
12	0,3961	0,9998	-3,1262	1

Нетрудно заметить, что при заданных параметрах поиск “зациклился” в окрестности оптимума. Для повышения эффективности поиска следует попытаться выбрать другие параметры поиска, но формальных рекомендаций по их выбору нет.

### КОНТРОЛЬНЫЕ ВОПРОСЫ

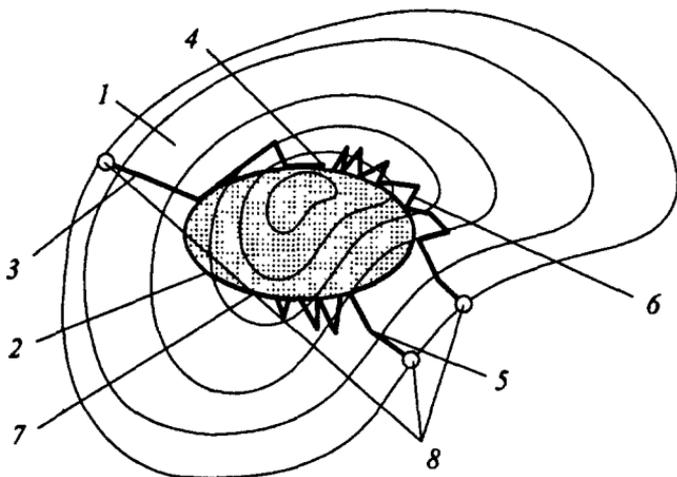
1. В чем заключается основное преимущество случайных методов поиска?
2. В чем достоинство метода слепого поиска?
3. Назовите недостаток слепого метода поиска.
4. Может ли в методе “блуждающего” поиска начальная точка выбираться случайным образом?
5. Может ли в методе поиска с “наказанием случайностью” начальная точка задаваться исследователем не случайно, а из своих соображений?
6. Может ли в методе слепого поиска начальная точка выбираться не случайно, а по желанию исследователя?
7. Сравните эффективность метода поиска с “наказанием случайностью” с методом наискорейшего спуска.
8. Сравните эффективность метода случайных направлений с градиентным методом в случае овражной  $R(x)$ .
9. Чем характеризуется случайный вектор, который используется в методе случайных направлений?
10. Что нужно сделать, чтобы получить более высокую точность решения задачи оптимизации в методе поиска с “наказанием случайностью”?

# МНОГОМЕРНАЯ УСЛОВНАЯ ОПТИМИЗАЦИЯ

## КОНЦЕПЦИЯ МЕТОДОВ

В данном разделе рассматриваются численные методы построения улучшающих последовательностей при наличии ограничений типа равенств (связей) и типа неравенств (ограничений) (рис. 20). Сюда не входят методы, использующие условия оптимальности. Во всех методах строится в допустимой области последовательность точек, в которых значения критерия улучшаются. Поиск осуществляется градиентным методом. При этом предполагается, что учащийся знаком с особенностями и условиями работы этого метода по крайней мере в объеме соответствующего раздела настоящего пособия.

Допустимая область может формироваться автономными ограничениями  $x_{i \min} \leq x_i \leq x_{i \max}$ , связями  $f_j(x_1, x_2, \dots, x_n) = 0$  ( $j = 1, \dots, m$ ) и ограничениями  $F_j(x_1, x_2, \dots, x_n) \geq 0$ , для  $j = 1, \dots, p$ .



**Рис. 20.** Иллюстрация траекторий поиска минимума функции при наличии ограничений типа неравенств: 1 — допустимая область; 2 — запрещенная область; 3 — траектория метода проектирования градиента; 4 — оптимум 1; 5 — траектория метода штрафов; 6 — траектория прямого метода с возвратом; 7 — оптимум 2; 8 — начальные точки поиска

Функции, задающие ограничения, могут формировать допустимую область с различными свойствами: монотонными, колебательными, с большой и малой кривизной и т.д., что оказывает влияние на эффективность методов поиска.

## ОСНОВНЫЕ МЕТОДЫ

### МЕТОД ШТРАФОВ

Метод штрафов позволяет перейти от исходной задачи с ограничениями к другой задаче без ограничений путем формирования новой структуры критерия оптимальности. Оптимальное решение новой задачи совпадает с решением исходной задачи и принадлежит исходной допустимой области.

Пусть имеем задачу

$$\begin{aligned} R(x) \rightarrow \min, & \quad f_i(x) = 0, & \quad i = 1, \dots, m, & \quad m < n, \\ x [1:n], & \quad F_j(x) \geq 0, & \quad j = 1, \dots, p. \end{aligned}$$

Формируется новый критерий, например с квадратичным “штрафом” для ограничений типа равенств:

$$R_{p1} = R(x) + \alpha \left( \sum f_i^2(x) + \sum F_{j-}(x) \right) = R(x) + H_1,$$

с модульным “штрафом”:

$$R_{p2} = R(x) + \alpha \left( \sum |f_i(x)| + \sum F_{j-}(x) \right) = R(x) + H_2,$$

с комбинированным “штрафом”:

$$R_{p3} = R(x) + \alpha \left( \sum f_i^2(x) + \sum |f_i(x)| + \sum F_{j-}(x) \right) = R(x) + H_3$$

(индексы суммирования для простоты опущены), где  $\alpha$  — коэффициент “штрафа”. Он характеризует вклад в общий критерий оптимальности.  $R_p$  — новый критерий,  $F_-(x)$  — срез функции  $F(x)$ ; он определяется по следующему правилу:

$$F_-(x) = \begin{cases} 0, & \text{если } F(x) > 0, \\ -F(x), & \text{если } F(x) < 0. \end{cases}$$

Отклонения от ограничений увеличивают критерий  $R_p$ , т.е. как бы накладывают “штраф”  $H$  на  $R_p$ . Уменьшение  $R_p$  происходит как за счет уменьшения  $R(x)$ , так и за счет уменьшения величины нарушения ограничения.

В точке минимума ограничения будут справедливы, т.е. равны нулю, и поэтому  $R^* = R_p^*$ . Однако это соотношение будет справедливо только при коэффициенте “штрафа”, стремящемся к бесконечности.

Увеличение  $\alpha$ , с одной стороны, не допускает больших отклонений в “запретные” стороны от ограничений (это хорошо), а с другой — увеличивает “овражность”  $R_p$  (овраг размерности  $m$  имеет место вдоль ограничений), что резко усложняет поиск оптимума  $R_p$  (это плохо).

Поэтому целесообразно решать задачу сначала с маленьким значением  $\alpha$ , потом его увеличивать. Это будет способствовать более быстрому выходу в область условного оптимума и точному его определению.

Существуют и другие структуры  $R_p$ , в частности учитывающие сравнительную важность отдельных ограничений. При этом каждое ограничение может иметь свой коэффициент штрафа.

Одна из возможных траекторий поиска минимума двумерной функции методом штрафов приведена на рис. 20.

### Пример.

Требуется найти минимум функции  $R(x_1, x_2) = Ax_1^3 + Bx_2^2 - Cx_1 - Dx_2$  при наличии ограничения типа равенства.

1. Коэффициенты выражения  $R(x_1, x_2)$ :  $A=1, B=1, C=1, D=1$ .

2. Интервал вычислений (автономные ограничения):  $x_{1\text{нач}} = -2, x_{1\text{кон}} = 2, x_{2\text{нач}} = -2, x_{2\text{кон}} = 2$ .

3. Начальная точка:  $x_{10} = -0,1864, x_{20} = 0,4340$ .

4. Прочие данные:  $h = 0,1$ ; погрешность = 0,05; штраф = 0,5; вид штрафа: модульный.

5. Ограничения типа равенства  $f(x_1, x_2) = A_{\text{огр}}x_1 + B_{\text{огр}}x_2^2 - 1 = 0$ .

Коэффициенты ограничения:  $A_{\text{огр}} = 5,00; B_{\text{огр}} = 9,30$ .

В соответствии с методом сформируем новый критерий, минимум которого будем искать градиентным методом  $R_p = R + \alpha f(x)$ . Величину  $\alpha$  выбираем равной 0,5.

Результаты вычислений приведены в табл. 31 без подробных промежуточных результатов. В таблице приведены для иллюстрации только первые 10 шагов, из которых видно, что поиск идет довольно медленно и отклонения постоянно колеблются, постепенно уменьшаясь. Характер поиска соответствует овражной функции  $R_p$ .

При решении практических задач можно сначала задавать меньшее значение  $\alpha$ , при этом поиск быстрее выйдет в район оптимума, а затем увеличивать коэффициент штрафа, что позволит точнее найти решение.

Таблица 31

№_итер	$x_1$	$x_2$	$R$	Отклонение (штраф)
0	-0,1864	0,4340	-0,0657	
1	-0,1015	0,5382	-0,1481	5,3762
2	-0,1398	0,4111	-0,1051	5,3069
3	-0,0537	0,5112	-0,1963	5,2776
4	-0,0915	0,3918	-0,1476	5,0088
5	-0,0046	0,4883	-0,2453	5,1936
6	-0,0421	0,3753	-0,1924	4,7599
7	-0,0797	0,2943	-0,1285	3,5737
8	0,0073	0,3730	-0,2412	4,6935
9	-0,0302	0,2926	-0,1768	3,5477
10	0,0572	0,3710	-0,2904	4,6975
...	...	...	...	...

### МЕТОД ПРЯМОГО ПОИСКА С ВОЗВРАТОМ

В этом методе условие строгого соблюдения ограничений  $f(x) = 0$  заменяют менее строгим, например:  $|f(x)| \leq \epsilon$ , где  $\epsilon$  — допустимое нарушение ограничения (только в процессе поиска).

В этом случае внутри допустимого “коридора”, который обрабатывается по обе стороны от ограничения типа равенств и в допустимой области ограничения типа неравенств, оптимум  $R(x)$  ищется любым методом без учета ограничений (например, градиентным

методом). Как только в процессе поиска достигается граница “рассматриваемого” ограничения  $|f(x)| = \epsilon$ , то происходит возврат на ограничение, причем это осуществляется по нормали к функции ограничения  $f(x)$  или  $F(x)$ .

При достижении границы ограничения типа неравенств  $F(x) = 0$  или автономных ограничений осуществляется “отскок” от границы в сторону допустимой области. Величина отскока может задаваться. Она влияет на скорость поиска и на возможность нахождения глобального оптимума в случае невыпуклой допустимой области (даже в случае одноэкстремальной функции критерия оптимальности задача с ограничением может иметь несколько локальных оптимумов).

Если в процессе поиска нарушено сразу несколько ограничений, то возврат осуществляется по сумме градиентов всех нарушенных ограничений.

Недостатком метода является сравнительно небольшая скорость поиска при движении вдоль ограничений. Процесс поиска вблизи оптимума замедляется.

Условием окончания поиска при наличии ограничений типа равенств обычно считается достижение заданной близости двух соседних точек возврата на ограничение. При наличии ограничений типа равенств условие остановки то же самое, что и в предыдущем случае, или малость градиента  $R(x)$ , если оптимум лежит в допустимой области.

Одна из возможных траекторий поиска минимума двумерной функции методом прямого поиска с возвратом приведена на рис. 20.

### *Пример.*

Для сравнения рассмотрим пример, полностью аналогичный примеру на с. 160; границу допустимого временного нарушения ограничения примем за 1.

Узловые результаты вычислений представим в виде таблицы, опуская большинство промежуточных результатов, так как они повторяют предыдущие методы. В табл. 32 “поиск ...” означает движение по градиенту  $R(x_1, x_2)$  до границы допустимого “коридора”, а “возврат ...” — возврат на ограничение.

Таблица 3 2

№ <sub>итер</sub>	$x_1$	$x_2$	Операция	$R$	Отклонение
0	-0,1864	0,4340	начальная точка	-0,0657	10,0000
1	0,0151	0,4528	поиск по $\text{grad } R(x_1, x_2)$ в доп. области	-0,2629	1,0037
2	-0,0359	0,3668	возврат на ограничение	-0,1964	0,1648
3	0,1058	0,3962	поиск ...	-0,3438	0,9815
4	0,0497	0,3135	возврат ...	-0,2647	0,1009
5	0,1763	0,3396	поиск ...	-0,3951	0,9573
6	0,0832	0,2220	возврат ...	-0,2554	0,0974
7	0,2267	0,2830	поиск ...	-0,4180	0,9435
8	0,1234	0,1743	возврат ...	-0,2654	0,0624
9	0,2670	0,2453	поиск ...	-0,4331	0,9327
10	0,1562	0,1442	возврат ...	-0,2758	0,0445

Две последние точки возврата на ограничение (8 и 10) отличаются друг от друга меньше, чем заданная погрешность:

$$(0,1234 - 0,1562)^2 + (0,1743 - 0,1442)^2 = 0,00198185$$

— это квадрат отклонения; отклонение примерно равно  $0,045 < 0,05$  (заданная погрешность). Поэтому можно прекратить поиск и последнюю точку считать решением.

Недостатком метода является трудоемкость вычисления нормали к границе ограничения. В примере эта часть вычислений не приведена ввиду большого объема; способ построения нормали можно найти в любом учебнике по высшей математике.

### МЕТОД ПРОЕКТИРОВАНИЯ ГРАДИЕНТА

В данном методе, так же как и в методе прямого поиска с возвратом, ограничения типа равенств заменяются менее строгим условием попадания в заданный “коридор”, т.е. вместо  $f(x) = 0$  учитывают  $|f(x)| \leq \varepsilon$  (только в процессе поиска). Ограничение типа неравенств тоже “разрешается нарушать”, т.е.  $F(x) \geq -\varepsilon$ .

При наличии ограничений типа равенств движение внутри “коридора” осуществляют в направлении касательной к поверхности ограничений в текущей точке. Для определения конкретного направления поиска в плоскости (в двухмерном случае касательная гиперплоскость вырождается в прямую линию) на нее проектируют градиент критерия оптимальности. В направлении проекции градиента осуществляется движение до тех пор, пока не будет достигнута граница допустимого “коридора” или найден экстремум по направлению. Из достигнутой точки осуществляется возврат на ограничение по нормали к нему. Далее все повторяется опять.

В области условного оптимума, как правило, возникает заикливание, т.е. смена движения вдоль ограничения на противоположное. В этом случае можно найти решение усреднением одних из координат поиска и вычислением других из уравнений ограничений или уменьшением величины “коридора”. Многократное уменьшение “коридора” позволяет точнее получить решение задачи; условием окончания поиска является близость двух точек возврата на ограничение.

При наличии ограничений типа неравенств в допустимой области поиск идет без учета ограничений (например, градиентным методом), а при достижении границы  $F(x) = 0$  — по касательной к границе ограничений (или суммы достигнутых ограничений) внутрь “запретной” области на величину  $\varepsilon$ . Затем осуществляется возврат на поверхность ограничения (по нормали к границе).

При определенных условиях может произойти “отрыв” траектории поиска от границы  $F(x) = 0$ , и дальнейшее движение может в допустимой области осуществляться любым методом поиска без ограничений. При достижении автономных ограничений дальнейшее движение осуществляется вдоль ограничения.

Метод довольно эффективен при движении вдоль ограничений небольшой кривизны. В этом случае редко осуществляется возврат на ограничение.

К недостаткам метода относится большой объем вычислений, связанный с вычислением касательной гиперплоскости, с проектированием градиента на нее.

Одна из возможных траекторий поиска минимума двумерной функции методом проектирования градиента приведена на рис. 20.

**Пример.**

Для сравнения рассмотрим пример, полностью аналогичный примеру в предыдущем разделе. Узловые результаты вычислений представим в виде таблицы, опуская большинство промежуточных результатов, так как они повторяют предыдущие методы. В табл. 33 “поиск ...” означает движение по касательной к ограничению до границы допустимого “коридора”, а “возврат ...” — возврат на ограничение.

Таблица 33

№ <sub>итер</sub>	$x_1$	$x_2$	Операция	$R$	Отклонение
0	-0,1864	0,4340	начальная точка	-0,0657	
1	0,1963	0,0180	поиск ...	-0,2065	0,5652
2	0,2376	0,2040	возврат ...	-0,3866	0,4823
3	0,2039	0,2410	поиск ...	-0,3784	0,2231
4	0,1636	0,2715	возврат ...	-0,3570	0,1002
5	0,0757	0,3234	поиск ...	-0,2941	0,1524
6	0,0761	0,3239	возврат ...	-0,2946	0,1020
7	0,1463	0,1293	поиск ...	-0,2557	0,2066
8	0,0392	0,2965	возврат ...	-0,2478	0,0459

Две последние точки возврата на ограничение (6 и 8) отличаются друг от друга меньше, чем заданная погрешность:

$$(0,0761 - 0,0392)^2 + (0,3234 - 0,2965)^2 = 0,00211$$

— это квадрат отклонения; отклонение примерно равно 0,046 < 0,05 (заданная погрешность). Поэтому можно прекратить поиск и последнюю точку считать решением.

Недостатком метода является трудоемкость вычисления касательной к границе ограничения. В примере эта часть вычислений не приведена ввиду большого объема; способ построения касательной можно найти в любом учебнике по высшей математике.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

### *Метод штрафов*

1. На что накладывается “штраф”?
2. Зачем увеличивают коэффициент “штрафа”?
3. Зачем можно уменьшать коэффициент “штрафа”?
4. Где выбирается начальная точка поиска?
5. Что надо делать, чтобы уменьшить “овражность” расширенного критерия оптимальности?
6. Разрешается ли нарушать ограничения в процессе поиска?
7. Каким методом предпочтительнее искать оптимум расширенного критерия оптимальности?
8. От чего зависит точность нахождения оптимума исходной задачи?
9. Как ускорить процесс отыскания оптимума исходной задачи?
10. Как формируется расширенный критерий оптимальности в случае поиска  $\max R(x)$ ?

### *Метод прямого поиска с возвратом*

1. Как формируется вспомогательный критерий оптимальности решаемой задачи  $R_1(x)$  при использовании рассматриваемого метода?
2. В каком направлении осуществляется движение после достижения ограничения типа равенства  $f(x) = 0$ ?
3. Когда происходит “возврат” в допустимую область при исходной задаче  $R(x) \rightarrow \max$  с ограничениями  $F(x) \geq 0$ ?
4. Какое влияние оказывает ширина допустимого “коридора” на эффективность поиска?
5. Какое влияние оказывает ширина допустимого “коридора” на точность нахождения решения?
6. Зачем вводится “коридор” в задаче оптимизации типа  $R(x) \rightarrow \max, F(x) \geq 0$ ?
7. Может ли процесс поиска решения остановиться не на ограничении  $f(x) = 0$ ?

8. Условие окончания поиска в задаче с ограничениями типа  $F(x) \geq 0$ .
9. Как определяется направление возврата в допустимую область в случае одновременного нарушения двух и более ограничений типа неравенств?
10. В каких случаях эффективность поиска будет низкой?

### ***Метод проектирования градиента***

1. Какой градиент проектируется в методе?
2. Куда проектируется градиент  $R(x)$ ?
3. Куда проектируется градиент  $f(x)$ ?
4. Какое выражение задает направление касательной к  $f(x_1, x_2) = 0$ ?
5. Какое выражение задает направление нормали к функции ограничения  $f(x)$  в точке  $(x_1^i, x_2^i)$ , лежащей на ограничении?
6. Зачем вводится допустимый “коридор” в запретной области для ограничения типа неравенств  $F(x) \geq 0$ ?
7. В каких случаях метод наиболее эффективен?
8. В каком направлении необходимо искать  $\min R(x)$  внутри допустимого “коридора”, если направление касательной к ограничению и градиента  $R(x)$  не совпадают?
9. В трехмерной задаче  $R(x_1, x_2, x_3) \rightarrow \min, f(x_1, x_2, x_3) = 0$ , имея только касательную к ограничению в данной точке, можно ли указать направление поиска?
10. Как находится направление поиска  $\min R(x)$  в каждой текущей точке поиска внутри допустимой области?

---

# ПРОГРАММНЫЕ СРЕДСТВА ДЛЯ ВЫЧИСЛИТЕЛЬНЫХ РАБОТ

---

Существует немало различных программных систем для проведения математических расчетов. Это пособие не претендует на рассмотрение всех имеющихся систем. Упомянем лишь наиболее распространенные, работающие как под DOS, так и под WINDOWS. Это накладывает отпечаток и на используемые компьютеры.

Для работы на компьютерах невысокой производительности, которые пока еще составляют значительную долю в учебных заведениях, можно использовать пакет Eureka.

Для обучения может представлять интерес “Электронный учебник по численным методам”, который работает под DOS и ориентирован прежде всего на первичное освоение методов, их исследование, контроль усвоения, а не на решение всех встречающихся в практике задач, хотя с его помощью тоже можно решать сравнительно широкий круг практических задач.

Для проведения вычислений на современных компьютерах следует рекомендовать, помимо Excel, широко распространенные пакеты MathCad, MathLab, Mathematica, Maple, Derive, Statistica, общая характеристика и рекомендуемая литература по которым приведены ниже.

## EUREKA

---

Интегрированная программная система Eureka предназначена для решения систем линейных и нелинейных уравнений и неравенств. Кроме того, Eureka обладает следующими возможностями:

- вычислять значения производных и определенных интегралов;
- осуществлять поиск максимума и минимума функций;
- работать с экспоненциальной, логарифмической, тригонометрическими функциями, а также с полиномами.

Используя систему Eureka, можно выводить на экран и печатать графики или таблицы значений функций; пересчитывать единицы измерений; создавать отчет о проведенной работе.

Пользователь системы избавлен от необходимости программирования процесса решения задачи. Eureka требует лишь описания условия задачи в форме, максимально приближенной к привычной алгебраической записи уравнений и неравенств.

## **ЭЛЕКТРОННЫЙ УЧЕБНИК ПО ЧИСЛЕННЫМ МЕТОДАМ**

---

Электронный учебник, выполненный в двух частях, представляет собой программные комплексы, рассчитанные на IBM-совместимые компьютеры.

