Федеральное агентство по образованию

Государственное образовательное учреждение
высшего профессионального образования

Владимирский государственный университет

С.Ю. КИРИЛЛОВА

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА

Учебное пособие

Владимир 2009

УДК 519.6
ББК 22.19
К43

Рецензенты

Доктор физико-математических наук, профессор, заведующий кафедрой теоретической физики Владимирского государственного гуманитарного университета
В.Г. Рау

Доктор технических наук, профессор кафедры «Информационные системы и информационный менеджмент» Владимирского государственного университета
Р.И. Макаров

Печатается по решению редакционного совета
Владимирского государственного университета

Кириллова, С. Ю. Вычислительная математика: учеб. пособие / К43 С. Ю. Кириллова ; Владим. гос. ун-т. – Владимир: Изд-во Владим. гос. ун-та, 2009. – 102 с. ISBN 978-5-89368-988-4.

Содержит материал по основным методам вычислительной математики для решения линейных и нелинейных уравнений и систем; интерполяции и аппроксимации функций; численного интегрирования и дифференцирования; решения обыкновенных дифференциальных уравнений. Рассматриваются вопросы точности методов. Пособие ориентировано на использование пакета MathCAD и тесно связано с изданными в 2004 г. в ВлГУ методическими указаниями к практическим занятиям по вычислительной математике (составитель С.Ю. Кириллова). В то же время сведения по методам решения могут быть использованы при работе с другим математическим приложением.

Предназначено для студентов специальностей 210402, 230101, 230201, 230202 очной формы обучения, а также для студентов других специальностей, изучающих дисциплины соответствующего профиля.

Табл. 2. Ил. 25. Библиогр.: 13 назв.

УДК 519.6
ББК 22.19

ISBN 978-5-89368-988-4

© Владимирский государственный университет, 2009

ВВЕДЕНИЕ

Сегодня имеет место широкое внедрение математических методов в разнообразные сферы деятельности человека: естественные, социальные, технические и экономические науки; прикладные науки управления, такие как менеджмент, принятие управляющих решений, социально-экономическое прогнозирование и т.д. Прикладные науки своими потребностями стимулируют развитие некоторых разделов математики.

Вычислительная математика – раздел математики, включающий круг вопросов, связанных с производством вычислений и использованием компьютеров. В более узком понимании вычислительная математика – *теория численных методов* решения типовых математических задач.

Особенностью вычислительной математики является то, что при решении вычислительных задач с помощью вычислительной техники (ВТ) человек оперирует машинными числами, которые являются дискретной проекцией вещественных чисел на конкретную архитектуру компьютера. Поэтому важную роль в вычислительной математике играют оценки точности алгоритмов и их устойчивость к представлениям машинных чисел в компьютере.

Рассматривая содержание дисциплины, выделим в вычислительной математике три больших раздела:

1. Применение компьютеров в различных областях научной и практической деятельности, в основном для анализа математических моделей.
2. Разработка методов и алгоритмов решения типовых математических задач, возникающих при исследованиях математических моделей.
3. Упрощение взаимоотношений человека с компьютерами, включая теорию и практику программирования, в том числе автоматизацию программирования.

Математическое моделирование различных объектов и процессов – неотъемлемый этап разработки систем любого вида (технических, информационных, экономических, социальных и т.д.); существенный

фактор в вопросах управления, планирования, прогнозирования. Исследование некоторых объектов и процессов может быть проведено исключительно на моделях (например нештатные ситуации в ядерных реакторах). Задача выбора модели должна решаться с учётом следующего требования. Степень достоверности, с которой результаты анализа модели позволяют исследовать конкретное явление (или класс явлений), должна соответствовать точности исходной информации.

Изучение реальных явлений на основе анализа построенных моделей, как правило, требует развития численных методов решения типовых математических задач (вычислительной математики в узком смысле слова) и привлечения компьютеров. В дисциплине рассматриваются наиболее распространенные задачи вычислительной математики:

1) численные методы линейной алгебры;
2) решение нелинейных уравнений и систем;
3) интерполяция функций;
4) методы приближения и аппроксимации функций;
5) численное интегрирование и дифференцирование;
6) решение обыкновенных дифференциальных уравнений.

Быстро развивающимся направлением вычислительной математики являются численные методы оптимизации. Задача оптимизации состоит в изучении экстремальных (наибольших или наименьших) значений функционалов на множествах, как правило, весьма сложной структуры. В первую очередь следует упомянуть задачи математического программирования (в том числе линейного и динамического), к которым сводятся многие задачи экономики.

Цель дисциплины – знакомство студентов инженерных специальностей с вычислительными методами как инструментом решения задач, встречающихся в их профессиональной деятельности. Практик должен знать, на чем основан метод, как применить его, что он может дать для решения той или иной задачи, какова погрешность вычислений. Вопросы детальных теоретических основ методов, теоретического анализа условий существования решения, сходимости метода и т.п. будут затрагиваться в необходимом объеме. Для инженера исходными данными служит содержательная задача, к которой он должен подобрать наиболее эффективный метод решения. Для вычислителя-практика важную роль играют время решения задачи, удобство обращения к алгоритму, обеспечиваемая точность решения.

В настоящее время невозможно говорить о математических вычислениях без применения компьютерной техники. Компьютеры и были созданы в первую очередь для проведения научных расчетов. До сих пор научные и инженерные расчеты остаются одной из важнейших, хотя, пожалуй, и не самой бросающейся в глаза сферой приложения компьютеров. За многие годы накоплены обширные библиотеки научных подпрограмм в первую очередь на языке FORTRAN, предназначенных для решения типовых задач. Сегодня, решая задачи вычислительной математики, можно пойти одним из следующих путей:

1) разработать на основе известных алгоритмов собственные программы на каком-либо универсальном языке высокого уровня;

2) использовать приложения общего назначения (например MS Excel);

3) применить специализированные математические программные системы.

Имеется целый ряд различных математических пакетов, реализующих разнообразные численные методы, а также способных производить аналитические математические преобразования. Различие численного и аналитического решений показано фрагментом документа MathCAD ниже.

Сущность 1) символьного и 2) численного решений

а) $\frac{5}{10} \to \frac{1}{2}$ $\frac{5}{10} = 0.5;$

б) $\frac{b}{k+k} \to \frac{1}{2}\cdot\frac{b}{k}$ $\frac{b}{k+k} =$

$b := 5 \quad k := 10 \quad \frac{b}{k+k} = 0.25;$

в) $y(x) := x^2 \quad \frac{d}{dx} y(x) \to 2\cdot x$ $\frac{d}{dx} y(x) =$

$x := 3 \quad \frac{d}{dx} y(x) = 6.$

Пожалуй, наиболее известными сегодня можно назвать следующие пакеты: Mathematica (фирма Wolfram Research), Maple (фирма Waterloo

Maple Inc), Matlab (фирма The MathWorks), MathCAD (фирма MathSoft Inc). Кроме того, находит достаточное применение Derive (фирма Soft Warehouse).

Пакет **Mathematica** популярен в научных кругах. Он позволяет выполнять символьные (аналитические) преобразования, в том числе операции математического анализа: дифференцирование, интегрирование и интегральные преобразования, разложение в ряды, решение дифференциальных уравнений и т.п. Обеспечивает также применение разнообразных численных методов. Обладает развитой двух- и трехмерной графикой. По своей сущности Mathematica представляет собой язык программирования высокого уровня, позволяющий реализовать традиционный процедурный и функциональный стили программирования, а также стиль правил преобразований. Пакет требует значительных ресурсов компьютера.

Пакет **Maple**, называемый системой символьных вычислений, также весьма популярен в научных кругах. Кроме аналитических преобразований, пакет в состоянии решать задачи численно. Характерной особенностью пакета является то, что он позволяет конвертировать документы в формат LaTeX – стандартный формат подавляющего большинства научных издательств мирового класса. Кроме того, ряд других программных продуктов используют интегрированный символьный процессор Maple. Интерфейс Maple интуитивно понятен, правила работы предельно просты.