Работа с учебником рассчитана на слабо подготовленного пользователя и не требует специальных навыков работы на ЭВМ. Он содержит описание порядка работы с программой, доступное в любой момент работы, и развитую систему информационной помощи.

Первая часть учебника по численным методам содержит следующие разделы:

1. Интерполяция.
2. Аппроксимация.
3. Решение нелинейных уравнений.
4. Вычисление интегралов.
5. Основы решения дифференциальных уравнений в обычных производных.
6. Основы решения дифференциальных уравнений в частных производных.
7. Методические указания по обучению.

Вторая часть учебника по численным методам оптимизации содержит следующие разделы:

1. Одномерная оптимизация.
2. Многомерная безусловная градиентная оптимизация.
3. Многомерная безусловная безградиентная оптимизация.
4. Многомерная случайная безусловная оптимизация.
5. Многомерная условная оптимизация.
6. Методические указания по обучению.

Построение каждого раздела базируется на необходимом сочетании знакомства с теорией метода, активного практического исследования особенностей работы методов в различных ситуациях, самоконтроля усвоения материала и алгоритмов реализации методов на ЭВМ. Использование всего арсенала приемов воздействия на сознание обучаемого позволяет активно на различных уровнях подсознания усваивать сложные математические методы и приобретать навыки их использования. Анализ характера процесса решения в различных ситуациях позволяет исследователю (т.е. обучающемуся в активной форме) распознавать особенности методов при решении конкретных задач и соответствующим образом корректировать параметры методов или сами методы для повышения эффективности их использования в будущей работе.

Одним из важных аспектов применения численных методов является структура функций (в нелинейных уравнениях, интегралах и т.д.) и критерия оптимальности (в задачах оптимизации). В связи с этим в учебнике значительное внимание уделяется формированию (а не только выбору) заданных особенностей функций и критериев оптимальности. Обучаемый может создавать практически любые структуры функций как из предлагаемого набора с несколькими степенями свободы изменения их свойств, так и вводить свои. Также широкие возможности предоставляются и при формировании ограничений и связей в задачах условной оптимизации.

Самоконтроль усвоения материала может быть по желанию преподавателя организован в различных режимах. Основная цель, которая при этом преследуется, — это не жесткая оценка, характеризующая “ответный показатель” меры усвоения, а обращение внимания обучаемого на многие особенности методов, которые или рассмотрены в теоретической части, или затрагивались при практической работе.

При использовании электронного учебника в аудиторных занятиях преподаватель может легко устанавливать свои задания, т.е. учебник служит не “программой однократного действия”, а инструментом, с помощью которого можно, решая широкий круг задач, существенно повысить эффективность обучения сложным современным математическим методам.

Электронный учебник является очень важной составной частью учебного пособия по вычислительным методам в математическом моделировании. Он позволяет с высокой степенью наглядности изучать соответствующие методы в режиме активной практики и исследования.

Требования к компьютеру для установки электронного учебника: IBM-совместимый компьютер с процессором 286 и выше, 1 Мбайт RAM, 2,5 Мбайта свободной памяти на винчестере на каждую из двух частей, дисплей не хуже EGA.

Авторские права на учебник зарегистрированы в РосАПО, свидетельство № 960181 от 20.05.96.

Электронный учебник полностью (обе части — две дискеты) или частично (одна часть по выбору — одна дискета) может быть выслан авторами наложенным платежом по персональным заявкам, которые можно направлять через издательство или непосредственно авторам по E-mail: [myvas@iubnt.yar.ru](mailto:myvas@iubnt.yar.ru), а также [myvas@mail.ru](mailto:myvas@mail.ru).

Стоимость каждой дискеты учебника примерно равна стоимости книги. Каждая дискета содержит возможность двух инсталляций программы на персональный компьютер.

Демонстрационная версия программы (усеченный вариант с ограниченными возможностями) может быть выслана по электронной почте бесплатно.

## MATHCAD

---

MathCad представляет собой интегрированную среду для выполнения, документирования и обмена результатами технических вычислений. Он позволяет пользователям вводить, редактировать и решать уравнения, визуализировать результаты, документировать их, а также обмениваться результатами анализа. MathCad служит средством вычислений, анализа и написания отчетов для профессионалов во всех областях науки и техники и студентов.

Записывая большинство математических формул в рабочем документе MathCad, как это делается на листе бумаги, вы можете тут же получить желаемый ответ. Если необходимо выполнить алгебраические преобразования или действия математического анализа над заданным выражением, то MathCad сделает это, снаб-

жая по вашему желанию выкладки соответствующими текстовыми комментариями. Если нужно найти числовые результаты, MathCad найдет их. В выкладках при поиске числового результата наряду с общеупотребительными математическими операциями и функциями может быть использовано большое количество встроенных функций численного анализа MathCad, таких, как функции отыскания собственных векторов матрицы, решения дифференциальных уравнений и генерации последовательности случайных чисел с заданным законом распределения.

Использование всех упомянутых возможностей MathCad является чисто рутинным и не требует никакого программирования. Вам не нужно ломать голову над способом решения задачи, MathCad это берет на себя, и вы можете всецело сосредоточиться на анализе результата. Например, чтобы найти значение интеграла, не нужно ни подыскивать подходящую квадратуру, ни отлаживать алгоритм, достаточно записать интеграл и нажать знак равенства.

MathCad допускает форматирование выводимых числовых результатов, использует при выводе единицы измерения (в том числе 95 встроенных) и автоматически изменяет числовое значение результата при изменении единицы его измерения. Для визуализации результатов MathCad строит 7 разновидностей двумерных и трехмерных графиков, каждая из которых предоставляет богатые возможности оформления и форматирования. На каждом двумерном графике может находиться одновременно до 16 различных кривых, имеющих по 6 атрибутов. MathCad может быть интегрирован с программным продуктом Axiom (в версии 6 поставляется отдельно), что позволяет пополнить список выводимых графиков еще 75 их типами.

Совмещая текст, графику и формулы, записанные в их привычном виде, документ MathCad выглядит, как страница учебника или научной статьи. При этом формулы являются “живыми” — стоит внести изменения в любую из них, как MathCad перерисует графики, выведет новые результаты и т.п. Можно анимировать любой график, записав его эволюцию при изменяющихся значениях параметра, а затем воспроизвести полученную мультипликацию со звуковым сопровождением.

Вычислительные возможности MathCad могут быть увеличены за счет расширения списка встроенных функций. Для этого

можно либо приобрести один из специализированных пакетов функций MathSoft, либо написать библиотеку таких функций самостоятельно, используя один из 32-разрядных компиляторов C/C++, причем в последнем случае самодельные функции ничем не будут отличаться от фирменных — они будут появляться в диалоговом окне MathCad, выдавать сообщения об ошибках и т.п.

MathCad 8 Professional — это существенный этап в эволюции MathCad. Он пользуется всеми преимуществами OLE2 для работы с другими приложениями, поддерживая технологии “drag and drop” и “in-place activation” для клиента и сервера. Встроенная версия Microsoft® Internet Explorer включает окно содержания, которое предоставляет доступ к ресурсам MathCad и HTML, расположенным в любом месте в Internet. Также разработан новый язык визуального программирования MathConnex для связи данных MathCad и других приложений. С появлением MathConnex пользователи получили полностью интегрированную среду для решения инженерных задач.

Для сложных задач анализа и создания отчетов используйте MathCad 8 Professional. Если вам не нужна вся мощь профессиональной редакции, а электронной таблицы и калькулятора все-таки недостаточно, попробуйте поработать с MathCad 8 Standard, отличным инструментом для повседневной работы.

## **MATHCAD EXPLORER**

---

22 июня 1998 г. производитель широкого класса математических и технических программ для профессионалов и студентов MathSoft Inc. анонсировал выпуск MathCad Explorer, самого совершенного инструментария для выполнения математических вычислений в Web. MathCad Explorer — это бесплатное приложение, доступное по адресу <http://www.mathsoft.com>, которое предоставляет студентам доступ ко всем тем функциям MathCad, которые используют миллионы профессионалов по всему миру. MathCad Explorer включает полный набор вычислительных функций, необходимых современному студенту, к тому же сопровождаемых обучающими средствами и возможностью виртуальных групповых занятий.

## Требования к компьютеру для установки MathCad

**MathCad 8 Professional:** персональный компьютер с процессором Pentium 90 или выше; Windows 95, 98, NT 4.0 или более поздние версии; 16 Мбайт RAM необходимо; 32 Мбайта рекомендуется; минимум 30 Мбайт на жестком диске; 80 Мбайт требуется для полной установки; дисковод CD; графическая карта и монитор VGA (рекомендуется Super VGA); мышь; доступ в Интернет для полноценной работы с дополнительными ресурсами.

**MathCad 7:** персональный компьютер с процессором 80486 или Pentium; Windows 95 или Windows NT 3.51 или более поздние версии; 16 Мбайт RAM необходимо; минимум 18 Мбайт на жестком диске; 55 Мбайт требуется для полной установки; дисковод CD; графическая карта и монитор VGA (рекомендуется Super VGA); мышь.

**MathCad 6:** персональный компьютер с процессором 80386, 80486 или Pentium; арифметический сопроцессор не обязателен, но его присутствие существенно увеличивает производительность; Windows 3.1 или более поздняя версия; Windows NT 3.5 или более поздняя версия, или Windows 95; не менее 8 Мбайт оперативной памяти; минимум 20 Мбайт на жестком диске для файлов MathCad и 3 Мбайта свободного пространства на том диске, на котором установлена операционная система Windows; видеомонитор и видеокарта, совместимые с Windows; мышь.

**Ахун 6:** персональный компьютер с процессором Pentium 90 или выше; Windows 95, 98, NT 4.0 или более поздние версии; 32 Мбайта RAM необходимо; математический сопроцессор; минимум 90 Мбайт на жестком диске, 145 Мбайт требуется для полной установки; дисковод CD; графическая карта и монитор VGA (рекомендуется Super VGA); мышь.

## MAPLE

Основной программный продукт, выпускаемый Университетом Ватерлоо, — пакет Maple V — называют системой символьных вычислений, или системой компьютерной алгебры. Он предназначен для выполнения самых разных математических вычислений, как аналитических, так и символьных. Его интерфейс интуитивно понятен, правила работы предельно просты, а

возможности внушительны. Он стал незаменимым средством в работе математиков и инженеров, студенты с его помощью легко справляются с труднейшими заданиями. При работе с пакетом возникает ощущение роста собственных математических знаний, легко решаются сложнейшие задачи.

Аналитические функции позволяют решать математические задачи, не назначая численные значения константам и приближения — переменным. Maple V понимает аналитические математические формулы и выдает ответ в виде таких же формул. Результат работы Maple V — точное аналитическое решение, дающее глубокое понимание сути решаемой задачи.

Maple V включает в себя богатую библиотеку аналитических функций для решения широкого класса общих и специализированных задач. Имеются процедуры для аналитического дифференцирования, интегрирования, решения уравнений, линейной алгебры, геометрии и многих других задач. Этот впечатляющий набор функций пригодится огромному числу пользователей: учителям и преподавателям вузов, школьникам и студентам, научным работникам и инженерам, многим другим людям.

Maple V используется во всем мире для учебы, подготовки к занятиям, разработки материалов для обучающих курсов по математическому анализу, алгебре, структурному проектированию, физике, химии, экономике и др. Первокурсникам и аспирантам пригодится Maple V как “подручное средство”, делающее математику более динамичной и полной смысла.

Требования к компьютеру для установки Maple: процессор Intel 486DX, Pentium или полностью с ними совместимые; или 486SX с сопроцессором; 32 Мбайта свободного дискового пространства; 8 Мбайт RAM; Microsoft Windows 3.1x, Windows NT 4.0 или Windows 95.

## DERIVE

---

Система Derive фирмы Soft Warehouse принадлежит к классу компьютерных систем для автоматизации математических вычислений и прежде всего — символьных (аналитических) преобразований. Система прекрасно справляется и с численными расче-

тами, сочетая их с использованием вполне современной графики, как двумерной, так и трехмерной.

В настоящее время в стране распространены и доступны версии Derive 3.11 под DOS и 4.02 под Windows. Система является многофункциональной, способной без внешних расширений эффективно решать самые разнообразные прикладные задачи, прежде всего задачи математического моделирования в науке, технике и экономике. Она имеет в своем инструментарии широкий спектр самых разнообразных методов, среди которых:

- вычисление алгебраических, тригонометрических, гиперболических, статистических и финансово-экономических функций, а также специальных математических функций;
- действия над числами, операции с действительными и комплексными числами, представление их в дробно-рациональной форме;
- символьные операции с многочленами, включая разложение их на простые множители и вычисление действительных и комплексных корней, дробно-рациональными функциями, функциями многих переменных;
- символьное и численное интегрирование и дифференцирование, вычисление пределов и сумм, нахождение разложений в ряды, тригонометрических и других функций;
- операции с векторами и матрицами, элементами которых могут быть числа или арифметические выражения;
- преобразования формул с использованием подстановок, разложение на множители и пр.;
- построение двумерных и трехмерных графиков в параметрической форме, в полярной и декартовой системах координат и т.п.

Универсальность и интегрированность системы Derive позволяет использовать ее для решения широкого круга математических и научно-технических задач.

Требования к компьютеру для установки системы: версия 3.11 может быть установлена на всех типах персональных компьютеров и требует скромных ресурсов: 1,2 Мбайта памяти на винчестере и минимальной оперативной памяти. Версия 4.0 для Windows требует 8 Мбайт оперативной памяти при использовании любой операционной системы (Windows 95/98, NT).

## MATLAB

---

MatLab — система, предназначенная для выполнения инженерных и научных расчетов и высококачественной визуализации получаемых результатов. Эта система применяется для математических расчетов, моделирования физических систем и управления техническими объектами.

Существуют версии системы MatLab для операционных сред Windows, UNIX и др. Обратим внимание на версию The Student Edition of MatLab, специально предназначенную для студентов. Эта версия включает ядро системы MatLab 5, а также 3 пакета прикладных программ Symbolic Mathematics Toolbox, Control System Toolbox и Signal Processing Toolbox. Они предназначены для выполнения математических, инженерных и научных расчетов. Студенческая версия системы продается по цене в 20 раз дешевле профессиональной версии. Ее ограничение связано только с размерами обрабатываемых массивов, которые не могут включать более 11 664 элементов.

Система работает с многомерными массивами, массивами записей и массивами ячеек, с матрицами, в том числе и разреженными, позволяет проводить анализ и обработку данных, включая аппроксимацию и интерполяцию, численное интегрирование, решение систем обыкновенных дифференциальных уравнений, вычисление минимумов и нулей функций, преобразование Фурье, свертку и фильтрацию и т.д. и т.п. Она может выполнять графические команды и функции.

## MATHEMATICA

---

Пакет Mathematica позволяет осуществлять широкий спектр символьных преобразований, включающих, наряду с другими, операции математического анализа: дифференцирование, интегрирование и интегральные преобразования, разложение в ряды, решение дифференциальных уравнений и т.п.

Одна из сильных сторон рассматриваемого программного продукта — развитая дву- и трехмерная графика, используемая для визуализации математических объектов.

По своей сущности Mathematica представляет собой язык программирования высокого уровня, позволяющий реализовать традиционный процедурный и функциональный стили программирования, а также стиль правил преобразований. Поскольку рассматриваемый программный продукт обеспечивает также применение разнообразных численных методов, то в совокупности символьные, графические и численные вычисления, выполняемые в одном сеансе использования Mathematica, превращает ее в удобный и мощный инструмент математических исследований.

## STATISTICA

---

Statistica 5.0 — интегрированная система комплексного статистического анализа и обработки данных в среде Windows — занимает устойчивое лидирующее положение на рынке статистического программного обеспечения. Одной из важных возможностей является обработка данных с точки зрения построения регрессионных моделей, прогнозирования поведения системы на основе этих моделей.

Система состоит из следующих основных частей:

- многофункциональной системы для работы с данными;
- мощной графической системы для визуализации данных и результатов статистического анализа;
- набора статистических модулей, в которых собраны группы логически связанных между собой статистических процедур;
- специального инструментария подготовки отчетов;
- встроенного языка Statistica Basic.

---

# МЕТОДИЧЕСКИЕ КОММЕНТАРИИ И ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

---

## МЕТОДИЧЕСКИЕ КОММЕНТАРИИ

---

Настоящее учебное пособие предназначено прежде всего для самостоятельной работы, которая не может дать весомых результатов только при теоретическом изучении книги. Очень важным элементом самообразования является активная практика, т.е. опыт решения различных вычислительных задач в широком диапазоне исходных данных, свойств функций и т.п. Обычно этот опыт приобретается годами при решении встречающихся в практической или учебной деятельности задач. Однако процесс его приобретения можно ускорить анализом решений, получаемых с помощью компьютеров, не обременяя себя излишними рутинными вычислениями. Это позволяет сделать, например, использование электронного учебника [5], где основное внимание уделено визуализации численных методов и активной практике. В электронном учебнике есть возможность практически в безграничных пределах варьировать особенности функций, начальные условия, алгоритмы и их параметры и т.д. Наблюдение в графической форме за процессом решения, анализ результатов позволяют глубже и быстрее усвоить изучаемый материал. Получить необходимый опыт можно и решая множество задач по соответствующей тематике на “типовом” программном обеспечении, где отсутствие визуализации не даст столь эффективного усвоения, но также существенно ускорит понимание теоретических основ и их практических особенностей.

Очень важное значение в образовании имеют методы самопроверки получаемых знаний, т.е. самотестирование. В каждом разделе пособия приведены контрольные вопросы, ответы на которые можно найти в этом разделе. Сама идея наличия ответов в книге может представляться спорной, так как чтение ответа не

стимулирует самостоятельные размышления над вопросом. Однако авторы считают, что знания нужны прежде всего обучаемо-му, поэтому полагаются на его добросовестность в использовании приведенных ответов лишь в качестве проверки своих мыслей. Ответы на большинство вопросов не лежат на поверхности, не очевидны только из кратко изложенных основ теории, а ясны после достаточно широкого применения методов на практике. В связи с этим авторы постарались ответы на все вопросы, несмотря на достаточно сжатую форму изложения, дать не краткие, а с дополнительными пояснениями. В этом случае, если обучаемый не захочет вникать в глубину вопроса, а лишь прочитает внимательно вопрос и соответствующий ответ и задумается хотя бы только над ним, то и это будет являться сравнительно эффективной формой обучения. Для некоторого затруднения процесса “добывания” ответа на поставленный вопрос все ответы собраны в конце книги, что может “вынудить” думать самостоятельно над вопросом хотя бы в течение поиска ответа.

## ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

---

### ИНТЕРПОЛЯЦИЯ

#### *Метод Лагранжа*

1. Полиномом  $n$ -й степени, так как именно он может быть проведен через заданные  $n + 1$  точки.
2. В принципе может, в методе не заложено ограничений на применение для экстраполяции.
3. На точность интерполяции влияют, как это видно из формулы погрешности: число узлов, свойства исходной функции (посредством производной  $(n + 1)$ -го порядка), расположение узлов в интервале, положение точки, в которой определяется погрешность.
4. Новые узлы добавлять можно, но интерполяционное значение функции необходимо рассчитывать заново, а не вносить поправку, обусловленную добавляемой точкой.
5. Узлы можно располагать произвольно.
6. Интерполяционная формула Лагранжа относится к полиномиальным функциям.

7. Дополнительная точка внутри интервала повысит точность интерполяции (если общее число точек не слишком велико).
8. Погрешность в узле всегда равна нулю, так как интерполяционная функция всегда проходит через все узлы.
9. С ростом числа узлов (на заданном интервале) погрешность сначала падает, что обусловлено лучшим совпадением исходной функции и интерполяционным полиномом более высокой степени, а затем может возрастать за счет высокой чувствительности к вычислительным погрешностям, особенно в коэффициентах при высоких степенях переменной  $x$ . Она может также возрастать и вследствие того, что полином высокой степени не сходится к интерполируемой функции.
10. Точность интерполяции в заданной точке можно повысить за счет некоторого повышения числа узлов и оптимального их размещения (см. метод Чебышева) на заданном интервале.

### *Метод Ньютона*

1. Может, в методе не заложено ограничений на применение для экстраполяции.
2. Нет, неравномерно располагать узлы интерполяции при использовании формул с конечными разностями нельзя, так как невозможно в этом случае рассчитать конечные разности; существует развитие методов Ньютона, позволяющее использовать неравномерно расположенные узлы, в этом случае используются разделенные, а не конечные разности.
3. Точность интерполяции в заданной точке можно повысить за счет некоторого увеличения числа узлов.
4. Можно получить конечную разность  $(n - 1)$ -го порядка по  $n$  точкам.
5. Конечная разность  $k$ -го порядка в  $i$ -й точке выражается следующим образом:

$$\Delta y_i^k = \Delta y_{i+1}^{k-1} - \Delta y_i^{k-1}.$$

6. Можно конечную разность любого порядка выразить через исходные значения функции, так как разность первого порядка непосредственно выражается через функции, разность второго порядка — через разности первого порядка, подставив выражения для которых в формулу для второй разности получим выражение второй разности через исходные функции, и т. д.
7. Первая интерполяционная формула базируется на конечных разностях, вычисленных в начальных точках, а вторая — на конечных разностях, вычисленных в конечных точках заданного интервала.

8. Для оценки погрешности интерполяции по имеющимся  $n$  точкам можно для интерполяции использовать меньшее число точек (например,  $n - 1$  или  $n - 2$ ), а остальные точки использовать для вычисления конечной разности высокого порядка, которая используется в формуле оценки погрешности интерполяции.
9. По трем заданным точкам можно получить интерполяционный полином второго порядка.
10. Интерполяционный полином степени  $n$  при наличии  $n+1$  точки единственный, так как он содержит  $n+1$  коэффициентов, определяемых однозначно с использованием заданных точек.