Подобно упомянутым выше пакетам, система **Matlab** фактически представляет собой своеобразный язык программирования высокого уровня, предназначенный для выполнения инженерных и научных расчетов, высококачественной визуализации получаемых результатов. Эта система применяется для математических расчетов, моделирования физических систем и управления техническими объектами. Характерной особенностью системы является то, что она позволяет сохранять документы в формате языка программирования С. Существуют различные версии системы, но все они требуют довольно значительных ресурсов компьютера.

Система **Derive** выполняет численные расчеты и символьные преобразования, обладает хорошими графическими средствами, позволяет эффективно решать самые разнообразные прикладные задачи, прежде всего задачи математического моделирования в науке, технике и экономике.

Пакет **MathCAD** популярен, пожалуй, более в инженерной, чем в научной среде. Характерной особенностью пакета является использование привычных стандартных математических обозначений, то есть документ

на экране выглядит точно так же, как обычный математический расчет. Для использования пакета не требуется изучать какую-либо систему команд, как, например, в случае пакетов Mathematica или Maple, хотя пакет и имеет встроенный язык программирования. Пакет ориентирован в первую очередь на проведение численных расчетов, но имеет встроенный символьный процессор Maple, что позволяет выполнять аналитические преобразования. В последних версиях предусмотрена возможность создавать связки документов MathCAD с документами Matlab. В отличие от упомянутых выше пакетов, MathCAD является средой визуального программирования, то есть не требует знания специфического набора команд. Простота освоения пакета, дружественный интерфейс, относительная непритязательность к возможностям компьютера явились главными причинами того, что именно этот пакет часто выбирают для изучения численных методов.

MathCAD представляет собой интегрированную среду для выполнения, документирования и обмена результатами технических вычислений. В состав MathCAD входят:

1) мощный текстовый редактор, позволяющий вводить, редактировать и форматировать как текст, так и математические выражения;

2) численный процессор, умеющий проводить расчеты по введенным формулам, используя встроенные численные методы;

3) символьный процессор, являющийся фактически системой искусственного интеллекта;

4) огромное хранилище справочной информации как математической, так и инженерной, оформленной в качестве интерактивной электронной книги.

С помощью MathCAD можно решать следующие задачи:

1) ввод разнообразных математических выражений (для дальнейших расчетов или создания документов, презентаций, Web-страниц);

2) проведение математических расчетов;

3) подготовка графиков с результатами расчетов;

4) ввод исходных данных и вывод результатов в текстовые файлы или файлы с базами данных в других форматах;

5) подготовка отчетов работы в виде печатных документов;

6) подготовка Web-страниц и публикация результатов в Интернете;

7) получение различной справочной информации из области математики.

Развитие пакета идет постоянно, появляются новые версии: MathCAD 6, MathCAD 7, MathCAD 8, MathCAD 2000, MathCAD 2001, MathCAD 13, MathCAD 14, MathCAD 15.

В последнее время просматривается тенденция к сближению и интеграции различных пакетов. Например, последние выпуски пакетов Mathematica и Maple имеют хорошие возможности для визуального программирования; в Matlab включена библиотека аналитических преобразований Maple; MathCAD позволяет работать совместно с Matlab.

Ознакомиться с основами работы в системе MathCAD предлагается при выполнении практической работы № 1 [2, с. 4].

Контрольные вопросы

1. Определение и содержание дисциплины «Вычислительная математика».
2. Какие способы решения задач вычислительной математики вы знаете?
3. В чем сущность аналитического и численного решения задачи?
4. Охарактеризуете математические программные системы.

1. ОСНОВЫ ТЕОРИИ ПОГРЕШНОСТЕЙ

1.1. Виды погрешностей

При численном решении математических и прикладных задач почти неизбежно появление на том или ином этапе их решения погрешностей следующих трех типов.

1) ***Погрешность задачи***, состоящая из двух частей: погрешности математической модели и погрешности параметров этой модели**.** Погрешность задачи связана с приближенным характером исходной содержательной модели (в частности, с невозможностью учесть все факторы в процессе изучения моделируемого явления), а также ее математического описания, параметрами которого служат обычно приближенные числа (например из-за принципиальной невозможности выполнения абсолютно точных измерений). Для вычислителя погрешность задачи следует считать ***неустранимой*** (***безусловной***), хотя постановщик задачи иногда может ее изменить.

2) ***Погрешность метода.*** Это погрешность, связанная со способом решения поставленной математической задачи и появляющаяся в результате подмены исходной математической модели другой или конечной последовательностью других более простых (например линейных) моделей. При создании численных методов закладывается возможность отслеживания таких погрешностей и доведения их до сколь угодно малого уровня. Отсюда естественно отношение к погрешности метода как к ***устранимой*** (или ***условной***).

3) ***Погрешность действий*** (погрешность округлений или ***вычислительная*** погрешность). Этот тип погрешностей обусловлен необходимостью выполнять арифметические операции над числами, усеченными до количества разрядов, зависящих от применяемой ВТ (если, разумеется, не используются специальные программные средства, реализующие, например, арифметику рациональных чисел).

Все три описанных типа погрешностей в сумме дают ***полную погрешность*** результата решения задачи. Поскольку первый тип погрешностей не находится в пределах компетенции вычислителя, для него он служит лишь ориентиром точности, с которой следует рассчитывать математическую модель. Нет смысла решать задачу существенно точнее, чем это диктуется неопределенностью исходных данных. Таким образом, погрешность метода подчиняют погрешности задачи. Наконец, при выводе оценок погрешностей численных методов обычно исходят из предположения, что все операции над числами выполняются точно. Это означает, что погрешность округлений не должна существенно отражаться на результатах реализации методов, т.е. должна подчиняться погрешности метода. Влияние погрешностей округлений не следует упускать из вида ни на стадии отбора и алгоритмизации численных методов, ни при выборе вычислительных и программных средств, ни при выполнении отдельных действий и вычислении значений функций.

Рассмотрим некоторые возможные подходы к учету погрешностей действий.

Пусть a – точное значение, a^* – приближенное значение некоторой величины.

Абсолютной погрешностью приближенного значения a^* называется величина $\Delta\left(a^*\right)=\left|a-a^*\right|$.

Относительной погрешностью значения a^* (при $a \neq 0$) называется величина $\delta(a^*) = \dfrac{\Delta(a^*)}{|a|}$.

Так как значение a, как правило, неизвестно, чаще получают ***оценки*** погрешностей вида $|a - a^*| \le \overline{\Delta}(a^*)$; $\dfrac{|a - a^*|}{|a|} \le \overline{\delta}(a^*)$.

Величины $\overline{\Delta}(a^*)$ и $\overline{\delta}(a^*)$ называют верхними ***границами*** (или просто границами) абсолютной и относительной погрешностей. Часто применяют также термин ***«предельные погрешности»***.

Пример 1.1. Абсолютная и относительная погрешности приближенного числа e [2, с. 8].

Число e – трансцендентное число, представляется бесконечной непериодической дробью $e = 2.71828...$ Приближенное значение числа $e^* = 2.7$. Граница абсолютной погрешности $\Delta(e^*) = |e - e^*| = |2.71828\ldots - 2.7| \le 0.019$; граница относительной погрешности числа

$$\delta(e^*) = |e - e^*| / |e^*| \le 0.019 / 2.7 \approx 0.007 .$$

Информацию о том, что a^* является приближенным значением числа a с абсолютной погрешностью $\Delta(a^*)$, также записывают в виде

$a = a^* \pm \Delta(a^*)$, т.е. $a^* - \Delta(a^*) \le a \le a^* + \Delta(a^*)$.

Числа a^* и $\Delta(a^*)$ принято записывать с одинаковым числом знаков после запятой. Например, результат взвешивания на бытовых весах с ценой деления 100 г: 3.40 ± 0,05 кг.

Значащими цифрами числа a^* называют все цифры в его записи, начиная с первой ненулевой слева.

Пример 1.2. Значащие цифры числа [2, с. 9].