### *Метод сплайнов*

1. Сплайновая интерполяция позволяет получить наиболее гладкую интерполяционную функцию.
2. Кубическим сплайном называют кусочнополиномиальную функцию, когда каждый полином является полиномом третьей степени.
3. Каждый кубический полином проходит через две соседние точки.
4. Один кубический сплайн содержит четыре коэффициента, которые нужно определить.
5. Для нахождения коэффициентов сплайна используются следующие условия:
  - прохождение сплайна через начальные для него точки;
  - прохождение сплайна через конечные для него точки;
  - гладкость общей интерполяционной функции (равенство первых производных в начале следующего и в конце текущего участков);
  - гладкость первой производной (равенство вторых производных в начале следующего и в конце текущего участков).
6. Функция называется гладкой, если ее первая производная непрерывна.
7. Узлы при сплайновой интерполяции могут располагаться произвольно.
8. К недостаткам сплайновой интерполяции относится сравнительно большой объем вычисляемых коэффициентов (при  $n+1$  узле будет  $n$  участков и, следовательно,  $n$  сплайнов, для каждого из которых необходимо найти четыре коэффициента).
9. Производные можно найти в узлах специальными вычислительными алгоритмами, минуя непосредственное нахождение всех коэффициентов сплайна, причем с высокой эффективностью; вычисление производных является одной из основных прикладных областей применения сплайнов.

10. Дополнительные граничные условия необходимы потому, что используемых условий не хватает для определения всех коэффициентов сплайнов (не хватает как раз двух, поэтому дополняют расчетные соотношения двумя граничными условиями на концах интервала).

## АППРОКСИМАЦИЯ

### *Метод наименьших квадратов*

1. Вообще говоря, можно, если задать степень аппроксимирующего полинома равной номеру последней точки (если нумерация точек идет от нуля). Однако в этом случае аппроксимирующая функция превращается в интерполяционную.
2. Весовые коэффициенты в критерии близости исходной и аппроксимирующей функций показывают относительную важность “вклада” точки в общую сумму квадратов отклонений (в случае квадратичного критерия). Варьируя их, можно повысить точность аппроксимации в отдельных точках, как правило, в наиболее важных в конкретной задаче.
3. Нельзя все весовые коэффициенты для повышения точности увеличить одновременно в одинаковое число раз, так как при этом не изменится их отношение и не изменится относительная важность отдельных точек.
4. Нет, не всегда увеличение суммы квадратов отклонений соответствует худшей близости исходной и аппроксимирующей функций. Например, при введении отличных от единицы весовых коэффициентов (например, всех одинаковых) критерий увеличится за счет коэффициентов, а не за счет увеличения отклонений.
5. Можно найти параметры в принципе любой аппроксимирующей функции.
6. При использовании в качестве аппроксимирующей функции, в которую искомые параметры входят нелинейно, система нормальных уравнений получится также нелинейной, и ее не всегда возможно решить аналитически, поэтому приходится применять поисковые методы для нахождения минимума квадратичной меры близости.
7. Система нормальных уравнений получается линейной только в случае, когда при квадратичной мере близости параметры в аппроксимирующую функцию входят линейно (например, в случае полиномиальных функций).

8. В том случае, когда система уравнений сильно нелинейна, тогда лучше непосредственно искать минимум меры близости поисковыми методами (т.е. методами поиска оптимума), хотя можно попытаться решать эту нелинейную систему предназначенными для этого методами.
9. Практически можно: для этого достаточно задать в этих точках очень высокие весовые коэффициенты, что приведет к точному прохождению аппроксимирующей функции через эти точки (число таких точек не должно быть большим по сравнению с общим числом точек).
10. Квадратичный критерий близости исходной и аппроксимирующей функций является дифференцируемой функцией, что делает возможным использование необходимых условий минимума для определения параметров аппроксимирующей функции; в случае линейности аппроксимирующей функции от искомого параметров квадратичный критерий близости имеет единственный экстремум, и притом обязательно минимум.

#### *Метод равномерного приближения*

1. К достоинствам критерия близости в этом методе можно отнести твердую уверенность в отсутствии больших отклонений, чем полученное при аппроксимации, а также более очевидную оценку близости исходной и аппроксимирующей функций.
2. Аппроксимирующий полином будет являться единственным только для базовой структуры критерия близости (максимальный модуль отклонения без весового коэффициента). Если же изменять весовой коэффициент, то получится другой аппроксимирующий полином.
3. Наилучшую степень аппроксимирующего полинома можно определить путем последовательного перебора, контролируя получающуюся величину меры близости, сложность и громоздкость уравнения.
4. Необходимые условия оптимальности предполагают дифференцирование функции, а модуль, применяющийся в критерии близости метода равномерного приближения, — функция недифференцируемая.
5. Ищется минимум, так как чем меньше мера близости исходных данных и аппроксимирующей функции, тем лучше.
6. Метод равномерного приближения не получил широкого применения вследствие недифференцируемости меры близости, что вызы-

ваает необходимость применения поисковых методов (для поиска минимума).

7. Разность между расчетным и заданным значениями функции минимизируется не в какой-то конкретной точке, а в той, в которой на данный момент поиска разность является максимальной.
8. Можно степень полинома задавать произвольно, но не выше количества исходных точек без единицы.
9. Для обеспечения не сильно отличающихся относительных отклонений в различных точках при большом размахе функции можно критерий близости сформировать из относительных, а не из абсолютных отклонений.
10. Нет, не может, так как такой полином в общем виде будет содержать коэффициентов больше, чем условий для их определения (прохождение через все заданные точки). Однако если задать "усеченный" полином с меньшим числом элементов, то можно.

## ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ

### *Простейшие методы*

1. При использовании метода прямоугольников для повышения точности интегрирования можно увеличить число участков разбиения исходного интервала, на каждом маленьком интервале интеграл будет вычисляться точнее, и, если число интервалов не слишком велико, общее значение интеграла будет получено с меньшей общей погрешностью.
2. Метод прямоугольников как один из самых простых методов находит применение при приближенных (оценочных) вычислениях интегралов с невысокой точностью.
3. При использовании метода трапеций для уменьшения погрешности интегрирования можно увеличить число участков разбиения исходного интервала, на каждом маленьком интервале интеграл будет вычисляться точнее, и, если число интервалов не слишком велико, общее значение интеграла будет получено с более высокой точностью.
4. Метод трапеций как один из самых простых методов находит применение при вычислениях интегралов со сравнительно невысокой точностью, а также может использоваться при интегрировании функций с невысокой скоростью изменения.

5. Можно; если подынтегральная функция будет кусочно-постоянной, то оба метода дадут точное значение интеграла, а если она будет линейной, то метод трапеций даст точный результат.

### *Метод Симпсона*

1. Подынтегральная функция заменяется в методе Симпсона параболой второго порядка.
2. Если средняя точка берется не в середине участка, то, во-первых, изменится вычислительная формула, во-вторых, метод не будет без ошибки интегрировать кубические параболы (именно за счет выбора третьей точки в середине участка метод обеспечивает интегрирование без ошибки и полинома третьей степени, а не только второй), в-третьих, при “двойном просчете” не удастся использовать ранее рассчитанные значения подынтегральной функции. что удлинит оценку погрешности.
3. Нет, не обязательно, если на исходном участке получающаяся погрешность интегрирования получается удовлетворительной.
4. Для данной линейной подынтегральной функции результат использования метода Симпсона совпадет с результатами методов трапеций, методов Ньютона — Котеса порядка 1 и выше, методов Гаусса, Чебышева. Только метод прямоугольников даст результат с погрешностью.
5. Интегрирование без ошибки кубических парабол методом Симпсона обеспечивается потому, что третья точка, участвующая на каждом участке в вычислении интеграла, берется в середине интервала.

### *Метод Ньютона — Котеса*

1. Нет, коэффициенты не постоянны, у каждого слагаемого свое значение коэффициента, которое изменяется при смене числа используемых точек.
2. Конечно, может результат, полученный методом Ньютона — Котеса, совпасть с результатом, полученным методом с более низкой точностью, например методом левых прямоугольников. Это может иметь место, во-первых, если подынтегральная функция постоянна (или кусочно-постоянна), и, во-вторых, если формула Ньютона — Котеса использует одну точку.
3. Да, может аппроксимироваться полиномом второго порядка, а также полиномом любого порядка.
4. В принципе порядок аппроксимирующего полинома методом не ограничивается, но при высоких порядках сложно вычислять коэффициенты в квадратурной формуле и практически используется

аппроксимация полиномом не выше 5–6-го порядка, а для более высокой точности можно разбить исходный интервал интегрирования на более мелкие участки.

5. Нет, точки должны располагаться равномерно.

### *Методы Чебышева и Гаусса*

1. Преобразование исходного интервала к диапазону  $(-1, 1)$  удобно осуществлять потому, что расчетные формулы получены один раз (и навсегда) именно для этого интервала, в противном случае приходилось бы для каждого интервала заново выводить формулы или пользоваться формулами преобразования для определения новых значений табулированных переменных.
2. Метод двойного просчета эффективен в том случае, когда при вычислении интеграла с удвоенным числом шагов используются ранее рассчитанные значения подынтегральной функции. В методах Чебышева и Гаусса точки по оси  $x$  расположены неравномерно, поэтому при удвоении числа шагов новые точки не совпадают по расположению с предыдущими.
3. В методе Гаусса весовые коэффициенты в квадратурной формуле не постоянны, как в методе Чебышева, а различны в каждой точке. За счет этих дополнительных “степеней свободы” без ошибки можно интегрировать уже полином степени  $2n - 1$  а не  $n$ , как в методе Чебышева (здесь  $n$  — число точек, в которых вычисляются значения подынтегральной функции).
4. Нет, пользоваться автоматическим подбором шага здесь не удастся, так как существуют проблемы оценки погрешности интегрирования.
5. Оба метода работают только для числа точек 2, 3, 4, 5, 6, 7, 9, при других значениях не удастся получить формул для расчета значений аргумента и весовых коэффициентов.

### *Общие вопросы численного интегрирования*

1. Предложенная функция линейная, следовательно, метод трапеций даст абсолютно точный результат. Он и будет наиболее эффективен, так как он из всех методов, способных вычислить этот интеграл точно, самый простой.
2. Автоматическим подбором шага интегрирования можно пользоваться в тех случаях, когда есть возможность оценки погрешности интегрирования. Особенно это предпочтительно при равномерном расположении точек и удвоении числа шагов, так как в этом случае можно не вычислять дважды отдельные значения подынтегральной

функции, а использовать уже найденные на предыдущем шаге; это относится ко всем простейшим методам, а также и к методам Симпсона, Ньютона — Котеса.

3. Предложенная функция квадратичная, следовательно, метод Симпсона даст абсолютно точный результат. Он и будет наиболее эффективен, так как он самый простой из всех методов, способных вычислить этот интеграл точно.
4. Предложенная функция кубическая, следовательно, метод Симпсона даст абсолютно точный результат. Он и будет наиболее эффективен, так как он самый простой из всех методов, способных вычислить этот интеграл точно.
5. При уменьшении числа разбиений  $n$  исходного интервала погрешность численного интегрирования, как правило, увеличивается. Исключения составляют случаи, когда интеграл вычисляется численным методом точно, в этих случаях уменьшение числа интервалов не изменит погрешность.

## МЕТОДЫ РЕШЕНИЯ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И ИХ СИСТЕМ

### *Отделение корней*

1. Отделение корней дает интервал, в котором находится только один корень, это обеспечивает работоспособность большинства методов уточнения корней.
2. Аналитически отделить корень разрывных функций можно, так как к критическим точкам относятся и точки, в которых производные не существуют, т.е. точки разрыва.
3. Если найден интервал, где отделен корень, то для его сокращения (если, например, одна из границ его находится в бесконечности) можно задавать произвольно значения аргумента и проверять знак функции; если же нет уверенности в том, что на участке один корень, этого делать нельзя.
4. Критическими точками, которые положены в основу аналитического отделения корней, называются такие точки, в которых первая производная равна нулю или не существует.
5. Если у функции существует одна критическая точка, то корней может быть: 2 — если знаки функции при  $x \rightarrow \infty$  и  $x \rightarrow -\infty$  одинаковы и противоположны знаку функции в критической точке; 1 — если один из знаков при  $x \rightarrow \infty$  или  $x \rightarrow -\infty$  совпадает со знаком функции

в критической точке; 0 — если все знаки функции в упомянутых точках одинаковы.

6. Для отыскания критических точек функции может оказаться необходимым решать нелинейное уравнение  $f'(x) = 0$ , что само по себе так же трудно и требует отделения корней, как и решение исходного заданного уравнения.

### ***Метод деления отрезка пополам***

1. Геометрический смысл метода половинного деления заключается в последовательном делении отрезка, где локализован корень, на две равные части.
2. Если нелинейная функция в левой части уравнения непрерывна, то метод половинного деления всегда позволит получить корень с заданной погрешностью, так как процесс решения в этом случае не зависит от свойств функции.
3. Один из концов отрезка следующего интервала в методе половинного деления всегда находится в середине текущего отрезка, а второй — на конце той половины, где функция  $f(x)$  имеет другой знак, чем в уже выбранной точке (чем в середине текущего отрезка).
4. Функции  $f(x)$  достаточно быть непрерывной, чтобы можно было гарантированно решить уравнение  $f(x) = 0$ .
5. Для нахождения хотя бы одного действительного корня уравнения  $f(x) = 0$  методом половинного деления необходимо выполнить операцию отделения корней, в противном случае корень может найтись в том случае, если при каждом делении отрезка пополам на одной из половинок будет получаться разный знак функции  $f(x)$  на ее концах.
6. Можно найти корень, находящийся на границе интервала.

### ***Метод хорд***

1. Метод хорд позволяет определять предварительно отделенные корни.
2. Геометрический смысл метода хорд заключается в замене нелинейной функции  $f(x)$  на участке отделения линейной, проходящей через концы участка, т.е. хордой.
3. Нет, не всегда. Для получения решения с заданной погрешностью, во-первых, функция должна быть монотонной (по крайней мере в окрестности корня), во-вторых, она должна иметь невысокую крутизну (см. требования к соотношению минимального и максимального значения производной на интервале).

4. В методе хорд для монотонной функции  $f(x)$  один конец является закрепленным, а второй определяется точкой пересечения хорды с осью  $x$ . Закрепленный конец выбирается исходя из анализа знаков функции и ее второй производной на концах интервала.
5. Для того чтобы решить нелинейное уравнение  $f(x)$  методом хорд, функция  $f(x)$  должна быть непрерывной и монотонной.
6. Закрепленный конец зависит от свойств функции нелинейного уравнения и может быть различным.

### ***Метод Ньютона***

1. Геометрический смысл метода Ньютона заключается в замене нелинейной функции  $f(x)$  на участке отделения линейной, являющейся касательной к одному из концов отрезка.
2. Начальное приближение  $x_0$  в методе Ньютона выбирается таким образом, чтобы касательная к функции в точке  $x_0$  пересекала ось  $x$  внутри начального интервала, где отделен корень. Это оценивается по знакам функции и второй производной или практически методом проб и ошибок.
3. “Закрепленным” будет правый конец.
4. “Закрепленным” будет правый конец.
5. Для того чтобы решить нелинейное уравнение  $f(x)$  методом Ньютона, функция  $f(x)$  должна быть непрерывной и монотонной.
6. Если функция  $f(x)$  немонотонна, то метод Ньютона в классическом варианте может не дать гарантированный результат (можно в отдельных случаях получить корень, если на каждом шаге заново определять закрепленный конец).

### ***Метод параболической аппроксимации***

1. Геометрический смысл метода параболической аппроксимации заключается в замене нелинейной функции  $f(x)$  на участке отделения параболой второго порядка, построенной по трем точкам.
2. Метод параболической аппроксимации представляет собой последовательность процедур аппроксимации и нахождения корня параболы (обе процедуры доведены до конечных расчетных формул).
3. Концы отрезка интервала на втором и последующих шагах выбираются так, чтобы последнее найденное приближение корня оказалось средней точкой, а крайними точками отрезка были ближайшие к ней, лежащие слева и справа.
4. В случае колебательного характера нелинейной функции аппроксимирующая парабола может или иметь две точки пересечения с осью

$x$ , или не иметь вовсе; в обоих случаях решение уравнения найти не удастся.

5. Нет, они могут лежать по разные стороны от корня.
6. Может, но гарантировать какой именно — нельзя.

### *Метод итераций*

1. Левая часть уравнения  $f(x) = 0$  в методе итераций не заменяется никакой функцией.
2. Сходимостью называется стремление текущего приближения к корню при увеличении числа шагов.
3. Приближение к корню может осуществляться с одной стороны (справа или слева) и с обеих сторон, т.е. иметь как бы колебательный характер приближения к корню.
4. Если на участке оказалось два корня, то в этом случае должно быть две точки пересечения прямой  $y = x$  с кривой  $y = \varphi(x)$ , для одной будут выполняться условия сходимости, а для другой — нет (если  $y = \varphi(x)$  не имеет разрывов).
5. Несходимость итерационного процесса уточнения корня может означать отсутствие корня или невыполнения условий сходимости, в последнем случае можно попробовать изменить структуру функции  $\varphi(x)$ , т.е. по другому алгоритму привести исходное уравнение  $f(x) = 0$  к виду, удобному для итераций.
6. Да, есть. При “колебательном” приближении к корню можно контролировать величину отрезка, где локализуется корень (он равен модулю разности двух соседних приближений), а при одностороннем приближении в условия сходимости входит сомножитель, зависящий от максимального значения производной  $\varphi'(x)$  на участке.

### *Особенности решения алгебраических уравнений*

1. Алгебраические уравнения имеют определенную структуру, благодаря чему можно строить более эффективные алгоритмы. Кроме того, при решении алгебраических уравнений появляется возможность оценивать количество корней заданного вида (например, положительных, отрицательных, попадающих в заданный интервал), область существования корней.
2. Нет, не всегда. Метод понижения порядка не требует предварительного отделения корней, найденный корень на каждом этапе будет определяться начальным приближением, которое выбирается произвольно (не следует забывать, что от близости начального приближения к корню существенно зависит сходимость метода).

3. С помощью правила Декарта можно определить точно количество положительных корней только в том случае, если их количество получилось равное 1 или 0.
4. Если полином  $P(x)$  в левой части уравнения является полным, то число действительных отрицательных корней определится по числу постоянств знаков в последовательности коэффициентов полинома  $P(x)$ .
5. Общее число корней алгебраического уравнения равно наивысшей степени полинома  $P(x)$  левой части уравнения.
6. Сумма положительных и отрицательных корней не обязательно равна наивысшей степени полинома в левой части уравнения, так как могут существовать и сопряженные комплексные корни.

### *Системы нелинейных уравнений*

1. Отделение корней при решении систем, как правило, не проводят так как в общем случае это сделать крайне трудно или вообще невозможно.
2. Фактически при расчете величины шага в методе Ньютона — Рафсона нелинейные функции линеаризуются, вследствие этого и не получается в новой точке решения, она рассматривается лишь как приближение к решению, и метод приобретает итерационный характер.
3. Скорость сходимости метода Ньютона — Рафсона существенно зависит от выбора начального приближения.
4. Сходимость метода итераций можно обеспечить способом приведения к виду, удобному для итераций (подробнее см. соответствующий раздел книги).
5. Для повышения точности решения достаточно уменьшить задаваемую погрешность, по достижении которой прекращается процесс решения.
6. Да, оказывает. Поскольку корни отделить практически невозможно, то выбор начальной точки поиска определяет корень, который будет найден; фактически перебирая различные начальные условия, можно находить разные корни, и если повезет, то даже все.

## **СИСТЕМЫ ЛИНЕЙНЫХ УРАВНЕНИЙ**

1. Точные методы решения систем линейных уравнений позволяют получить решение непосредственно в аналитическом виде или в виде алгоритма с конечным числом шагов, а приближенные — как предел некоторой последовательности при стремлении числа итераций к бесконечности.

2. Систему из трех уравнений, конечно, проще всего решать любым точным методом, например методом Крамера.
3. Итерационные методы наиболее предпочтительны в задачах высокой размерности, когда точные методы могут дать решение с большой накопившейся вычислительной погрешностью.
4. Скорость сходимости в методе итераций зависит от свойств системы уравнений (определяющейся коэффициентами уравнений) и от начальных условий поиска решения.
5. Если размерность задачи высока, то получить решение с погрешностью не хуже заданной точным методом невозможно, так как нет приемов коррекции накапливающихся вычислительных ошибок, становящихся значительными при большом числе вычислений, а есть только приемы их контроля.
6. В методе Гаусса можно ввести дополнительные переменные в каждом уравнении, равные сумме всех коэффициентов уравнения, с которыми проводят все операции как с коэффициентами. При этом невыполнение равенства этих переменных сумме коэффициентов является признаком накопившейся вычислительной погрешности.
7. Метод Крамера относится к точным методам.
8. Метод итераций будет сходиться, если все суммы модулей коэффициентов строк (или столбцов) преобразованной системы уравнений будут меньше единицы.
9. Число шагов для получения решения с заданной погрешностью можно оценить заранее, используя выражение для погрешности. Для этого надо подставить в выражение заданное значение погрешности и решить получающееся уравнение относительно числа итераций (оно находится в показателе степени).
10. Наличие вычислительной ошибки в итерационных методах приводит лишь к увеличению числа итераций для достижения решения, так как итерационные методы обладают свойством самоисправляемости ошибок.

## ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ

### *Метод Эйлера*

1. Решением дифференциального уравнения является функция, а не число, как при решении конечных уравнений.
2. Начальные условия должны задаваться для численного решения дифференциального уравнения.

3. Модифицированный метод Эйлера относится к группе одношаговых методов, так как для нахождения значения функции в следующей точке требуется знание только одной текущей точки.
4. Модифицированный метод Эйлера имеет погрешность, пропорциональную  $h^2$ , поскольку он является, в отличие от метода Эйлера, методом второго порядка.
5. Основные отличия одношаговых и многошаговых методов заключаются в следующем:
  - одношаговые методы требуют для расчета следующей точки знание только одной текущей, а многошаговые требуют знание нескольких предыдущих точек;
  - одношаговые обладают свойством самостартования, а многошаговые требуют получения нескольких предыдущих точек для запуска, что обычно делается одношаговыми методами;
  - многошаговые методы не допускают применение переменного шага решения уравнения, а одношаговые допускают.
6. Метод Эйлера позволяет решать системы дифференциальных уравнений.
7. Можно решать задачи, не относящиеся к задачам Коши, хотя существуют другие, более подходящие для этой цели методы.
8. Задание начальных условий при решении задачи Коши обязательно.
9. В явных схемах модифицированного метода Эйлера искомое значение функции в следующей точке выражается в явной форме (при этом придется сначала найти приближенное значение, а затем уточнить его), в неявных схемах искомое значение функции находится из решения нелинейного уравнения, так как оно не выражается в явном виде.
10. Погрешность решения можно оценить с использованием приема двойного просчета; именно этот прием положен в основу автоматического подбора шага для получения решения с заданной погрешностью.

### *Метод Рунге — Кутты*

1. При использовании метода четвертого порядка необходимо на каждом шаге четыре раза вычислять правую часть уравнения.
2. Погрешность решения можно оценить с использованием приема двойного просчета; именно этот прием положен в основу автоматического подбора шага для получения решения с заданной погрешностью.

3. Оценку погрешности для подбора шага можно задавать как в относительных, так и в абсолютных величинах.
4. Метод относится к одношаговым, поэтому для расчета следующего значения функции не нужно ни одного предыдущего значения.
5. Метод Рунге — Кутта относится к численным методам решения дифференциальных уравнений.
6. Рекуррентная формула при решении уравнения  $y' = f(x, y)$  имеет вид:  $y^{j+1} = y^j + (k_1 + 2k_2 + 2k_3 + k_4) / 6$ , где  $j$  — номер шага,  $k_1, k_2, k_3, k_4$  — коэффициенты, рассчитываемые по формулам

$$k_1 = \Delta x f(x, y), \quad k_2 = \Delta x f\left(x + \frac{\Delta x}{2}, y + \frac{k_1}{2}\right),$$

$$k_3 = \Delta x f\left(x + \frac{\Delta x}{2}, y + \frac{k_2}{2}\right), \quad k_4 = \Delta x f(x + \Delta x, y + k_3),$$

где  $\Delta x$  — шаг решения.

7. К недостаткам метода (четвертого порядка) можно отнести необходимость четырехкратного вычисления правых сторон уравнений на каждом шаге.
8. Погрешность метода пропорциональна  $\Delta x^4$ , где  $\Delta x$  — шаг решения.
9. Метод Рунге — Кутта является одношаговым, поэтому можно применять переменный шаг при его использовании, так как для нахождения следующей точки требуется только одна текущая.
10. Шаг решения можно автоматически подобрать последовательно уменьшая его вдвое, оценивая при этом разность найденного значения функции при двух значениях шага: как только разница не будет превышать заданной погрешности, так текущий шаг можно принять за шаг решения уравнения.

### *Многошаговые методы*

1. К достоинствам многошаговых методов относят меньший требующийся объем вычислений при реализации метода, так как при одинаковом порядке метода (например, четвертом в методах Рунге — Кутта и Милна) требуется не четыре, а только два раза вычислять правую часть дифференциального уравнения (хотя требуется дополнительная память для хранения предыдущих точек).
2. Метод Милна реализуется за два этапа: прогноза и коррекции.
3. На этапе прогноза находится достаточно грубое приближенное значение искомой функции в заданной точке.

4. На этапе коррекции определяется с учетом ранее найденного прогноза более точного значения функции в заданной точке.
5. Этап коррекции может проводиться итерационно: если разница между прогнозируемым и скорректированным значениями достаточно велика, то на новом этапе коррекции предыдущее скорректированное значение используется в качестве прогнозного, и так может повторяться многократно, т.е. реализуется итерационный процесс.
6. В методах прогноза и коррекции часть необходимых для расчета нового значения функции величин берется из предыдущих точек, а не вычисляется на каждом шаге, как в одношаговых; поэтому число раз вычисления правой части уравнения меньше, чем в одношаговых методах.
7. Погрешность в методе Милна можно оценить по формуле  $\epsilon \approx \approx |y_{i+1} - y''_{i+1}|/29$ .
8. На первых шагах при использовании методов прогноза и коррекции отсутствуют значения функции в предыдущих точках; для их получения и применяют одношаговые методы.
9. Методами прогноза и коррекции можно решать систему дифференциальных уравнений.
10. Порядок метода определяет количество предыдущих значений функции, требующееся для расчета одного следующего.

## **МЕТОДЫ ОДНОМЕРНОГО ПОИСКА**

### *Метод деления пополам*

1. Метод пригоден для любых непрерывных одноэкстремальных функций, не имеющих горизонтальных участков.
2. Сходимость метода и его эффективность не зависят от свойств функции.
3. Точки вычисления критерия оптимальности располагаются слева и справа от точки середины текущего отрезка, где локализован экстремум, на расстоянии, равном желаемой погрешности.
4. Повысить точность нахождения решения можно, задавая меньшее значение погрешности.
5. Решение считается достигнутым тогда, когда величина отрезка, где локализован экстремум, станет не больше заранее заданной погрешности по  $x$ .

6. Если функция непрерывная одноэкстремальная, не имеющая горизонтальных участков, то вид ее не оказывает влияния на процесс нахождения решения. Если же имеются горизонтальные участки, то очередная середина отрезка может попасть на него, и тогда слева и справа от этой середины функция будет иметь одинаковые значения, что не позволит выбрать половинку, в которой находится экстремум.
7. Если функция удовлетворяет предъявляемым к ней свойствам (см. ответы на вопросы 1 и 5), метод всегда гарантированно даст решение.
8. В качестве следующего берется тот отрезок, в котором значение функции больше (при поиске максимума).
9. Для вычисления погрешности в 1 % от величины интервала следует произвести 7 делений отрезка пополам (каждое деление уменьшает отрезок в два раза, следовательно, при 7 делениях отрезок уменьшится в  $2^7 = 128$  раз), а при каждом делении вычисляется два значения функции (слева и справа), поэтому общее число раз вычисления критерия будет 14.
10. Нет, не может. Погрешность задана в относительных единицах от величины интервала, следовательно, при уменьшении интервала уменьшится и величина погрешности.

### *Метод сканирования*

1. Метод позволяет найти экстремум для любых непрерывных функций.
2. Метод позволяет находить глобальный экстремум, и эффективность не зависит от свойств функции, экстремум которой ищется.
3. Точки вычисления критерия “размещаются” на оси  $x$  равномерно с шагом, равным заданной погрешности по  $x$ .
4. Основная задача модернизации метода заключается в снижении количества повторов вычисления функции для получения решения с заданной погрешностью.
5. Основное достоинство модернизированного метода заключается в снижении количества повторов вычисления функции для получения решения с заданной погрешностью, но при этом повышается вероятность пропуска “острого” глобального экстремума.
6. Точность нахождения решения можно повысить путем уменьшения задаваемой погрешности.
7. Решением является то значение  $x$ , которое соответствует наибольшему (при поиске максимума) значению функции.

8. Самое большое значение  $R(x)$  можно найти путем сравнения всех вычисленных значений; это нетрудно, так как число их конечно.
9. Метод очень легок в алгоритмизации, так как не содержит никаких специальных операций, кроме последовательного вычисления функции в заданных точках, при этом выбор максимального значения может совмещаться с переходом от точки к точке.
10. Вид функции  $R(x)$ , если она удовлетворяет предъявляемым требованиям непрерывности, никак не влияет на процесс нахождения решения.

### *Метод золотого сечения*

1. Нет, не может. Погрешность задана в относительных единицах от величины интервала, следовательно, при уменьшении интервала уменьшится и величина погрешности и затраты на поиск не изменятся.
2. Если функция непрерывна и одноэкстремальна, то метод гарантированно даст решение.
3. Вид функции, если она удовлетворяет требованиям метода (см. предыдущий вопрос и ответ), не оказывает влияния на процесс нахождения решения.
4. Следующий отрезок берется тот, внутри которого находится максимальное (при поиске максимума) на текущем этапе поиска значение функции. Отрезок, внутри которого находится максимум, определяется простым сравнением значений критерия в точках.
5. Метод золотого сечения обеспечивает наиболее эффективный поиск экстремума функции для произвольных непрерывных функций.
6. Для повышения точности нахождения решения необходимо просто уменьшить задаваемую погрешность.
7. Вообще говоря, вид функции никак не влияет на процесс нахождения решения, если функция имеет один экстремум.
8. По правилу золотого сечения на каждом шаге делится тот интервал, внутри которого на данном этапе находится искомым экстремум.
9. Золотым сечением называется такое деление отрезка на две неравные части, при котором отношение меньшей части к большей равно отношению большей части ко всему отрезку, т.е.  $a/c/cb = cb/ab$ .
10. На каждом шаге, кроме первого, когда необходимо дважды вычислить значение функции, функция вычисляется только один раз. Так происходит потому, что уже имеющаяся точка на участке от предыдущего этапа является одной из точек золотого сечения отрезка.

### *Метод параболической аппроксимации*

1. Методом параболической аппроксимации можно найти экстремум непрерывной одноэкстремальной функции.
2. Метод может обеспечить более высокую сходимость к решению за счет квадратичной, а не линейной аппроксимации функции на каждом шаге.
3. Условием окончания метода является сокращение величины интервала, где локализован экстремум, до уровня заданной погрешности.
4. Аппроксимирующая парабола строится по трем точкам; их достаточно для нахождения трех коэффициентов параболы из условия ее прохождения через эти точки.
5. В общем случае повышение степени аппроксимирующей параболы, с одной стороны, может привести к увеличению сходимости метода, а с другой — к дополнительным затратам по вычислению коэффициентов такой параболы и нахождению ее максимума.
6. Для повышения точности нахождения решения необходимо просто уменьшить задаваемую погрешность.
7. Нет, не всегда. В отдельных случаях парабола может иметь на каком-то шаге экстремум, находящийся за пределами текущего интервала даже для одноэкстремальной функции.
8. На первом шаге в качестве точек для построения параболы берутся крайние и средняя точки исходного интервала; на следующих шагах в качестве внутренней точки обычно берется точка с наибольшим текущим значением функции, а в качестве крайних — две ближайшие слева и справа от нее.
9. Да, нахождение решения за один шаг возможно, но только в случае поиска экстремума квадратичной функции, которая точно совпадет с аппроксимирующей параболой.
10. Вид функции  $R(x)$  оказывает довольно сложное влияние на процесс нахождения решения; поиск будет лучше, если функция будет более гладкой.

## ГРАДИЕНТНЫЕ МЕТОДЫ МНОГОМЕРНОГО ПОИСКА

### *Метод градиента*

1. При применении алгоритма с использованием направляющих косинусов поиск будет более эффективным, так как появляется возможность однозначно управлять величиной шага путем выбора

коэффициента пропорциональности шага. В исходном базовом алгоритме величина шага зависит еще и от величины модуля градиента, который трудно оценить заранее, хотя в этом случае по мере приближения к оптимуму шаг сам будет уменьшаться за счет уменьшения градиента, а в рекомендуемом методе необходимо применять специальные приемы.

2. Обычно угол между двумя соседними направлениями поиска при приближении к оптимуму меняется очень сильно, так как имеет место “проскок” экстремума при почти каждом шаге.
3. Градиентом функции  $R(x_1, x_2)$  называется вектор, который показывает направление и скорость возрастания функции; он может быть получен через свои проекции на оси координат, которые равны соответствующим частным производным от  $R$ :  $\text{grad } R(x_1, x_2) = \{ \partial R / \partial x_1, \partial R / \partial x_2 \}$ .
4. Основные свойства градиента вытекают из его определения: направление градиента совпадает с направлением самого крутого возрастания функции (именно поэтому его и применяют для определения направления движения к оптимуму), градиент всегда перпендикулярен линии уровня, проходящей через ту точку, в которой вычислен градиент, модуль градиента (т.е. величина градиента) характеризует скорость возрастания функции.
5. Эффективность поиска оценивается по числу повторов вычисления функции, экстремум которой ищется: чем меньше раз вычислялась функция для получения решения с заданной погрешностью, тем эффективнее поиск.
6. Вблизи оптимума предпочтение можно отдать алгоритму, в котором коэффициент пропорциональности шага уменьшается вдвое при проскоке экстремума.
7. При использовании алгоритма  $x^j = x^{j-1} - h \text{grad } R(x)$  (знак “минус” стоит в случае поиска минимума функции) величина шага  $\Delta x = h \text{grad } R(x)$  убывает в районе оптимума вследствие уменьшения градиента  $R(x)$  (в точке экстремума градиент равен нулю).
8. Отличие алгоритмов вычисления шага  $\Delta x = h \text{grad } R(x)$  и  $\Delta x = h \cos \varphi$  прежде всего заключается в отсутствии влияния величины градиента на величину шага во втором алгоритме, благодаря чему выбору коэффициента пропорциональности шага  $h$  можно однозначно управлять шагом, увеличивая вдали от оптимума или уменьшая его вблизи оптимума.

9. При поиске максимума функции в алгоритме шага стоит знак “+” (ищем в направлении градиента), а при поиске минимума — знак “-” (ищем в направлении, противоположном направлению градиента); в обоих случаях эффективность метода одинакова.
10. Вычисление градиента по методу с парными пробами, с одной стороны, ухудшает поиск (появляются дополнительные вычислительные затраты при определении частных производных); с другой стороны, улучшает поиск за счет более точного вычисления градиента.

### *Метод наискорейшего спуска*

1. Основные ограничения метода наискорейшего спуска от метода градиента заключаются в том, что в методе градиента градиент вычисляется на каждом шаге, а в методе наискорейшего спуска после одного вычисления градиента следует одномерный поиск оптимума по направлению градиента, т.е. делается несколько шагов при одном вычислении градиента.
2. При поиске минимума  $R(x)$  поиск из каждой точки осуществляется в направлении, противоположном направлению градиента в текущей точке.
3. Градиент в методе наискорейшего спуска вычисляется точно так же, как и в методе градиента; способ вычисления градиента не зависит от метода поиска.
4. Направление градиента не остается постоянным в области поиска, поэтому экстремум в одном направлении не соответствует экстремуму функции, и требуется опять и опять искать оптимум по новым направлениям градиентов, пока не будет выполнено условие окончания поиска.
5. В качестве условия окончания поиска задается требуемая малость модуля градиента функции, т.е. должно выполняться условие  $|\text{grad } R(x)| \leq \epsilon$  (в области оптимума градиент точно равен нулю, но достичь этого значения практически невозможно, можно лишь приблизиться сколь угодно близко к нему).
6. Наиболее эффективен метод вдали от оптимума, особенно при отсутствии “оврагов” функции.
7. Даже при одноэкстремальной функции по направлению градиента может иметь место многоэкстремальность, что ограничивает применимость многих методов одномерного поиска; на практике чаще всего применяют метод последовательного сканирования до первого локального экстремума по направлению.

8. Конечно, можно найти максимум функции; название метода просто сохранило исторические корни происхождения метода, вид находимого экстремума зависит только от того, в какую сторону по градиенту совершаются рабочие шаги.
9. Нет, нельзя, так как два соседних направления поиска в методе наискорейшего спуска всегда ортогональны.
10. Влияние вычислительных погрешностей не очень существенно, оно приводит к неортогональности следующего направления поиска и как следствие к некоторому замедлению поиска в целом.

### *Метод тяжелого шарика*

1. “Масса шарика” увеличивает его инерционность, что приведет к колебаниям поиска в окрестности оптимума.
2. Да, поиск может ускориться, в этом основное достоинство метода. Ускорение происходит вследствие “разгона” шарика за счет его инерционности.
3. Можно найти любой вид экстремума, проведя коррекцию знаков в формулах рабочих шагов (“+” заменить на “-”).
4. К недостаткам метода относят необходимость задания сразу двух параметров, определяющих эффективность поиска, влияние которых взаимосвязано.
5. Метод пригоден для задач любой размерности, но градиентные методы в одномерных задачах вырождаются, так как градиент всегда направлен по оси  $x$ .
6. Да, при удачном выборе двух параметров, определяющих эффективность метода, метод может остановиться в “глубоком” экстремуме. Это происходит за счет инерционности “шарика”, вследствие которой он “выскакивает” из “мелких” локальных экстремумов.
7. “Помещение шарика в вязкую среду” способствует затуханию колебаний в районе оптимума.
8. В приведенном алгоритме  $x^{j+1} = x^j - \alpha(x^j - x^{j-1}) - h \text{grad} R(x^j)$  величина  $\alpha$  определяет “память” алгоритма, т.е. учитывает влияние предыдущей точки, поэтому увеличение этого параметра метода может привести к более быстрому затуханию в районе оптимума, когда градиент функции мал.
9. Да, метод тяжелого шарика является методом второго порядка, так как новая точка поиска рассчитывается с использованием двух предыдущих.

10. Метод тяжелого шарика предпочтительно использовать при поиске экстремума многоэкстремальной функции с одним явно выраженным глобальным экстремумом и “мелкими” локальными, хотя он эффективно работает и при поиске экстремума одноэкстремальных функций.

### *Метод сопряженных градиентов*

1. Квадратичные методы оптимизации от линейных отличаются использованием в алгоритме вторых производных (или двух предыдущих точек для расчета одной следующей) и, как следствие, более высокой скоростью сходимости к решению, а также более сложным в реализации алгоритмом.
2. Метод сопряженных градиентов более эффективен, так как он относится к методам второго порядка.
3. Алгоритм метода записывается следующим образом:
$$x^{j+1} = x^j - h(\text{grad } R(x^j) + \alpha \text{ grad } R(x^{j-1})).$$
4. Влияние вычислительных погрешностей очень существенно: при их наличии поиск резко замедляется, так как они значительно искажают вторые производные (которые присутствуют в алгоритме косвенно).
5. Метод наиболее эффективен для квадратичных функций.
6. Необходимость прямо или косвенно вычислять вторые производные, которые очень чувствительны к различным погрешностям.
7. Различия в первом шаге методом наискорейшего спуска и методом сопряженных градиентов нет.
8. Ищется экстремум критерия в направлении поиска.
9. Эффективность методов вдали от оптимума примерно одинакова.
10. Нет, невозможно, так как метод предполагает нахождение производных для вычисления градиента.

## **БЕЗГРАДИЕНТНЫЕ МЕТОДЫ МНОГОМЕРНОЙ ОПТИМИЗАЦИИ**

### *Метод Гаусса — Зайделя*

1. Если в функции, оптимум которой мы ищем, нет взаимодействия переменных (которое выражается, например, в произведении переменных), то одного цикла поиска достаточно, чтобы найти экстремум функции. Если же такое взаимодействие есть, то положение

экстремума по одной переменной зависит от значения других переменных, поэтому найденное экстремальное значение в одном цикле поиска не соответствует экстремуму функции и недостаточно провести поиск по всем переменным один раз.

2. Областью наиболее предпочтительного использования метода является оптимизация сепарабельных функций (в которых отсутствуют взаимосвязи переменных), так как в этом случае оптимум по каждой из переменных не зависит от значения других, и решение найдется за один цикл.
3. Можно найти оптимум квадратичной функции за один цикл, если эта функция сепарабельная.
4. Да, можно, и не только можно, а это будет обязательно, так как функция сепарабельная.
5. Порядок чередования переменных не может влиять на получаемое решение, так как решение находится по задаваемой погрешности; порядок чередования влияет лишь на эффективность поиска, т.е. на затраты.
6. Порядок чередования переменных влияет на эффективность поиска, т.е. на затраты, так как свойства функции в общем случае в различных направлениях различные.
7. В качестве условия окончания поиска используется невозможность нахождения лучшего значения функции ни по одной переменной.
8. Поиск с последствием как модификация метода необходим для исключения “застревания” поиска в узких оврагах (особенно имеющих направление примерно по биссектрисе координатных углов); последствие позволяет “проскакивать эти ловушки”.
9. Основное достоинство метода — простота, заключающаяся в последовательном применении одномерных (несложных по сравнению с многомерными) методов поиска.
10. Основной недостаток метода — низкая эффективность поиска для функций, имеющих явно выраженное взаимодействие переменных.

### *Метод Розенброка*

1. Метод “запускается” выполнением первого цикла по методу Гаусса — Зайделя.
2. Этот недостаток вообще не свойствен методу.
3. Метод наиболее предпочтителен для квадратичных несепарабельных функций, так как в этом случае решение найдется после первого поворота осей координат за один цикл.

4. Нет, не существует однозначного положения осей в начале каждого цикла: в направлении изменения функции в предыдущем цикле “устанавливается” ось той переменной, которая будет первой изменяться в следующем цикле, остальные оси располагаются ортогонально по отношению к этой оси (а при размерности задачи 3 и выше положение других осей, удовлетворяющее этому требованию, не единственное).
5. Нет, не обязательно новая система координат должна быть правой, так как по каждой оси экстремум ищется в обоих направлениях.
6. В случае сепарабельной функции решение найдется за первый же цикл, который выполняется в соответствии с методом Гаусса — Зайделя, поэтому поворачивать оси вообще не придется.
7. В случае поиска экстремума квадратичной функции необходимо всего сделать два цикла поиска (даже в случае несепарабельной функции), поэтому придется сделать  $2n$  одномерных поисков, где  $n$  — размерность задачи.
8. При оптимизации любой сепарабельной функции (в том числе трехмерной квадратичной) оси координат вообще не придется поворачивать, так как оптимум найдется за один цикл поиска.
9. Нет, не обязательно поворачивать оси координат при поиске экстремума  $n$ -мерной квадратичной сепарабельной функции, так как при оптимизации любой сепарабельной функции оптимум найдется за один цикл поиска.
10. Да, для квадратичной сильно овражной функции метод будет работать очень эффективно, с его помощью оптимум найдется за два цикла (т.е. после всего одного поворота осей координат).