Значащую цифру числа a^* называют ***верной***, если абсолютная погрешность числа не превосходит единицы разряда, соответствующего этой цифре.

Пример 1.3. Верные цифры числа [2, с. 9].

1.2. Погрешность действий

Рассмотрим оценку погрешности результата вычисления значения дифференцируемой функции m переменных $y = f(\boldsymbol{x}) = f(x_1^*, x_2^*, ..., x_m^*)$, вычисление которой производится при приближенно заданных значениях аргументов $x_1^*, x_2^*, ..., x_m^*$. Точная абсолютная погрешность результата есть

$$\Delta(y^*) \le \left| f(\boldsymbol{x}) - f(\boldsymbol{x}^*) \right|$$

– модуль полного приращения функции. Главной, т.е. линейной частью этого приращения, будет полный дифференциал dy. Таким образом имеем:

$$\Delta(y^*) \approx |dy| = \left| \sum_{j=1}^{m} \frac{\partial y}{\partial x_j} dx_j \right| \le \sum_{j=1}^{m} \left| \frac{\partial y}{\partial x_j} \right| \cdot \Delta(x_j^*). \qquad (1.1)$$

Эту величину приближенно можно принять в качестве границы абсолютной погрешности результата.

Для относительной погрешности функции справедливо следующее приближенное равенство:

$$\delta(y^*) = \frac{\Delta(y^*)}{\left| f(x^*) \right|} \approx \sum_{j=1}^{m} \left| \frac{\partial y}{\partial x_j} \right| \cdot \frac{\Delta(x_j^*)}{\left| f(x^*) \right|} = \sum_{j=1}^{m} \frac{\left| f'_{x_j}(x^*) \right|}{\left| f(x^*) \right|} \cdot \left| x_j^* \right| \cdot \delta(x_j^*). \qquad (1.2)$$

Как частные случаи формул (1.1), (1.2) можно получить известные правила оценивания погрешностей результатов арифметических действий.

Абсолютная погрешность алгебраической суммы (суммы или разности) не превосходит суммы абсолютных погрешностей слагаемых, т.е.

$$\Delta(a^* \pm b^*) \le \Delta(a^*) + \Delta(b^*).$$

Если a и b – ненулевые числа одного знака, то справедливы неравенства

$$\delta(a^* + b^*) \le \delta_{\max},$$

где $\delta_{\max} = \max\left\{\delta(a^*), \delta(b^*)\right\}$.

С вычитанием приближенных чисел дело обстоит хуже: оценка *относительной погрешности разности двух приближенных положительных чисел*

$$\delta(a^* - b^*) = \frac{\Delta(a^* - b^*)}{\left| a^* - b^* \right|} \le \frac{\Delta(a^*) + \Delta(b^*)}{\left| a^* - b^* \right|} = \frac{\delta(a^*) \cdot \left| a^* \right| + \delta(b^*) \cdot \left| b^* \right|}{\left| a^* - b^* \right|} \le \frac{\left| a^* + b^* \right|}{\left| a^* - b^* \right|} \cdot \delta_{\max}$$

указывает на возможность сильного возрастания погрешности при $a^* - b^* \to 0$. В этом случае говорят о потере точности при вычитании близких чисел.

Для *относительных погрешностей произведения и частного приближенных чисел* верны оценки

$$\delta(a^* \cdot b^*) = \delta(a^* / b^*) \approx \delta(a^*) + \delta(b^*).$$

Для абсолютных погрешностей произведения и частного приближенных чисел можно использовать оценку

$$\Delta(a* \cdot b*) \le \delta(a* \cdot b*) \cdot |a* \cdot b*|.$$

Пример 1.4. Погрешности арифметических действий [2, с. 9]. Пример представлен в среде MathCAD в электронном комплексе для практических занятий – файл *R2_ex4.mcd*.

Пример 1.5. Погрешность функции многих переменных [2, с. 10]. Пример представлен в среде MathCAD в электронном комплексе для практических занятий – файл *R2_ex5.mcd*.