### *Симплексный метод*

1. Симплексом в  $n$ -мерном пространстве называется фигура, содержащая  $n+1$  вершину (на плоскости — треугольник, в трехмерном пространстве — тетраэдр).
2. В базовом методе вершина нового симплекса находится путем отражения “выбрасываемой” симметрично относительно центра противоположащей грани.
3. Признаком заикливания симплексного метода является “выбрасывание” только что полученной новой вершины.
4. Причиной заикливания является выход в область оптимума; искомый оптимум при заикливании находится внутри (или на границе) симплекса.
5. Условием растяжения симплекса является получение новой точки, которая лучше самой хорошей точки предыдущего симплекса.

6. Алгоритм сжатия симплекса можно представить в следующем виде:

$$\tilde{x}^j = bx^j + (1-b) \sum_{\substack{i=1 \\ i \neq j}}^n x^i.$$

7. Условием окончания поиска является заданная малость размера симплекса.
8. В базовом алгоритме точкой, относительно которой отражается старая вершина при построении новой, является середина противоположной грани. В некоторых модификациях такой точкой является центр “тяжести” противоположной грани (под “тяжестью” вершин грани понимается значение функции, экстремум которой ищем).
9. В овражных функциях эффективнее будет работать модификация, в которой новая вершина получается отражением относительно центра “тяжести” противоположной грани.
10. Базовому методу соответствует алгоритм

$$\tilde{x}^j = \frac{2}{n} \sum_{i=1}^{n+1} x^i - \left(1 - \frac{2}{n}\right) x^j.$$

### *Метод параллельных касательных*

1. Направление одномерных исходных поисков выбирается произвольно.
2. Метод наиболее эффективен для квадратичных функций.
3. Основное достоинство метода — простота, заключающаяся в последовательном применении одномерных (несложных по сравнению с многомерными) методов поиска.
4. Метод более эффективен для задач невысокой размерности, так как с ростом размерности резко растет число одномерных поисков, необходимых для получения решения.
5. Метод является итерационным при неквадратичной функции, экстремум которой ищем.
6. Нет, не обязательно: оптимум будет располагаться на прямой, соединяющей точки оптимума по параллельным направлениям, но не только между этими точками.
7. Для квадратичной сепарабельной функции эффективность метода не зависит от выбора начальных направлений поиска, так как в этом случае решение обязательно найдется за три одномерных поиска.

8. Общие затраты поиска несомненно зависят от выбора начальной точки: во-первых, число шагов до оптимума по направлению бесспорно зависит от выбора начальной точки; во-вторых, от положения начальной точки зависит “расстояние” до оптимума.
9. Да, зависит: погрешность поиска оптимума по направлению фактически приведет к тому же результату, что и поиск для неквадратичной функции.
10. В общем случае поиск можно заканчивать, если из последней найденной в соединяющем две точки направлении оптимальной точки не удастся найти лучшую точку по исходному направлению.

## МЕТОДЫ СЛУЧАЙНОГО ПОИСКА

1. Основные преимущества случайных методов поиска заключаются в высокой эффективности методов в задачах высокой размерности и в возможности большинства методов отыскивать глобальный экстремум.
2. Достоинство слепого поиска — в высокой вероятности нахождения глобального оптимума.
3. К недостаткам слепого метода следует отнести высокое число вычислений (или измерений) функции, экстремум которой ищем.
4. Начальная точка в методе блуждающего поиска может выбираться случайным образом, как и в других методах.
5. Конечно может (в любом методе) начальная точка задаваться на основе априорных сведений, а не случайно.
6. Конечно может (в любом методе) начальная точка задаваться на основе априорных сведений, а не случайно.
7. Эффективность этих методов для двухмерной задачи примерно одинакова, для многомерных — метод с “наказанием случайностью” может оказаться более эффективным.
8. Для овражной функции метод случайных направлений более эффективен, так как свойства функции не оказывают существенного влияния на характер поиска.
9. Случайный вектор, используемый в методе случайных направлений, имеет фиксированный модуль, а случайное — только направление.
10. Для более высокой точности решения задачи следует увеличить число попыток получения лучшего решения.

## МЕТОДЫ МНОГОМЕРНОЙ УСЛОВНОЙ ОПТИМИЗАЦИИ

### *Метод штрафов*

1. Штраф “накладывается” на исходную функцию, экстремум которой мы ищем.
2. Коэффициент штрафа увеличивают для более точного нахождения оптимума исходной функции, так как при больших штрафах будет меньше нарушение ограничений. Возрастание овражности, появляющееся с ростом коэффициента штрафа, не страшно, поскольку увеличение производят в районе оптимума.
3. Уменьшать коэффициент штрафа можно для снижения овражности расширенной функции вдали от оптимума, с тем чтобы быстрее выйти в район искомого оптимального решения.
4. Начальная точка поиска выбирается произвольно.
5. Для уменьшения овражности расширенного критерия оптимальности необходимо уменьшать коэффициент штрафа.
6. Да, в процессе поиска происходит нарушение ограничений, причем тем больше, чем меньше коэффициент штрафа.
7. Искать оптимум расширенного критерия оптимальности предпочтительнее всего методами овражного поиска, так как его скорость изменения в различных направлениях сильно различается.
8. Точность нахождения оптимума исходной задачи зависит, конечно, от задаваемой погрешности решения и от величины коэффициента штрафа (при большем коэффициенте точность будет выше).
9. Ускорить процесс решения можно путем последовательного решения задачи с постепенно возрастающим коэффициентом штрафа: при малых коэффициентах процесс поиска будет идти быстрее, но будут значительно нарушаться ограничения, с ростом коэффициента процесс поиска будет замедляться (но это уже не страшно, так как это происходит сравнительно близко к оптимуму), но ограничения будут нарушаться мало.
10. При поиске максимума критерия в исходной задаче штраф необходимо вычитать из исходного критерия (т.е. уменьшением “наказывать” его, штрафовать).

### *Метод прямого поиска с возвратом*

1. Вспомогательный критерий просто равен исходному критерию.
2. После достижения ограничения типа равенств  $f(x) = 0$  (т.е. после возврата на ограничение) дальнейший поиск осуществляется по

градиенту (при использовании градиентных методов, широко применяющихся здесь) исходного критерия оптимальности.

3. Возврат в допустимую область происходит при достижении ограничения  $F(x) = 0$ .
4. Чем больше будет ширина допустимого “коридора” для ограничения типа равенства, тем меньше будет число возвратов на ограничение, но в окрестности оптимума может возникнуть зацикливание.
5. Если не будет зацикливания (при котором обычно сокращают величину “коридора”), то на точность решения ширина коридора не влияет, так как окончание поиска (при ограничении типа равенства) осуществляется по близости двух соседних точек возврата на ограничения.
6. Коридор в задаче оптимизации при ограничении  $F(x) \geq 0$  не вводится вообще.
7. Нет, не может, так как окончание поиска осуществляется по близости двух соседних точек возврата на ограничения.
8. Условием окончания поиска при ограничениях типа  $F(x) \geq 0$  является достижение экстремума исходного критерия (условие достижения зависит от используемого метода, при градиентных методах — это малость модуля градиента), если решение находится внутри допустимой области, и близость двух соседних точек нарушения допустимой области, если решение находится на границе допустимой области.
9. При одновременном нарушении двух и более ограничений возврат в допустимую область осуществляется по сумме (векторной!) градиентов нарушенных ограничений.
10. Низкой эффективностью будет в тех случаях, когда градиент критерия близок к нормали к поверхности ограничений (в двухмерном случае — когда линия ограничения примерно ортогональна к градиенту), так как в этом случае будет очень малое продвижение вдоль ограничения при движении по градиенту исходного критерия.

### *Метод проектирования градиента*

1. Проектируется градиент исходного критерия оптимальности на поверхность, касательную к ограничению.
2. Градиент исходного критерия оптимальности проектируется на поверхность, касательную к ограничению.
3. Градиент ограничения  $f(x)$  никуда не проектируется.
4. Направление касательной задает градиент  $\text{grad } f(x_1, x_2)$ .

5. Направление нормали к функции ограничения  $f(x) = 0$  в точке  $(x_1^i, x_2^i)$  задается выражением  $x_2 - x_2^i = \partial f / \partial x_2 (x_1 - x_1^i)$ .
6. Допустимый “коридор” вводится для обеспечения движения вдоль ограничения.
7. Метод наиболее эффективен в тех случаях, когда градиент критерия близок к нормали к поверхности ограничений (в двухмерном случае — когда линия ограничения примерно ортогональна к градиенту), так как в этом случае будет очень большое продвижение вдоль ограничения при движении по проекции градиента исходного критерия.
8. Внутри допустимого “коридора” поиск всегда осуществляется по касательной к поверхности ограничения в направлении, которое задает проекция градиента на эту поверхность.
9. В трехмерной задаче ограничение  $f(x_1, x_2, x_3) = 0$  представляет собой поверхность, касательной к которой будет плоскость, вдоль которой великое множество направлений, и без знания проекции градиента выбрать нужное не представляется возможным. В двумерном же случае ограничение будет представлять собой линию, касательной будет прямая линия, и вдоль нее и надо осуществлять поиск; в этом случае можно и не проектировать градиент.
10. Внутри допустимой области направление поиска определяется градиентом критерия оптимальности без учета ограничений.

## **ТЕХНОЛОГИЯ ВЫЧИСЛЕНИЙ**

---

Рассмотренные в книге методы полезны, конечно, не только для чисто теоретического изучения, но и для решения практических задач. В настоящее время имеется немало возможностей для реализации вычислений как в автоматическом режиме по готовым программам, так и путем самостоятельного программирования методов. Многие ранее разработанные алгоритмы на различных языках высокого уровня сейчас трудно применить, так как они реализованы в виде самостоятельных отдельных программ или процедур, которые необходимо куда-то вставлять. Поэтому в данном пособии приведены некоторые алгоритмы, реализующие рассмотренные методы, для распространенного программного средства, присутствующего почти в любом компьютере, — Microsoft Excel. Возможно, что это не самое лучшее

средство для вычислительных работ, но другие программные продукты, такие, как, например, MathCad, MathLab, Mathematica и др., специально ориентированные для математических расчетов, имеются далеко не в каждом компьютере и пока не могут применяться массово.

Приведенные в данном разделе программы выполнены в виде макросов на Visual Basic для Excel 7.0. Они не охватывают все методы, рассмотренные в книге, прежде всего по причине ее ограниченного объема. Представленные алгоритмы достаточно просты, выполнены без “программистских ухищрений” в надежде на самостоятельное знакомство с ними с целью изучения техники их написания и последующего применения в других алгоритмах, но предполагают некоторое предварительное знакомство с используемым языком программирования. Все программы построены примерно в одном стиле и состоят из нескольких самостоятельных макросов, управляющих вводом исходной информации и вычислениями по различным алгоритмам. В большинстве случаев предусмотрено наряду с числовым графическое представление результатов решения (например, при решении дифференциальных уравнений) или исходных данных (например, подынтегральной функции при вычислении интеграла, нелинейной функции при отыскании ее корней). После ввода исходных данных их можно корректировать, заменять, а потом в любое время решать поставленную задачу. Все приведенные программы объединены в отдельные тематические модули. В каждом модуле имеются свои начальные запускаемые макросы (Нелин\_уравнение\_исходные\_данные, Диффур\_исходные\_данные, Интеграл\_исходные\_данные), которые подготавливают в необходимой форме все исходные данные. Последующие макросы вызываются кнопками управления автоматически.

Приведенные в книге программы не охватывают все теоретически изложенные методы. Многие из нерассмотренных методов уже реализованы в Excel и не требуют каких-либо специальных усилий (кроме знакомства с принципами работы Excel). Это относится к аппроксимации, решению систем линейных уравнений, задачам оптимизации. Следует отметить, что и нелинейные уравнения могут решаться в Excel без предварительной подготовки

какого-либо макроса, и этой возможностью следует пользоваться. Но нетрудно убедиться, что далеко не всегда при этом можно получить нужные результаты: если уравнение  $f(x) = 0$  имеет несколько корней, то найден будет только один, причем иногда лежащий за желаемыми пределами интервала по  $x$ . Приведенные же в книге программы позволяют задавать интересующие границы расположения корня, визуально наблюдать график функции в заданном диапазоне, т.е. они расширяют типовые функции Excel. Задачи оптимизации неплохо решаются в Excel, поэтому при выборе алгоритмов для программной реализации в книге, в связи с ограниченностью ее объема, они не приведены. \*

В качестве примеров решения задач вычислительного характера на Excel рекомендуем познакомиться с [11, глава 8], где, в частности, наряду с описанием особенностей работы с этим программным продуктом рассматриваются алгоритмы решения дифференциальных уравнений и некоторых задач линейного и нелинейного программирования типовыми средствами Excel, а также [14], где очень подробно и доступно студенту с начальной подготовкой рассматриваются особенности и приемы решения различных оптимизационных задач: линейного, целочисленного, нелинейного, стохастического программирования, а также задач многопараметрической оптимизации. Помимо общих приемов решения таких задач приводится и множество конкретных примеров решения задач оптимального распределения ресурсов и оптимального проектирования.

## Модуль 1. Решение нелинейных уравнений

```
Public Function ГрафикF(sled)
    ActiveWindow.ScrollColumn = 1
    ActiveWindow.ScrollRow = 1
    ActiveSheet.ChartObjects.
        Add(6,2; 15; 223,4; 150,1).Select
    Application.CutCopyMode = False
    ActiveChart.ChartWizard Source:=Range(sled); _
        Gallery:=xlXYScatter; Format:=6; _
        PlotBy:=xlColumns; _
        CategoryLabels:=1; SeriesLabels:=0; HasLegend:=2; _
        Title:="График функции f(x)"; _
        CategoryTitle:="x"; ValueTitle:="f(x)"; _
        ExtraTitle:=""
```

```
Range("A1").Select
End Function
```

```
' Вычисление функции f(x)
```

```
Public Function FuncNU(Xt As Double)
    Range("A1").Select
    ActiveCell.Offset(1; 10).FormulaR1C1 = Xt
    FuncNU = ActiveCell.Offset(3; 7).Value
End Function
```

```
' Вычисление корней методом сканирования
```

```
Sub scan()
    Dim Xt As Double
    Dim Xt1 As Double
    Dim Xn As Double
    Dim Xk As Double
    Dim Eps As Double
    Dim element As Variant
    Dim myArray() As Variant
    ReDim myArray(0)
    Range("A1").Select
    ' Xn = "H6"
    Xn = ActiveCell.Offset(5; 7).Value
    ' Xk = "H7"
    Xk = ActiveCell.Offset(6; 7).Value
    inputVal = Application.InputBox("Введите " & _
        "относительную погрешность решения по X в %" & _
        "от диапазона X на графике в пределах 10..0,01")
    Eps = inputVal / 100
    Xt = Xn
    Yt = FuncI(Xt)
    If Yt = 0 Then
        myArray(UBound(myArray)) = Xt
        ReDim Preserve myArray(UBound(myArray) + 1)
        myArray(UBound(myArray)) = "?"
    End If
    Nx = 1 / Eps
    If Nx < 10 Then
        Nx = 10
        Eps = 1 / Nx * 100
    End If
    If Nx > 10000 Then
        Nx = 10000
        Eps = 1 / Nx
    End If
    Hx = (Xk - Xn) * Eps
    If Nx > 20 Then StepScan = 5 Else StepScan = 1
    For i = 1 To Nx / StepScan
```

```

Xt = Xn + i * Hx * StepScan
' промежуточный вывод
ActiveCell.Offset(8; 7).FormulaR1C1 = Xt
Yt1 = FuncI(Xt)
If (Yt * Yt1 < 0) Or (Yt1 = 0) Then
    Xt = Xt - Hx * StepScan
    Yt3 = Yt
    For j = 1 To StepScan
        Xt1 = Xt + j * Hx
        Yt2 = FuncI(Xt1)
        If (Yt3 * Yt2 < 0) Or (Yt2 = 0) Then
            myArray(UBound(myArray)) = Xt1 - Eps / 2
            ReDim Preserve myArray(UBound(myArray) + 1)
            myArray(UBound(myArray)) = "?"
        End If
        Yt3 = Yt2
    Next j
End If
Yt = Yt1
Next i
' вывод
ActiveCell.Offset(10; 5).FormulaR1C1 = " Погрешность "
ActiveCell.Offset(10; 7).Select
Selection.NumberFormat = "0,00"
ActiveCell.Offset(0; 0).FormulaR1C1 = Eps * 100
ActiveCell.Offset(0; 1).FormulaR1C1 = "%"
Range("A1").Select
ActiveCell.Offset(10; 5).Range("A1:D1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Selection.Interior.ColorIndex = 4
Selection.Font.ColorIndex = 3
' Вывод корней на панель и в таблицу в столбики N и O
i = 1
Range("A1").Select
ActiveCell.Offset(i + 1; 13).FormulaR1C1 = _
    "Корни уравнения"
For Each element In myArray
    If element <> "?" Then
        MsgBox "Корень " & i & "=" & element; vbOKOnly; _
            "Корни уравнения"
        ActiveCell.Offset(i + 2; 13).FormulaR1C1 = i
        ActiveCell.Offset(i + 2; 14).FormulaR1C1 = element
        i = i + 1
    End If
Next
End Sub

```

'Вычисление корня методом хорд

Sub Root1()

Dim Xt As Double

Dim Xt1 As Double

Dim Xn As Double

Dim Ya As Double

Dim Yb As Double

Dim Xk As Double

Dim Eps As Double

Dim right As Boolean

Range("A1").Select

' Xn = "H6"

Xn = ActiveCell.Offset(5; 7).Value

' Xk = "H7"

Xk = ActiveCell.Offset(6; 7).Value

ActiveCell.Offset(10; 5).FormulaR1C1 = " " "

ActiveCell.Offset(10; 7).FormulaR1C1 = " " "

ActiveCell.Offset(9; 7).FormulaR1C1 = " " "

ActiveCell.Offset(9; 5).FormulaR1C1 = " " "

inputVal = Application.InputBox("Введите " & \_  
"абсолютную погрешность решения по f(x)")

Eps = inputVal

Yt = FuncI(Xn)

Xt = Xn

i = 0

right = True

While (Abs(Yt) > Eps) And (i < 1000)

Yb = FuncI(Xk)

Ya = FuncI(Xn)

Yt = FuncI(Xt)

i = i + 1

If right Then

dY = Yb - Yt

If Abs(dY) > 0,0000000001 Then

Xt = Xt - FuncI(Xt) \* (Xk - Xt) / dY

Yt = FuncI(Xt)

If (Yt \* Yb > 0) Then

right = False

Xk = Xt

GoTo Metka

End If

End If

If Not right Then

dY = Yt - Ya

If Abs(dY) > 0,0000000001 Then

Xt = Xt - FuncI(Xt) \* (Xt - Xn) / dY

```

    Yt = FuncI(Xt)
    If (Yt * Ya > 0) Then
        right = True
        Xn = Xt
    End If
End If
' промежуточный вывод
Metka: ActiveCell.Offset(8; 7).FormulaR1C1 = Xt
Wend
' вывод
ActiveCell.Offset(10; 5).FormulaR1C1 = _
    "Погрешность f(X) "
ActiveCell.Offset(10; 7).Select
Selection.NumberFormat = "0,000000"
ActiveCell.Offset(0; 0).FormulaR1C1 = Eps
Range("A1").Select
ActiveCell.Offset(9; 7).FormulaR1C1 = i
ActiveCell.Offset(9; 5).FormulaR1C1 = _
    "Кол-во итераций"
ActiveCell.Offset(9; 5).Range("A1:D2").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Selection.Interior.ColorIndex = 4
Selection.Font.ColorIndex = 3
' Вывод корня в таблицу в столбики N и O
Range("A1").Select
ActiveCell.Offset(3; 13).FormulaR1C1 = _
    "Корень уравнения"
ActiveCell.Offset(4; 13).FormulaR1C1 = "X="
ActiveCell.Offset(4; 14).FormulaR1C1 = Xt
ActiveCell.Offset(5; 13).FormulaR1C1 = "f(x)="
ActiveCell.Offset(5; 14).FormulaR1C1 = Yt
End Sub

```

*' Построение графика функции*

```

Public Function GraficNU()
    Dim Xt As Double
    Dim Xn As Double
    Range("A1").Select
    ' X0="H6"
    Xn = ActiveCell.Offset(5; 7).Value
    ' Xk="H7"
    Xk = ActiveCell.Offset(6; 7).Value
    ' Nx="L2"
    Nx = ActiveCell.Offset(1; 11).Value
    Xt = Xn

```

```

i = 0
j = 0
Ypred = FuncNU(Xn)
Range("A1").Select
ActiveCell.Offset(i + 4; 10).FormulaR1C1 = Xn
ActiveCell.Offset(i + 4; 11).FormulaR1C1 = Ypred
If Ypred = 0 Then j = j + 1
Hx = (Xk - Xn) / Nx
Xt = Xn + Hx
sled = "K5:L"
While Xt <= 1,001 * Xk
    Yt = FuncNU(Xt)
    If (Yt * Ypred < 0) Or (Yt = 0) Then j = j + 1
    i = i + 1
    ActiveCell.Offset(i + 4; 10).FormulaR1C1 = Xt
    ActiveCell.Offset(i + 4; 11).FormulaR1C1 = Yt
    Xt = Xt + Hx
    Ypred = Yt
Wend
I_ = Str(i + 5)
Range("X1").Select
ActiveCell.FormulaR1C1 = I_
Range("Y1").Select
ActiveCell.FormulaR1C1 = "=TRIM(RC[-1])"
I_ = Selection.Value
sled = sled + I_
ГрафикF (sled)
Range("A1").Select
ActiveCell.Offset(9; 5).FormulaR1C1 = _
    "Количество корней"
ActiveCell.Offset(9; 7).FormulaR1C1 = j
End Function

Sub GraficFNU()
    Range("A1").Select
    ActiveCell.Offset(9; 7).FormulaR1C1 = "          "
    ActiveCell.Offset(10; 7).FormulaR1C1 = "          "
    ActiveSheet.ChartObjects.Select
    Selection.Delete
    GraficNU
    ActiveSheet.Buttons.Select
    Selection.Delete
    Keys
End Sub

Public Function Keys()
Range("A1").Select

```

*\* Nroot = "H9"*

Nroot = ActiveCell.Offset(9; 7).Value

If Nroot > 1 Then

ActiveSheet.Buttons.

Add(17,4; 173,4; 102,2; 20,4).Select

Selection.OnAction = "Scan"

Selection.Characters.Text = "Все корни"

With Selection.

Characters(Start:=1; Length:=12).Font

.Name = "Arial Cyr"

.FontStyle = "Regular"

.Size = 10

.Strikethrough = False

.Superscript = False

.Subscript = False

.OutlineFont = False

.Shadow = False

.Underline = xlNone

.ColorIndex = xlAutomatic

End With

End If

If Nroot = 1 Then

ActiveSheet.Buttons.

Add(120,4; 173,4; 102,2; 20,4).Select

Selection.OnAction = "Root1"

Selection.Characters.Text = "Корень в интервале"

With Selection.

Characters(Start:=1; Length:=17).Font

.Name = "Arial Cyr"

.FontStyle = "Regular"

.Size = 10

.Strikethrough = False

.Superscript = False

.Subscript = False

.OutlineFont = False

.Shadow = False

.Underline = xlNone

.ColorIndex = xlAutomatic

End With

End If

ActiveSheet.Buttons.

Add(240; 173,4; 164,4; 20,4).Select

Selection.OnAction = "GraficFNU"

Selection.Characters.Text =

"График после коррекции исх.дан."

```

With Selection.Characters(Start:=1; Length:=31).Font
    .Name = "Arial Cyr"
    .FontStyle = "Regular"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlNone
    .ColorIndex = xlAutomatic
End With
Range("A1").Select
End Function

```

```

Sub Нелин_уравнение_исходные_данные()
    Dim Xt As Double
    Dim Yt; Xn; Yn; Hx; Xk As Double
    Range("A1").Select
    ActiveCell.Offset(0; 1).FormulaR1C1 = _
        "      Решение нелинейного уравнения " & _
        "вида f(x) = 0 "
    ActiveCell.Offset(0; 1).Range("A1:E1").Select
    Selection.Font.ColorIndex = 3
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic
    With Selection.Interior
        .ColorIndex = 19
        .Pattern = xlSolid
    End With
    Range("A1").Select
    ' Присвоение переменной K2 имени x
    ActiveWorkbook.Names.Add Name:="x"; _
        RefersToR1C1:="-R2C11"
    ' Организация ввода исходных данных
    inputfunc = Application. _
        InputBox("Введите функцию f(x)")
    Xf = "=" & inputfunc
    inputVal = Application. _
        InputBox("Введите начало интервала по X - X0")
    Xn = inputVal
    inputVal = Application. _
        InputBox("Введите конечное значение по X - Xk")

```

```

Xk = inputVal
inputVal = Application. _
    InputBox("Введите желаемое число точек для " & _
        "графика Nx")
Nx = inputVal
If Nx < 0 Then Nx = 40
If Nx > 300 Then Nx = 300
Range("A1").Select
    ' "G4" = "f(x,y)"
ActiveCell.Offset(3; 6).FormulaR1C1 = "f(x)"
    ' "F5" = "Начальные условия"
ActiveCell.Offset(4; 5).FormulaR1C1 = _
    "Граничные условия"
    ' "G6" = "X0"
ActiveCell.Offset(5; 6).FormulaR1C1 = "X0"
    ' "G7" = "Y0"
ActiveCell.Offset(6; 6).FormulaR1C1 = "Xk"
ActiveCell.Offset(8; 5).FormulaR1C1 = _
    "Текущ. приближ."
ActiveCell.Offset(7; 5).FormulaR1C1 = _
    "Точки для графика"
ActiveCell.Offset(0; 10).FormulaR1C1 = "X текущ."
ActiveCell.Offset(0; 11).FormulaR1C1 = _
    "Nx - число точек"
ActiveCell.Offset(1; 11).FormulaR1C1 = Nx
ActiveCell.Offset(2; 10).FormulaR1C1 = "Функция f(x)"
ActiveCell.Offset(3; 10).FormulaR1C1 = "    X"
ActiveCell.Offset(3; 11).FormulaR1C1 = "    f(x)"
ActiveCell.Offset(9; 5).FormulaR1C1 = "    "
ActiveCell.Offset(10; 5).FormulaR1C1 = "    "
ActiveCell.Offset(9; 7).FormulaR1C1 = "    "
ActiveCell.Offset(10; 7).FormulaR1C1 = "    "
ActiveCell.Offset(0; 8).Columns("A:A"). _
    EntireColumn.ColumnWidth = 3,11
ActiveCell.Offset(2; 5).Range("A1:D10").Select
With Selection.Interior
    .ColorIndex = 8
    .Pattern = xlSolid
End With
ActiveCell.Offset(0; 0).Range("A1:D10").Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic

```

```

Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Range("A1").Select
ActiveCell.Offset(3; 7).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(2; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(2; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(1; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(-2; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Borders(xlBottom)
    .Weight = xlMedium
    .ColorIndex = xlAutomatic
End With
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(0; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Xk

```

```

ActiveCell.Offset(-1; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Xn
ActiveCell.Offset(-2; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Xf
ActiveCell.Offset(4; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Nx
ActiveCell.Offset(1; 0).Select
Selection.Interior.ColorIndex = 2
GraficNU
Keys
Range("A1").Select
End Sub

```

## Модуль 2. Решение дифференциальных уравнений

```

Public Function ГрафикF(sled)
ActiveSheet.ChartObjects.Select
Selection.Delete
ActiveWindow.ScrollColumn = 1
ActiveWindow.ScrollRow = 1
ActiveSheet.ChartObjects. _
    Add(6,22; 15; 223,4; 150,1).Select
Application.CutCopyMode = False
ActiveChart.ChartWizard Source:=Range(sled); _
    Gallery:=xlXYScatter; Format:=6; _
    PlotBy:=xlColumns; CategoryLabels:=1; _
    SeriesLabels:=0; HasLegend:=2; _
    Title:="График решения y(x)"; _
    CategoryTitle:="x"; ValueTitle:="y(x)"; _
    ExtraTitle:=""
End Function

```

### *'Вычисление правой стороны диффура*

```

Public Function Func(Xt As Double; Yt As Double)
Range("A1").Select
ActiveCell.Offset(1; 10).FormulaR1C1 = Xt
ActiveCell.Offset(1; 11).FormulaR1C1 = Yt
Func = ActiveCell.Offset(3; 7).Value
End Function

```

### *'Метод Рунге — Кутта*

```

Sub Runge_kutta()
' Этот макрос решает диффура методом Рунге — Кутта
' Возможен запуск с кнопки (после макроса Диффура_исходные_данные)
' или вызовом после подготовки исходных данных вручную

```

- ' *Исходные данные:*
- ' *функция  $f(x,y)$  - Н4*
- ' *Начальные условия по X - Н6, по Y - Н7*
- ' *Конечное значение X - Н8*
- ' *Шаг решения h - Н9*
- ' *Текущее X - К2, Y - L2*
- ' *Переменная К2 имеет имя x*
- ' *Переменная L2 имеет имя y*
- ' *Результаты: график + массив x(К5: ) и Y(L5: )*

```

Dim Xt As Double
Dim Yt As Double
Range("A1").Select
' Yn = "Н7"
Yn = ActiveCell.Offset(6; 7).Value
' Xn = "Н6"
Xn = ActiveCell.Offset(5; 7).Value
' Hx = "Н9"
Hx = ActiveCell.Offset(8; 7).Value
' Xk = "Н8"
Xk = ActiveCell.Offset(7; 7).Value
Xt = Xn
Yt = Yn
i = 0
sled = "К5:L"
ActiveCell.Offset(i + 4; 10).FormulaR1C1 = Xt
ActiveCell.Offset(i + 4; 11).FormulaR1C1 = Yt
While Xt < Xk
    k_1 = Hx * Func(Xt; Yt)
    k_2 = Hx * Func(Xt + Hx / 2; Yt + k_1 / 2)
    k_3 = Hx * Func(Xt + Hx / 2; Yt + k_2 / 2)
    k_4 = Hx * Func(Xt + Hx; Yt + k_3)
    Yt = Yt + (k_1 + 2 * k_2 + 2 * k_3 + k_4) / 6
    Xt = Xt + Hx
    ActiveCell.Offset(i + 5; 10).FormulaR1C1 = Xt
    ActiveCell.Offset(i + 5; 11).FormulaR1C1 = Yt
    i = i + 1
Wend
I_ = Str(i + 5)
ActiveCell.Offset(0; 30).FormulaR1C1 = I_
ActiveCell.Offset(0; 31).FormulaR1C1 = "=TRIM(RC[-1])"
I_ = ActiveCell.Offset(0; 31).Value
sled = sled + I_
ГрафикF (sled)
End Sub

```

'Метод Эйлера

```
Sub Eyler()  
  ' Этот макрос решает диффур методом Эйлера  
  ' Возможен запуск с кнопки (после макроса Диффур_исходные_данные)  
  ' или вызовом после подготовки исходных данных вручную  
  ' Исходные данные при инициализации от A1:  
  ' функция f(x,y) - H4  
  ' Начальные условия по X - H6, по Y - H7  
  ' Конечное значение X - H8  
  ' Шаг решения h - H9  
  ' Текущее x - K2, Y - L2  
  ' Переменная K2 имеет имя x  
  ' Переменная L2 имеет имя y  
  ' Результаты: график + массив x(K5: ) и Y(L5: )  
  Dim Xt As Double  
  Dim Yt As Double  
  Range("A1").Select  
  ' Yn="H7"  
  Yn = ActiveCell.Offset(6; 7).Value  
  ' Xn="H6"  
  Xn = ActiveCell.Offset(5; 7).Value  
  ' Hx="H9"  
  Hx = ActiveCell.Offset(8; 7).Value  
  ' Xk="H8"  
  Xk = ActiveCell.Offset(7; 7).Value  
  Xt = Xn  
  Yt = Yn  
  i = 0  
  sled = "K5:L"  
  ActiveCell.Offset(i + 4; 10).FormulaR1C1 = Xt  
  ActiveCell.Offset(i + 4; 11).FormulaR1C1 = Yt  
  While Xt < Xk  
    Yt = Yt + Hx * Func(Xt; Yt)  
    Xt = Xt + Hx  
    ActiveCell.Offset(i + 5; 10).FormulaR1C1 = Xt  
    ActiveCell.Offset(i + 5; 11).FormulaR1C1 = Yt  
    i = i + 1  
  Wend  
  I_ = Str(i + 5)  
  ActiveCell.Offset(0; 30).FormulaR1C1 = I_  
  ActiveCell.Offset(0; 31).FormulaR1C1 = "=TRIM(RC[-1])"  
  I_ = ActiveCell.Offset(0; 31).Value  
  sled = sled + I_  
  ГрафикF (sled)  
End Sub
```

```

Sub Диффур_исходные_данные()
    Dim Xt As Double
    Dim Yt; Xn; Yn; Hx; Xk As Double
    Range("A1").Select
    ActiveCell.Offset(0; 1).
        FormulaR1C1 = "    Решение " &
        "дифференциального уравнения  $dy/dx=f(x,y)$ "
    ActiveCell.Offset(0; 1).Range("A1:F1").Select
    Selection.Font.ColorIndex = 3
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.BorderAround Weight:=xlMedium;
        ColorIndex:=xlAutomatic
    With Selection.Interior
        .ColorIndex = 19
        .Pattern = xlSolid
    End With
    Range("A1").Select
    ' Присвоение переменной K2 имени x
    ActiveWorkbook.Names.Add Name="x";
    RefersToR1C1:="=R2C11"
    ' Присвоение переменной L2 имени Y
    ActiveWorkbook.Names.Add Name="y";
    RefersToR1C1:="=R2C12"
    ' Организация ввода исходных данных
    inputfunc = Application.
        InputBox("Введите функцию f(x,y)")
    Xf = "=" & inputfunc
    inputVal = Application.
        InputBox("Введите начальные условия по X - X0")
    Xn = inputVal
    inputVal = Application.
        InputBox("Введите начальные условия по Y - Y0")
    Yn = inputVal
    inputVal = Application.
        InputBox("Введите конечное значение по X - Xk")
    Xk = inputVal
    inputVal = Application.
        InputBox("Введите шаг решения по X")
    Hx = inputVal
    ' "G4" = "f(x,y)"
    ActiveCell.Offset(3; 6).FormulaR1C1 = "f(x,y)"
    ' "F5" = "Начальные условия"

```

```

ActiveCell.Offset(4; 5).FormulaR1C1 = _
    "Начальные условия"
    ' "G6"="X0"
ActiveCell.Offset(5; 6).FormulaR1C1 = "X0"
    ' "G7"="Y0"
ActiveCell.Offset(6; 6).FormulaR1C1 = "Y0"
    ' "F8"="Конечное знач. Xk"
ActiveCell.Offset(7; 5).FormulaR1C1 = _
    "Конечное знач. Xk"
    ' "F9"="Шаг решения h"
ActiveCell.Offset(8; 5).FormulaR1C1 = "Шаг решения h"
ActiveCell.Offset(0; 10).FormulaR1C1 = "X текущ."
ActiveCell.Offset(0; 11).FormulaR1C1 = "Y текущ."
ActiveCell.Offset(2; 10).FormulaR1C1 = "Р Е Ш Е Н И Е"
ActiveCell.Offset(3; 10).FormulaR1C1 = " X"
ActiveCell.Offset(3; 11).FormulaR1C1 = " Y"
ActiveCell.Offset(0; 9).Columns("A:A"). _
    EntireColumn.ColumnWidth = 3,11
ActiveCell.Offset(2; 5).Range("A1:D8").Select
With Selection.Interior
    .ColorIndex = 8
    .Pattern = xlSolid
End With
ActiveCell.Offset(0; 0).Range("A1:D8").Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Range("A1").Select
ActiveCell.Offset(3; 7).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(2; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone

```

```

Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(3; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(-2; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(1; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
With Selection.Borders(xlBottom)
    .Weight = xlMedium
    .ColorIndex = xlAutomatic
End With
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(-1; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Yn
ActiveCell.Offset(-1; 0).Select
Selection.Interior.ColorIndex = 2

ActiveCell.FormulaR1C1 = Xn
ActiveCell.Offset(-2; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Xf
ActiveCell.Offset(5; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Hx
ActiveCell.Offset(-1; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Xk
ActiveSheet.Buttons. _
    Add(47,4; 173,4; 102,2; 20,4).Select
Selection.OnAction = "Eyler"
Selection.Characters.Text = "Метод Эйлера"

```

```

With Selection.Characters(Start:=1; Length:=12).Font
    .Name = "Arial Cyr"
    .FontStyle = "Regular"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlNone
    .ColorIndex = xlAutomatic
End With
ActiveSheet.Buttons. _
    Add(232,4; 173,4; 102,2; 20,4).Select
Selection.OnAction = "Runge_kutta"
Selection.Characters.Text = "Метод Рунге-Кутты"
With Selection.Characters(Start:=1; Length:=17).Font
    .Name = "Arial Cyr"
    .FontStyle = "Regular"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlNone
    .ColorIndex = xlAutomatic
End With
Range("A1").Select
End Sub

```

### Модуль 3. Вычисление интегралов

```

Public Function ГрафикF(sled)
    ActiveWindow.ScrollColumn = 1
    ActiveWindow.ScrollRow = 1
    ActiveSheet.ChartObjects.
        Add(6,2; 15; 223,4; 150,1).Select
    Application.CutCopyMode = False
    ActiveChart.ChartWizard Source:=Range(sled); _
        Gallery:=xlXYScatter; Format:=6; _
        PlotBy:=xlColumns; CategoryLabels:=1; _
        SeriesLabels:=0; HasLegend:=2; _
        Title:="График функции f(x)"; CategoryTitle:="x"; _
        ValueTitle:="f(x)"; ExtraTitle:=""

```

```
Range("A1").Select
End Function
```

*' Вычисление подынтегральной функции*

```
Public Function FuncI(Xt As Double)
    Range("A1").Select
    ActiveCell.Offset(1; 10).FormulaR1C1 = Xt
    FuncI = ActiveCell.Offset(3; 7).Value
End Function
```

*' Вычисление интеграла методом Симпсона*

```
Sub Simp()
    Dim Xt As Double
    Dim Xn As Double
    Dim Xk As Double
    Range("A1").Select
    ' X0="H6"
    Xn = ActiveCell.Offset(5; 7).Value
    ' Xk="H7"
    Xk = ActiveCell.Offset(6; 7).Value
    ' Nx="H8"
    Nx = ActiveCell.Offset(7; 7).Value
    tr = 0
    Hx = (Xk - Xn) / Nx
    For i = 1 To Nx
        Xt = Xn + (i - 1) * Hx
        Yt = FuncI(Xt)
        Yt1 = FuncI(Xt + Hx / 2)
        Yt2 = FuncI(Xt + Hx)
        tr = tr + (Yt + 4 * Yt1 + Yt2)
    Next i
    tr = Hx / 2 * tr / 3
    ' Расчет с шагом Hx/2
    tr1 = 0
    Hx = (Xk - Xn) / Nx / 2
    For i = 1 To Nx * 2
        Xt = Xn + (i - 1) * Hx
        Yt = FuncI(Xt)
        Yt1 = FuncI(Xt + Hx / 2)
        Yt2 = FuncI(Xt + Hx)
        tr1 = tr1 + (Yt + 4 * Yt1 + Yt2)
    Next i
    tr1 = Hx / 2 * tr1 / 3
    ' Расчет погрешности
    tr = Abs(tr - tr1) / 15
    ' вывод
    ActiveCell.Offset(9; 5).FormulaR1C1 = "      Интеграл ="
```

```

ActiveCell.Offset(9; 7).FormulaR1C1 = tr1
ActiveCell.Offset(10; 5).FormulaR1C1 = "Погрешность <"
ActiveCell.Offset(10; 7).FormulaR1C1 = tr
ActiveCell.Offset(9; 5).Range("A1:D1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Selection.Interior.ColorIndex = 4
Selection.Font.ColorIndex = 3
ActiveCell.Offset(1; 0).Range("A1:D1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Selection.Interior.ColorIndex = 4
Selection.Font.ColorIndex = 3
ActiveCell.Offset(-8; -5).Range("A1").Select
End Sub

```

*' Вычисление интеграла методом трапеций*

```

Sub Trap()
    Dim Xt As Double
    Dim Xn As Double
    Dim Xk As Double
    Range("A1").Select
    ' X0 = "H6"
    Xn = ActiveCell.Offset(5; 7).Value
    ' Xk = "H7"
    Xk = ActiveCell.Offset(6; 7).Value
    ' Nx = "H8"
    Nx = ActiveCell.Offset(7; 7).Value
    ' вычисление с шагом Hx
    tr = 0
    Hx = (Xk - Xn) / Nx
    Xt = Xn
    While Xt < Xk
        Yt = FuncI(Xt)
        Yt1 = FuncI(Xt + Hx)
        tr = tr + Hx * (Yt + Yt1) / 2
        Xt = Xt + Hx
    Wend
    ' вычисление с шагом Hx/2
    tr1 = 0
    Hx = (Xk - Xn) / Nx / 2
    Xt = Xn
    While Xt < Xk
        Yt = FuncI(Xt)
        Yt1 = FuncI(Xt + Hx)
        tr1 = tr1 + Hx * (Yt + Yt1) / 2
        Xt = Xt + Hx
    Wend

```

```

' Расчет погрешности
tr = Abs(tr - tr1) / 3
' вывод
ActiveCell.Offset(9; 5).FormulaR1C1 = "    Интеграл ="
ActiveCell.Offset(9; 7).FormulaR1C1 = tr1
ActiveCell.Offset(10; 5).FormulaR1C1 = _
    "    Погрешность <"
ActiveCell.Offset(10; 7).FormulaR1C1 = tr
ActiveCell.Offset(9; 5).Range("A1:D1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Selection.Interior.ColorIndex = 4
Selection.Font.ColorIndex = 3
ActiveCell.Offset(1; 0).Range("A1:D1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Selection.Interior.ColorIndex = 4
Selection.Font.ColorIndex = 3
ActiveCell.Offset(-8; -5).Range("A1").Select
End Sub

Public Function Grafic()
Dim Xt As Double
Range("A1").Select
' X0="H6"
Xn = ActiveCell.Offset(5; 7).Value
' Xk="H7"
Xk = ActiveCell.Offset(6; 7).Value
' Nx="H8"
Nx = ActiveCell.Offset(7; 7).Value
Xt = Xn
i = 0
Hx = (Xk - Xn) / Nx
sled = "K5:L"
While Xt <= Xk
    Yt = FuncI(Xt)
    Range("A1").Select
    ActiveCell.Offset(i + 4; 10).FormulaR1C1 = Xt
    ActiveCell.Offset(i + 4; 11).FormulaR1C1 = Yt
    Xt = Xt + Hx
    i = i + 1
Wend
I_ = Str(i + 4)
I_ = Trim(I_)
sled = sled + I_
ГрафикF (sled)
End Function

```

```

Sub GraficF()
    Range("A1").Select
    ActiveCell.Offset(9; 7).FormulaR1C1 = _
        "
    ActiveCell.Offset(10; 7).FormulaR1C1 = _
        "
    ActiveSheet.ChartObjects.Select
    Selection.Delete
    Grafic
End Sub

Sub Интеграл_исходные_данные()
    Dim Xt As Double
    Dim Yt; Xn; Yn; Hx; Xk As Double
    Range("A1").Select
    ActiveCell.Offset(0; 1).FormulaR1C1 = _
        "      Вычисление интеграла от функции      f(x) "
    ActiveCell.Offset(0; 1).Range("A1:E1").Select
    Selection.Font.ColorIndex = 3
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic
    With Selection.Interior
        .ColorIndex = 19
        .Pattern = xlSolid
    End With
    Range("A1").Select
    ' Присвоение переменной K2 имени x
    ActiveWorkbook.Names.Add Name:="x"; _
        RefersToR1C1:="=R2C11"
    ' Организация ввода исходных данных
    inputfunc = Application. _
        InputBox("Введите функцию f(x)")
    Xf = "=" & inputfunc
    inputVal = Application. _
        InputBox("Введите начало интервала по X - X0")
    Xn = inputVal
    inputVal = Application. _
        InputBox("Введите конечное значение по X - Xk")
    Xk = inputVal
    inputInt = Application. _
        InputBox("Введите число шагов по X")
    Nx = inputInt

```

```

If Nx < 1 Then Nx = 10
If Nx > 100 Then Nx = 100
Range("A1").Select
    ' "G4"="f(x,y)"
ActiveCell.Offset(3; 6).FormulaR1C1 = "f(x)"
    ' "F5"="Начальные условия"
ActiveCell.Offset(4; 5).FormulaR1C1 = _
    "Граничные условия"
    ' "G6"="X0"
ActiveCell.Offset(5; 6).FormulaR1C1 = "X0"
    ' "G7"="Y0"
ActiveCell.Offset(6; 6).FormulaR1C1 = "Xk"
ActiveCell.Offset(0; 10).FormulaR1C1 = "X текущ."
ActiveCell.Offset(2; 10).FormulaR1C1 = "Функция f(x)"
ActiveCell.Offset(3; 10).FormulaR1C1 = "    X"
ActiveCell.Offset(3; 11).FormulaR1C1 = "    f(x)"
ActiveCell.Offset(7; 5).FormulaR1C1 = "Число шагов Nx"
ActiveCell.Offset(9; 5).FormulaR1C1 = "    "
ActiveCell.Offset(10; 5).FormulaR1C1 = "    "
ActiveCell.Offset(9; 7).FormulaR1C1 = "    "
ActiveCell.Offset(10; 7).FormulaR1C1 = "    "
ActiveCell.Offset(0; 8).Columns("A:A"). _
    EntireColumn.ColumnWidth = 3,11
ActiveCell.Offset(2; 5).Range("A1:D10").Select
With Selection.Interior
    .ColorIndex = 8
    .Pattern = xlSolid
End With
ActiveCell.Offset(0; 0).Range("A1:D10").Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
Range("A1").Select
ActiveCell.Offset(3; 7).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic

```

```

ActiveCell.Offset(2; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(2; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(-1; 0).Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Borders(xlBottom)
    .Weight = xlMedium
    .ColorIndex = xlAutomatic
End With
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(0; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Xk
ActiveCell.Offset(-1; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Xn
ActiveCell.Offset(-2; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Xf
ActiveCell.Offset(4; 0).Select
Selection.Interior.ColorIndex = 2
ActiveCell.FormulaR1C1 = Nx
ActiveSheet.Buttons. _
    Add(17,4; 173,4; 102,2; 20,4).Select
Selection.OnAction = "Trap"
Selection.Characters.Text = "Метод трапеций"
With Selection.Characters(Start:=1; Length:=12).Font
    .Name = "Arial Cyr"
    .FontStyle = "Regular"

```

```

        .Size = 10
        .Strikethrough = False
        .Superscript = False
        .Subscript = False
        .OutlineFont = False
        .Shadow = False
        .Underline = xlNone
        .ColorIndex = xlAutomatic
End With
ActiveSheet.Buttons. _
    Add(120,4; 173,4; 102,2; 20,4).Select
Selection.OnAction = "Simp"
Selection.Characters.Text = "Метод Симпсона"
With Selection.Characters(Start:=1; Length:=17).Font
    .Name = "Arial Cyr"
    .FontStyle = "Regular"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlNone
    .ColorIndex = xlAutomatic
End With
ActiveSheet.Buttons. _
    Add(240; 173,4; 164,4; 20,4).Select
Selection.OnAction = "GraficF"
Selection.Characters.Text = _
    "График после коррекции исх.дан."
With Selection.Characters(Start:=1; Length:=31).Font
    .Name = "Arial Cyr"
    .FontStyle = "Regular"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlNone
    .ColorIndex = xlAutomatic
End With
Grafic
End Sub

```

## Модуль 4. Интерполяция

*' Удаление точки, на которую указывает курсор*

```
Sub Удалить_точку()  
    Dim X As Double  
    ActiveSheet.ChartObjects.Select  
    Selection.Delete  
    ActiveSheet.Buttons.Select  
    Selection.Delete  
    Ra = ActiveCell.Row  
    Co = ActiveCell.Column  
    Nx = ActiveCell.Offset(4 - Ra; 5 - Co).Value  
    If (Ra < 5) Or (Co > 2) Or ((Ra > 4 + Nx + 1)) Then  
        MsgBox ("Вы забыли указать удаляемую точку!"); _  
            vbOKOnly; "ВНИМАНИЕ!"  
        GoTo Metka3  
    End If  
    If Co = 2 Then Col = -1 Else Col = 0  
    ActiveCell.Offset(5 - Ra + Nx + 1; Col + 2). _  
        Range("A1").Select  
    Selection.ClearContents  
    Selection.Borders(xlLeft).LineStyle = xlNone  
    Selection.Borders(xlRight).LineStyle = xlNone  
    Selection.Borders(xlTop).LineStyle = xlNone  
    Selection.Borders(xlBottom).LineStyle = xlNone  
    Selection.BorderAround LineStyle:=xlNone  
    Selection.Interior.ColorIndex = xlNone  
    Range("A1").Select  
    For i = Ra To (5 + Nx + 2)  
        ActiveCell.Offset(i - 1; 0).FormulaR1C1 = _  
            ActiveCell.Offset(i; 0).Value  
        ActiveCell.Offset(i - 1; 1).FormulaR1C1 = _  
            ActiveCell.Offset(i; 1).Value  
    Next i  
    ActiveCell.Offset(5 + Nx; Col).Range("A1:C1").Select  
    Selection.Borders(xlLeft).LineStyle = xlNone  
    Selection.Borders(xlRight).LineStyle = xlNone  
    Selection.Borders(xlTop).LineStyle = xlNone  
    Selection.Borders(xlBottom).LineStyle = xlNone  
    Selection.BorderAround LineStyle:=xlNone  
    Selection.Interior.ColorIndex = xlNone  
    ActiveCell.Offset(-1; 0).Range("A1").Select  
    Selection.BorderAround Weight:=xlMedium; _  
        ColorIndex:=xlAutomatic  
    With Selection.Interior  
        .ColorIndex = 39  
        .Pattern = xlSolid  
    End With
```

```

ActiveCell.Offset(0; 2).Range("A1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Interior
    .ColorIndex = 39
    .Pattern = xlSolid
End With
ActiveCell.Offset(0; -1).Range("A1").Select
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround LineStyle:=xlNone
Selection.Interior.ColorIndex = xlNone
GoTo Metka3

```

Metka3:

```

Range("A1").Select
Nx = Nx - 1
ActiveCell.Offset(3; 4).FormulaR1C1 = Nx
i = 4 + Nx + 2
i_ = Str(i)
i_ = Trim(i_)
sled1 = "A5:C" + i_
i_ = Str(Nx + 1)
i_ = Trim(i_)
Range("A1").Select
sled = "=Lagrang_Qwik(R[-" & i_ & "]C[-2]:RC[-1])"
ActiveCell.Offset(5 + Nx; 2).FormulaR1C1 = sled
Graphic (sled1)
Key

```

End Sub

*' Вставка дополнительной точки перед той,*

*' на которую указывает курсор*

Sub вставка\_точки()

```

Dim X As Double
ActiveSheet.ChartObjects.Select
Selection.Delete
ActiveSheet.Buttons.Select
Selection.Delete
Ra = ActiveCell.Row
Co = ActiveCell.Column
Nx = ActiveCell.Offset(4 - Ra; 5 - Co).Value
If (Ra < 5) Or (Co > 2) Or ((Ra > 4 + Nx + 2)) Then
    MsgBox "Вы забыли указать место вставки " & _
        "новой точки!"; _
        vbOKOnly; "ВНИМАНИЕ!"
GoTo Metka3

```

```

End If
If Co = 2 Then Col = -1 Else Col = 0
ActiveCell.Offset(5 - Ra + Nx + 1; Col + 2). _
    Range("A1").Select
Selection.ClearContents
Selection.Borders(xlLeft).LineStyle = xlNone
Selection.Borders(xlRight).LineStyle = xlNone
Selection.Borders(xlTop).LineStyle = xlNone
Selection.Borders(xlBottom).LineStyle = xlNone
Selection.BorderAround LineStyle:=xlNone
Selection.Interior.ColorIndex = xlNone
Range("A1").Select
ActiveCell.Offset(4 + Nx + 2; 2).Range("A1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Interior
    .ColorIndex = 39
    .Pattern = xlSolid
End With
ActiveCell.Offset(0; -2).Range("A1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Interior
    .ColorIndex = 39
    .Pattern = xlSolid
End With
ActiveCell.Offset(-1; 0).Range("A1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Interior
    .ColorIndex = 40
    .Pattern = xlSolid
End With
ActiveCell.Offset(; 1).Range("A1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Interior
    .ColorIndex = 40
    .Pattern = xlSolid
End With
' Сдвиг элементов
Range("A1").Select
For i = 5 + Nx + 2 To Ra Step -1
    ActiveCell.Offset(i; 0).FormulaR1C1 = _
        ActiveCell.Offset(i - 1; 0).Value
    ActiveCell.Offset(i; 1).FormulaR1C1 = _
        ActiveCell.Offset(i - 1; 1).Value

```

```

Next i
Nx = Nx + 1
ActiveCell.Offset(Ra - 1; 0).ClearContents
ActiveCell.Offset(Ra - 1; 1).ClearContents
    ' Ввод XY
Метка2:
inputVal = Application.InputBox("Введите " & _
    "значение аргумента X, больше предыдущего " & _
    "и меньше последующего или для отмены " & _
    "ввода ESC/Отмена")
If inputVal = False Then
    Nx = Nx - 1
    For i = Ra To (5 + Nx + 2)
        ActiveCell.Offset(i - 1; 0).FormulaR1C1 = _
            ActiveCell.Offset(i; 0).Value
        ActiveCell.Offset(i - 1; 1).FormulaR1C1 = _
            ActiveCell.Offset(i; 1).Value
    Next i
    ActiveCell.Offset(5 + Nx + 1; Col). _
        Range("A1:C1").Select
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.BorderAround LineStyle:=xlNone
    Selection.Interior.ColorIndex = xlNone
    ActiveCell.Offset(-1; 0).Range("A1").Select
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic
    With Selection.Interior
        .ColorIndex = 39
        .Pattern = xlSolid
    End With
    ActiveCell.Offset(0; 2).Range("A1").Select
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic
    With Selection.Interior
        .ColorIndex = 39
        .Pattern = xlSolid
    End With
    ActiveCell.Offset(0; -1).Range("A1").Select
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.BorderAround LineStyle:=xlNone
    Selection.Interior.ColorIndex = xlNone

```

```

    GoTo Metka3
End If
ActiveCell.Offset(3; 4).FormulaR1C1 = Nx
X = inputVal
If Ra = 5 Then
    If (ActiveCell.Offset(Ra; 0).Value <= X) Then
        MsgBox "Значения X надо вводить " & _
            "по возрастанию. Введите другое " & _
            "значение, меньше первого"; _
            vbOKOnly; "ВНИМАНИЕ!"
        GoTo Metka2
    End If
End If
If Ra > 5 And Ra < 5 + Nx Then
    If (ActiveCell.Offset(Ra; 0).Value <= X) Then
        MsgBox "Значения X надо вводить " & _
            "по возрастанию. Введите другое " & _
            "значение, меньше следующего"; _
            vbOKOnly; "ВНИМАНИЕ!"
        GoTo Metka2
    End If
    If (ActiveCell.Offset(Ra - 2; 0).Value >= X) Then
        MsgBox "Значения X надо вводить " & _
            "по возрастанию. Введите другое " & _
            "значение, больше предыдущего"; _
            vbOKOnly; "ВНИМАНИЕ!"
        GoTo Metka2
    End If
End If
If Ra = 5 + Nx Then
    If (ActiveCell.Offset(Ra - 2; 0).Value >= X) Then
        MsgBox "Значения X надо вводить " & _
            "по возрастанию. Введите другое " & _
            "значение, больше предыдущего"; _
            vbOKOnly; "ВНИМАНИЕ!"
        GoTo Metka2
    End If
End If
If inputVal <> False Then
Metka1:
    inputVal1 = Application. _
        InputBox("Введите значение функции Y")
    Range("A1").Select
    ActiveCell.Offset(Ra; 0).Range("A1:B1").Select
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic
    ActiveCell.Offset(0; 1).Range("A1").Select

```

```

        Selection.BorderAround Weight:=xlMedium; _
            ColorIndex:=xlAutomatic
        ActiveCell.Offset(-1; -1).FormulaR1C1 = inputVal
        ActiveCell.Offset(-1; 0).FormulaR1C1 = inputVal1
        If inputVal1 = False Then
            GoTo Metka3
        End If
    End If
End Sub

Metka3:
    Range("A1").Select
    i = 4 + Nx + 2
    i_ = Str(i)
    i_ = Trim(i_)
    sled1 = "A5:C" + i_
    i_ = Str(Nx + 1)
    i_ = Trim(i_)
    Range("A1").Select
    sled = "=Lagrang_Qwik(R[-" & i_ & "]C[-2]:RC[-1])"
    ActiveCell.Offset(5 + Nx; 2).FormulaR1C1 = sled
    Graphic (sled1)
    Key
End Sub

```

*' Построение графика*

```

Function Graphic(sled)
    ActiveSheet.ChartObjects. _
        Add(156; 27; 238,2; 144).Select
    Application.CutCopyMode = False
    ActiveChart.ChartWizard Source:=Range(sled); _
        Gallery:=xlXYScatter; Format:=2; _
        PlotBy:=xlColumns; CategoryLabels:=1; _
        SeriesLabels:=0; HasLegend:=2; _
        Title:="График функции f(x) и ее " & _
            "восстановленное значение "; _
        CategoryTitle:="X"; ValueTitle:="f(X)"; _
        ExtraTitle:=""
    Range("A1").Select
End Function

```

*' Управляющие ключи*

```

Function Key()
    ActiveSheet.Buttons. _
        Add(168,2; 175,8; 100,2; 13,2).Select
    Selection.OnAction = "Вставка точки"
    Selection.Characters.Text = "Вставить точку "
    With Selection.Characters(Start:=1; Length:=38).Font
        .Name = "Arial Cyr"
        .FontStyle = "Regular"
    End With
End Function

```

```

        .Size = 10
        .Strikethrough = False
        .Superscript = False
        .Subscript = False
        .OutlineFont = False
        .Shadow = False
        .Underline = xlNone
        .ColorIndex = xlAutomatic
End With
ActiveCell.Select
Range("A1").Select
ActiveSheet.Buttons.
    Add(278,2; 175,8; 100,2; 13,2).Select
Selection.OnAction = "Удалить_точку"
Selection.Characters.Text = "Удалить точку "
With Selection.Characters(Start:=1; Length:=38).Font
    .Name = "Arial Cyr"
    .FontStyle = "Regular"
    .Size = 10
    .Strikethrough = False
    .Superscript = False
    .Subscript = False
    .OutlineFont = False
    .Shadow = False
    .Underline = xlNone
    .ColorIndex = xlAutomatic
End With
ActiveCell.Select
Range("A1").Select
End Function

Function Lagrang_Qwik(Rang)
    ' Расчет значения функции в заданной точке по методу Лагранжа
    Ys = 0
    Ra = ActiveCell.Row
    Co = ActiveCell.Column
    Nx = ActiveCell.Offset(4 - Ra; 5 - Co).Value
    Ra = 5 - Ra + (Nx + 1)
    Co = 3 - Co
    X = ActiveCell.Offset(0 + Ra; -2 + Co).Value
    For j = 0 To Nx
        Yp = ActiveCell.
            Offset(-Nx - 1 + j + Ra; -1 + Co).Value
        For l = 0 To Nx
            Mn = 1
            Xl = ActiveCell.
                Offset(-Nx - 1 + l + Ra; -2 + Co).Value

```

```

Xj = ActiveCell. _
    Offset(-Nx - 1 + j + Ra; -2 + Co).Value
If l <> j Then Mn = (X - Xl) / (Xj - Xl)
Yp = Yp * Mn
Next l
Ys = Ys + Yp
Next j
Lagrang_Qwik = Ys
End Function

```

```

Sub Интерполяция_исходные_данные()
    Dim X As Double
    ' Очистка экрана перед началом работы
    Range("A1").Select
    ActiveSheet.ChartObjects.Select
    Selection.Delete
    ActiveSheet.Buttons.Select
    Selection.Delete
    ActiveCell.Columns("A:F").EntireColumn.Select
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.BorderAround LineStyle:=xlNone
    Selection.Interior.ColorIndex = xlNone
    Selection.ClearContents
    Range("C1").Select
    ActiveCell.Offset(0; 1).Range("A1:C1").Select
    ActiveCell.Offset(0; 1).FormulaR1C1 = "Интерполяция "
    Selection.Font.ColorIndex = 3
    Selection.Borders(xlLeft).LineStyle = xlNone
    Selection.Borders(xlRight).LineStyle = xlNone
    Selection.Borders(xlTop).LineStyle = xlNone
    Selection.Borders(xlBottom).LineStyle = xlNone
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic
    With Selection.Interior
        .ColorIndex = 19
        .Pattern = xlSolid
    End With
    Range("A1").Select
    ActiveCell.Offset(3; 0).Range("A1:B1").Select
    With Selection.Interior
        .ColorIndex = 34
        .Pattern = xlSolid
    End With
    ActiveCell.Select

```

```

Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(0; 1).Range("A1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
ActiveCell.Offset(0; 1).Range("A1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Interior
    .ColorIndex = 39
    .Pattern = xlSolid
End With

ActiveCell.Offset(0; 1).Range("A1:B1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic
With Selection.Interior
    .ColorIndex = 34
    .Pattern = xlSolid
End With

' Организация ввода исходных данных
i = 0
Range("A1").Select
ActiveCell.Offset(3 + i; 0).FormulaR1C1 = " X "
ActiveCell.Offset(3 + i; 1).FormulaR1C1 = " Y(x) "
ActiveCell.Offset(3 + i; 2).FormulaR1C1 = "Yрасч(x) "
Do
Metka2:
    If i = 0 Then
        inputVal = Application.InputBox("Введите " & _
            "начальное значение аргумента X0")
    Else
        inputVal = Application.InputBox("Введите " & _
            "значение аргумента X" & i & _
            ", большее предыдущего или для " & _
            "завершения ввода ESC/Отмена")
    End If
    If (i >= 1) And (inputVal <> False) Then
        X = inputVal
        Range("A1").Select
        For j = 0 To i - 1
            If ActiveCell.Offset(4 + j; 0).Value > X Then
                MsgBox "Значения X надо вводить по " & _
                    "возрастанию. Введите другое " & _
                    "значение, больше предыдущего"; _
                    vbOKOnly; "ВНИМАНИЕ!"
            End If
        Next j
        GoTo Metka2
    End If

```

```

        End If
    Next j
End If
If inputVal <> False Then
Metka1:
    inputVal1 = Application. _
        InputBox("Введите значение функции Y" & i)
    Range("A1").Select
    ActiveCell.Offset(4 + i; 0). _
        Range("A1:B1").Select

    With Selection.Interior
        .ColorIndex = 40
        .Pattern = xlSolid
    End With
    ActiveCell.Select
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic
    ActiveCell.Offset(0; 1).Range("A1").Select
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic
    ActiveCell.Offset(0; -1).FormulaR1C1 = inputVal
    ActiveCell.Offset(0; 0).FormulaR1C1 = inputVal1
    i = i + 1
    If inputVal1 = False Then GoTo Metka1
End If
Loop Until (inputVal = False)
Nx = i - 1
If Nx < 2 Then
    MsgBox "Нужно ввести хотя бы два значения X!"; _
        vbOKOnly; "ВНИМАНИЕ!"
    Goto Metka2
End If
Range("A1").Select
ActiveCell.Offset(3; 3).FormulaR1C1 = "Nx = 0, ..."
ActiveCell.Offset(3; 4).FormulaR1C1 = Nx
    ' Ввод точки, в которой нужно восстановить функцию
Range("A1").Select
Metka3:
    inputVal = Application.InputBox("Введите " & _
        "значение аргумента X, при котором " & _
        "нужно восстановить функцию")
If inputVal <> False Then
    Range("A1").Select
    ActiveCell.Offset(4 + Nx + 1; 0).Range("A1").Select
    Selection.BorderAround Weight:=xlMedium; _
        ColorIndex:=xlAutomatic

```

```

With Selection.Interior
    .ColorIndex = 39
    .Pattern = xlSolid
End With
ActiveCell.Offset(0; 0).FormulaR1C1 = inputVal
ActiveCell.Offset(0; 2).Range("A1").Select
Selection.BorderAround Weight:=xlMedium; _
    ColorIndex:=xlAutomatic

With Selection.Interior
    .ColorIndex = 39
    .Pattern = xlSolid
End With
i_ = Trim(Str(Nx + 1))
sled = "=Lagrang_Qwik(R[" & i_ & "]C[-2]:RC[-1])"
ActiveCell.Offset(0; 0).FormulaR1C1 = sled
Else
    MsgBox "Нужно ввести интерполируемое " & _
        "значение X!"; _
        vbOKOnly; "ВНИМАНИЕ!"
    GoTo Metka3
End If
Range("A1").Select
i = Str(4 + Nx + 2)
i = Trim(i)
sled = "A5:C" + i
Graphic (sled)
Key
End Sub

```

## РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Численные методы / И.И. Данилина и др. – М.: Высшая школа, 1976. – 368 с.
2. Гутер Р.С., Овчинский Б.В. Элементы численного анализа и математической обработки результатов опыта. – М.: Наука, 1970. – 432 с.
3. Калиткин Н.Н. Численные методы. – М.: Наука, 1978. – 512 с.
4. Крылов В.И., Бобков В.В., Монастырский П.И. Вычислительные методы. Т. 1. – М.: Наука, 1976. – 302 с.
5. Васильков Ю.В., Боровков А.В. Электронный учебник по численным методам оптимизации. РосАПО № 960181 20.05.96.
6. Воробьев Г.Н., Данилова А.Н. Практикум по численным методам. – М.: Высшая школа, 1979. – 184 с.
7. Бахвалов Н.С. Численные методы. – М.: Наука, 1973. – 630 с.
8. Химмельблау Д. Прикладное нелинейное программирование. – М.: Мир, 1975. – 536 с.
9. Растринин Л.А. Современные принципы управления сложными объектами. – М.: Сов. радио, 1980. – 230 с.
10. Форсайт Дж., Малькольм М., Моулер К. Машинные методы математических вычислений / Пер. с англ. Х.Д. Икрамова. – М.: Мир, 1980. – 280 с.
11. Додж М., Кината К., Стинсон К. Эффективная работа с Excel 7.0 для Windows 95. – С.-Петербург: Питер Пресс, 1997. – 1031 с.
12. Долголатев В.Г. Работа в Excel 7.0 для Windows 95 на примерах. – М.: Бином, 1995.- 384 с.
13. Николь Н., Албрехт Р. Электронные таблицы Excel 5.0 для квалифицированных пользователей: Пер. с нем. – М., 1995. – 304 с.
14. Карлберг К. Excel для Windows в вопросах и ответах. – С.-Петербург, 1995. – 416 с.
15. Курицкий Б. Поиск оптимальных решений средствами Excel 7.0. СПб.: ВHV – С.-Петербург, 1997. – 384 с.
16. Очков В.Ф. Mathcad 8.0 Pro для студентов и инженеров. – М.: КомпьютерПресс, 1999.
17. Плис А.И., Сливина Н.А. Mathcad: математический практикум. – М.: Финансы и статистика, 1999. – 655 с.
18. Mathcad 6.0 Plus. Финансовые, инженерные и научные расчеты в среде Windows 95: Пер. с англ. – М.: Филинь, 1996. – 712 с.

19. *Очков В.Ф.* Mathcad 7.0 Pro для студентов и инженеров. – М.: Компьютер Пресс, 1998. – 384 с.
20. *Дьяконов В.П.* Справочник по Mathcad PLUS 6.0. – М.: СК Пресс, 1997. – 336 с.
21. *Дьяконов В.П.* Справочник по Mathcad 7.0 PRO. – М.: СК Пресс, 1998. – 352 с.
22. *Дьяконов В.П., Абраменкова И.В.* Mathcad 7 в математике, физике и в Internet. М.: Нолидж, 1998. – 352 с.
23. *Потемкин В.Г.* Система инженерных и научных расчетов MATLAB 5.x: В 2-х т. – М.: Диалог-МИФИ, 1999. – Т. 1 – 366 с.; Т. 2 – 304 с.
24. *Потемкин В.Г.* Система MATLAB. Справочное пособие. – М.: Диалог-МИФИ, 1997. – 350 с.
25. *Потемкин В.Г.* MATLAB 5 для студентов. Справочное пособие. – М.: Диалог-МИФИ, 1998. – 314 с.
26. *Говорухин В.Н., Цибулин В.Г.* Введение в Maple. Математический пакет для всех. – М.: Мир, 1997. – 208 с.
27. *Прохоров Г.В., Леднев М.А., Колбеев В.В.* Пакет символьных вычислений Maple V. – М.: Компания “Петит”, 1997. – 200 с.
28. *Прохоров Г.В.* и др. Математический пакет Maple V Release 4: Руководство пользователя / Г.В. Прохоров, К.И. Желнов, М.А. Леднев. – Калуга: Облиздат, 1998. – 200 с.
29. *Манзон Б.М.* Maple V Power Edition. – М.: Филинь, 1998. – 240 с.
30. *Дьяконов В.П.* Математическая система Maple V R3/R4/R5. – М.: Солон, 1998. – 400 с.
31. *Аладьев В.З., Шишаков М.Л.* Введение в среду пакета Mathematica 2.2. – М.: Филинь, 1997. – 368 с.
32. *Дьяконов В.П.* Системы символьной математики Mathematica 2 и Mathematica 3. Справочное издание. – М.: СК ПРЕСС, 1998. – 328 с.
33. *Боровиков В.П., Боровиков И.П.* STATISTICA — Статистический анализ и обработка данных в среде Windows. – М.: Филинь, 1997. – 608 с.
34. *Боровиков В.П.* Популярное введение в программу STATISTICA. – М.: Компьютер Пресс, 1998. – 267 с.
35. *Дьяконов В.П., Бирюков С.* Derive в России. – Монитор-Аспект. – 1995. – № 3.
36. *Дьяконов В.П.* Жемчужина символьной математики. – Монитор-Аспект. – 1993. – № 2.
37. *Дьяконов В.П.* Справочник по системе символьной математики Derive. – М.: СК ПРЕСС, 1998.

# ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

- Адекватность** 11, 16, 26  
**Алгоритм** 7, 8, 16, 26, 46, 55, 65,  
66, 68, 70, 77, 83, 97, 101, 105,  
112, 115, 121, 131, 132, 134–139,  
142–145, 149, 155, 156  
**Алгоритм коррекции шага** 125,  
126, 134, 135, 156  
**Аппроксимация** 8, 19, 20, 27, 40  
**Аппроксимации критерий** 41, 46  
**Аппроксимирующая функция**  
40–44, 101
- Весовые коэффициенты**  
-- аппроксимации 41  
-- численного интегрирования 49
- Глобальный максимум критерия**  
113  
**Глобальный минимум критерия**  
136  
**Глобальная оптимизация** 133, 151,  
162  
**Градиент целевой функции**  
123–135, 155, 162–164, 167
- Золотое сечение** 116
- Идентификация** 27  
**Инструменты моделирования**  
26–28  
**Интегрирование численное** 8, 19,  
20, 25, 27, 48, 49, 51  
**Интервалы существования корней**  
61  
**Интерполяция** 8, 19, 20, 27, 29–40  
**Интерполяционная функция** 52,  
101  
-- Лагранжа 31  
-- Ньютона 33
- Итерация** 87, 95–97, 114, 116, 117  
**Итерационный процесс**  
(процедура) 61, 73, 79–82, 85,  
97, 147
- Квадратичный критерий близости**  
41, 43  
**Квадратичный трехчлен** 82–84  
**Квадратурные формулы** 48  
**Конечные разности** 33  
**Критерий оптимальности** 22, 24,  
25, 110–118, 137, 139, 147, 152,  
154, 159, 164, 166  
**Критерий оптимальности**  
сепарабельный 138, 148–150
- Метод аппроксимации**  
-- наименьших квадратов 42–46  
-- равномерного приближения 46  
**Методы интерполяции**  
-- Лагранжа 31–33, 35, 37, 39  
-- Ньютона 33–36, 39, 40  
-- сплайновая 37  
-- Чебышева 36  
**Методы математического**  
программирования 23, 25  
**Методы оптимизации** 110  
**Методы спуска** 111  
**Методы численного**  
интегрирования  
--- Гаусса 20, 55–59  
--- Ньютона — Котеса 54, 55, 58  
--- прямоугольников 49–52, 57,  
58  
--- Симпсона 52, 53, 58  
--- трапеций 49–52, 57, 58  
--- Чебышева 20, 55, 56, 58, 59

**Направляющий косинус** 124, 155  
**Область существования корней** 76  
 --- метод кольца 77, 78  
 --- метод предельных значений 78  
**Ограничения** 110, 158, 165, 166  
 - автономные 158, 160, 164  
 - активные 122  
 - типа неравенств 158, 161–164, 166, 167  
 - типа равенств 158–164, 166  
**Определение числа корней алгебраических уравнений (правило Декарта)** 75  
**Оптимизационные задачи** 21  
**Оптимизация одномерная (методы)**  
 -- деления отрезка пополам 114–115, 121  
 -- золотого сечения 116, 117, 121  
 -- параболической аппроксимации 118, 119, 122  
 -- сканирования 112, 113, 120  
**Оптимизация многомерная безградиентная (методы)**  
 --- Гаусса — Зайделя 138, 139–141, 147, 148  
 --- параллельных касательных 146, 147, 150  
 --- Розенброка 140, 141, 149  
 --- симплексный 142–144, 149  
**Оптимизация многомерная безусловная градиентная (методы)**  
 --- градиента 124–126, 133–135, 155, 156, 160, 161, 164  
 --- наискорейшего спуска 128, 129, 131, 135, 139, 154  
 --- сопряженных градиентов 130, 131, 135, 136  
 --- тяжелого шарика 132, 133, 136

**Оптимизация многомерная случайная (методы)**  
 --- блуждающего поиска 155–157  
 --- поиска с “наказанием случайностью” 154, 157  
 --- слепого поиска 152, 157  
 --- случайных направлений 152, 153, 157  
**Оптимизация многомерная условная (методы)**  
 --- проектирования градиента 163–165, 167  
 --- прямого поиска с возвратом 161, 162, 166  
 --- штрафов 159, 160, 166  
**Отделение корней** 60–63, 67, 69, 88, 89

### Погрешности

- интерполяции 30–35  
 - решения дифференциальных уравнений 104, 106, 108, 109  
 - решения задач оптимизации 112–114, 117, 119, 120, 121, 135, 136, 139, 160, 163, 165  
 - решения систем линейных уравнений 93–97, 99  
 - решения систем нелинейных уравнений 85, 87  
 - уточнения корней 64, 69, 70, 71, 74  
 - численного интегрирования 51, 53, 56

**Полиномы** 30, 31, 34, 37, 38

**Предельные оценки корней** 61, 76

--- метод Лагранжа 76

--- метод Ньютона 77

**Решение дифференциальных уравнений** 8, 25, 27, 28, 100–109

--- одношаговые методы (Эйлера, Рунге — Кутты) 100–108

--- многошаговые методы  
(метод Милна) 100, 106, 107,  
109

Решение систем линейных  
уравнений

--- приближенные методы 92,  
94, 99

--- метод простой итерации 95,  
96, 99

--- метод Зайделя 98, 99

--- точные методы (Гаусса,  
Крамера, оптимального  
исключения) 92, 93, 94, 99

Решение систем нелинейных урав-  
нений 8, 25, 27, 28, 84–88, 91

--- метод итераций 86, 87, 91

--- метод Ньютона — Рафсона  
84, 85, 91

Ряд Тейлора 30, 84, 101

Сглаживание 29

Система нормальных уравнений 43

Симплекс 142–146, 149

– размеры 143, 144

– растяжение 143, 144, 149

– регулярный 142

– редукция 144

– сжатие 143, 144, 149

Сложная система 12, 13

Сплайн 37

Сплайн кубический 37

Схема Горнера 80

Теорема Безу 80

Трендовые модели 18, 19

Узел интерполяции 29, 30

Улучшающая последовательность  
111, 112, 122, 150

Уравнения алгебраические 61

Уравнения нелинейные общего  
вида 60

Условия окончания поиска  
оптимума 117, 138, 148, 150,  
152, 162, 163, 167

Условия сходимости к решению  
73, 80, 87, 94, 95, 98, 99

Уточнение корней алгебраических  
уравнений 79–84

--- уточнение действительного  
корня 79, 80

--- уточнение комплексной пары  
корней (метод Хичкока) 82, 83

Уточнение корней нелинейных  
уравнений общего вида 60, 64,  
75

--- метод деления отрезка  
пополам 65, 89

--- метод комбинированный 69,  
70

--- метод Ньютона (касатель-  
ных) 68, 69, 90

--- метод параболической  
аппроксимации 70, 71, 90

--- метод простой итерации  
72–74, 90, 91

--- метод сканирования 64, 65

--- метод хорд 65–67, 68, 89

Численное решение 16

Шаг (решения) 100–105, 108, 109

Шаг (движения к оптимуму) 112,  
113, 122–126, 128–140, 142,  
148, 150, 152–156, 161

“Штраф” 159, 160, 166

– квадратичный 159

– комбинированный 159

– модульный 159, 160

Экстраполяция 29

## СОДЕРЖАНИЕ

<b>Предисловие</b> . . . . .	<b>3</b>
<b>ВВЕДЕНИЕ</b> . . . . .	<b>7</b>
<b>Основы математического моделирования</b> . . . . .	<b>9</b>
<b>Концепция моделирования</b> . . . . .	<b>9</b>
<b>Примеры задач математического моделирования</b> . . . . .	<b>18</b>
<b>Инструменты моделирования</b> . . . . .	<b>26</b>
<b>ОБРАБОТКА ТАБЛИЧНЫХ ДАННЫХ</b> . . . . .	<b>29</b>
<b>Интерполяция</b> . . . . .	<b>29</b>
<b>Концепция интерполяции</b> . . . . .	<b>29</b>
<b>Основные методы</b> . . . . .	<b>31</b>
<b>Метод Лагранжа</b> . . . . .	<b>31</b>
<b>Метод Ньютона</b> . . . . .	<b>33</b>
<b>Метод Чебышева</b> . . . . .	<b>36</b>
<b>Метод сплайнов</b> . . . . .	<b>37</b>
<b>Контрольные вопросы</b> . . . . .	<b>39</b>
<b>Аппроксимация</b> . . . . .	<b>40</b>
<b>Концепция аппроксимации</b> . . . . .	<b>41</b>
<b>Основные методы</b> . . . . .	<b>42</b>
<b>Метод наименьших квадратов</b> . . . . .	<b>42</b>
<b>Метод равномерного приближения</b> . . . . .	<b>46</b>
<b>Контрольные вопросы</b> . . . . .	<b>46</b>
<b>Численное интегрирование</b> . . . . .	<b>48</b>
<b>Концепция численного интегрирования</b> . . . . .	<b>48</b>
<b>Основные методы</b> . . . . .	<b>49</b>
<b>Простейшие методы</b> . . . . .	<b>49</b>
<b>Метод Симпсона</b> . . . . .	<b>52</b>
<b>Метод Ньютона — Котеса</b> . . . . .	<b>54</b>
<b>Методы Чебышева и Гаусса</b> . . . . .	<b>55</b>
<b>Контрольные вопросы</b> . . . . .	<b>57</b>

<b>МЕТОДЫ РЕШЕНИЯ УРАВНЕНИЙ И ИХ СИСТЕМ . . . . .</b>	<b>60</b>
<b>Методы решения нелинейных уравнений . . . . .</b>	<b>60</b>
Концепция методов . . . . .	60
Отделение корней . . . . .	62
Уточнение корней . . . . .	64
Метод сканирования . . . . .	64
Метод деления отрезка пополам . . . . .	65
Метод хорд . . . . .	65
Метод Ньютона (касательных) . . . . .	68
Комбинированный метод . . . . .	69
Метод параболической аппроксимации . . . . .	70
Метод простой итерации . . . . .	72
Определение числа корней алгебраических уравнений . . . . .	75
Предельные оценки и область существования корней алгебраических уравнений . . . . .	76
Метод Лагранжа . . . . .	76
Метод Ньютона . . . . .	77
Метод кольца . . . . .	77
Метод предельных значений . . . . .	78
Уточнение корней алгебраических уравнений . . . . .	79
Уточнение действительного корня . . . . .	79
Уточнение комплексной пары корней (метод Хичкока) . . . . .	82
Решение систем нелинейных уравнений . . . . .	84
Концепция методов . . . . .	84
Метод Ньютона — Рафсона . . . . .	84
Метод итераций . . . . .	86
Контрольные вопросы . . . . .	88
<b>Решение систем линейных уравнений . . . . .</b>	<b>92</b>
Концепция методов . . . . .	92
Точные методы . . . . .	93
Приближенные методы . . . . .	94
Контрольные вопросы . . . . .	99
<b>Основы решения дифференциальных уравнений . . . . .</b>	<b>100</b>
Концепция решения дифференциальных уравнений . . . . .	100
Основные методы . . . . .	102
Метод Эйлера и модифицированный метод Эйлера . . . . .	102
Метод Рунге — Кутты . . . . .	105
Метод Милна . . . . .	106
Контрольные вопросы . . . . .	107

<b>МЕТОДЫ ОПТИМИЗАЦИИ</b> . . . . .	<b>110</b>
<b>Одномерная оптимизация</b> . . . . .	<b>112</b>
Концепция методов . . . . .	112
Основные методы . . . . .	112
Метод сканирования . . . . .	112
Метод деления пополам . . . . .	114
Метод золотого сечения . . . . .	116
Метод параболической аппроксимации . . . . .	118
Контрольные вопросы . . . . .	120
<b>Многомерная безусловная градиентная оптимизация</b> . . . . .	<b>122</b>
Концепция методов . . . . .	122
Основные методы . . . . .	124
Метод градиента . . . . .	124
Метод наискорейшего спуска . . . . .	128
Метод сопряженных градиентов . . . . .	130
Метод тяжелого шарика . . . . .	132
Контрольные вопросы . . . . .	134
<b>Многомерная безградиентная оптимизация</b> . . . . .	<b>137</b>
Концепция методов . . . . .	137
Основные методы . . . . .	138
Метод Гаусса — Зайделя . . . . .	138
Метод Розенброка . . . . .	140
Симплексный метод . . . . .	142
Метод параллельных касательных . . . . .	146
Контрольные вопросы . . . . .	148
<b>Многомерная случайная оптимизация</b> . . . . .	<b>150</b>
Концепция методов . . . . .	150
Основные методы . . . . .	152
Метод слепого поиска . . . . .	152
Метод случайных направлений . . . . .	152
Метод поиска с “наказанием случайностью” . . . . .	154
Метод с “блуждающим” поиском . . . . .	155
Контрольные вопросы . . . . .	157
<b>Многомерная условная оптимизация</b> . . . . .	<b>158</b>
Концепция методов . . . . .	158
Основные методы . . . . .	159
Метод штрафов . . . . .	159
Метод прямого поиска с возвратом . . . . .	161
Метод проектирования градиента . . . . .	163
Контрольные вопросы . . . . .	166

<b>ПРОГРАММНЫЕ СРЕДСТВА ДЛЯ ВЫЧИСЛИТЕЛЬНЫХ РАБОТ . . . . .</b>	<b>168</b>
<b>Eureka . . . . .</b>	<b>168</b>
<b>Электронный учебник по численным методам . . . . .</b>	<b>169</b>
<b>MathCad . . . . .</b>	<b>171</b>
<b>Mathcad explorer . . . . .</b>	<b>173</b>
<b>Maple . . . . .</b>	<b>174</b>
<b>Derive . . . . .</b>	<b>175</b>
<b>Matlab . . . . .</b>	<b>177</b>
<b>Mathematica . . . . .</b>	<b>177</b>
<b>Statistica . . . . .</b>	<b>178</b>
<b>МЕТОДИЧЕСКИЕ КОММЕНТАРИИ И ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ . . . . .</b>	<b>179</b>
<b>Методические комментарии . . . . .</b>	<b>179</b>
<b>Ответы на контрольные вопросы . . . . .</b>	<b>180</b>
<b>Интерполяция . . . . .</b>	<b>180</b>
<b>Аппроксимация . . . . .</b>	<b>183</b>
<b>Численное интегрирование . . . . .</b>	<b>185</b>
<b>Методы решения нелинейных уравнений и их систем . . . . .</b>	<b>188</b>
<b>Системы линейных уравнений . . . . .</b>	<b>192</b>
<b>Дифференциальные уравнения . . . . .</b>	<b>193</b>
<b>Методы одномерного поиска . . . . .</b>	<b>196</b>
<b>Градиентные методы многомерного поиска . . . . .</b>	<b>199</b>
<b>Безградиентные методы многомерной оптимизации . . . . .</b>	<b>203</b>
<b>Методы случайного поиска . . . . .</b>	<b>207</b>
<b>Методы многомерной условной оптимизации . . . . .</b>	<b>208</b>
<b>Технология вычислений . . . . .</b>	<b>210</b>
<b>Модуль 1. Решение нелинейных уравнений. . . . .</b>	<b>212</b>
<b>Модуль 2. Решение дифференциальных уравнений . . . . .</b>	<b>222</b>
<b>Модуль 3. Вычисление интегралов . . . . .</b>	<b>228</b>
<b>Модуль 4. Интерполяция . . . . .</b>	<b>236</b>
<b>РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА . . . . .</b>	<b>247</b>
<b>Предметный указатель . . . . .</b>	<b>249</b>

Учебное издание

**Васильков Юрий Викторович**  
**Василькова Наталья Николаевна**

## **КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ ВЫЧИСЛЕНИЙ В МАТЕМАТИЧЕСКОМ МОДЕЛИРОВАНИИ**

Ведущий редактор *Л. Д. Григорьева*  
Художественный редактор *Ю. И. Артюхов*  
Технические редакторы *И. В. Белюсенок, И. В. Завгородняя*  
Корректор *Т. М. Васильева*  
Обложка и оформление художника *Н. М. Биксентеева*  
Компьютерная верстка *А. Н. Канатникова*

ИБ № 3995

Подписано в печать 24.07.2002. Формат 60×88<sup>1</sup>/<sub>16</sub>  
Печать офсетная. Гарнитура «Таймс».  
Усл. п. л. 15,68. Уч.-изд. л. 13,22  
Тираж 3000 экз. Заказ 2500. «С» 176

Издательство «Финансы и статистика»  
101000, Москва, ул. Покровка, 7  
Телефон (095) 925-35-02, факс (095) 925-09-57

ГУП «Великолукская городская типография»  
Комитета по средствам массовой информации Псковской области  
182100, Великие Луки, ул. Полиграфистов, 78/12  
Тел./факс: (811-53) 3-62-95  
E-mail: VTL@MART.RU