1.3. Понятие о погрешностях машинной арифметики

Все вычисления в компьютере в силу конечности разрядной сетки выполняются с конечной точностью. Возникающие при этом погрешности называют *вычислительными*, или *машинными*. Как правило, эти погрешности невелики и в большинстве случаев учет их влияния не производится. Но необходимо представлять, какими возможностями располагают компьютеры в плане решения вычислительных задач, уметь их грамотно использовать и знать границы этих возможностей.

Рассмотрим арифметику вещественных чисел, для представления которых в компьютере применяют в основном два способа: с фиксированной и с плавающей точкой. Второй способ употребляется значительно чаще.

В большинстве современных компьютеров с использованием двоичной системы счисления машинное число с плавающей точкой x в нормализованном виде представляется в экспоненциальной форме $x = \pm\mu \times 2^p$, где μ – *мантисса*, причем всегда $1 \le |\mu| < 2$, p – целое число, называемое двоичным *порядком*.

Например, $A = -17{,}5_{10} = -10001{,}1_2 = -1{,}00011_2 \cdot 2^4$.

Пределы изменения порядка p ограничивают абсолютную величину используемых чисел, т.е. *диапазон представления чисел* в компьютере. Все представимые числа удовлетворяют неравенствам: $0 < X_0 \le |x| < X_\infty$, где $X_0 = 2^{p_{\min}}$, $X_\infty = 2^{p_{\max}+1}$ (как правило, $p_{\min} = -p_{\max}$). Все числа, по модулю большие X_∞, не представимы на ЭВМ и рассматриваются как ***машинная бесконечность***. Все числа, по модулю меньшие X_0, для ЭВМ не отличаются от нуля и рассматриваются как ***машинный нуль***.

Количество t цифр, которое отводится для записи мантиссы, называется *разрядностью мантиссы*. Эта величина ограничивает относительную точность представления чисел в компьютере, то есть границу относительной погрешности представления чисел в ЭВМ. В качестве характеристики этой точности используют величину ε_M, называемую ***машинный эпсилон*** (***macheps***). Эта величина определяется как расстояние между единицей и ближайшим следующим за ней числом системы машинных чисел с плавающей точкой. Машинный эпсилон определяется разрядностью мантиссы и способом округления чисел, реализованным на конкретной ЭВМ. Покажем, что $\varepsilon_M \approx 2^{-t}$. Пусть $x^* = \mu \cdot 2^p$, тогда граница абсолютной погрешности представления этого числа равна $\overline{\Delta}(x^*) \approx 2^{-t-1} \cdot 2^p$. Поскольку $\frac{1}{2} \le \mu < 1$, то величина относительной погрешности представления оценивается так:

$$\overline{\delta}(x^*) \approx \frac{\overline{\Delta}(x^*)}{|x^*|} \approx \frac{2^{-t-1} \cdot 2^p}{\mu \cdot 2^p} = \frac{2^{-t-1}}{\mu} \le \frac{2^{-t-1}}{2^{-1}} = 2^{-t}.$$

Пример 1.6. Машинные нуль, бесконечность, эпсилон [2, с. 10]. Пример представлен в среде MathCAD в электронном комплексе для практических занятий – файл *R2_ex6.mcd*.

Для обработки процессором действительные числа с плавающей точкой должны быть в формате IEEE (ANSI / IEEE Standart 754 - 1985). Формат стандарта IEEE состоит из знака (старший бит), смещенного порядка, или характеристики (d бит), и мантиссы (t бит); по умолчанию предполагается, что мантисса дополнена еще одним битом, содержащим 1.

Возможные типы представления действительных чисел с плавающей точкой показаны в следующей таблице:

Тип	N – длина (байт)	d (бит)	Смещение	t (бит)	X_0	X_∞	ε_M
Короткие действительные (*Single*)	4	8	127	23	$2^{-126} \approx 10^{-39}$	$2^{127} \approx 10^{38}$	2^{-23}
Действительные обычной точности (*Real*)	6	8	127	39	$2^{-126} \approx 10^{-39}$	$2^{127} \approx 10^{38}$	2^{-39}
Действительные удвоенной точности (*Double*)	8	11	1023	52	$2^{-1022} \approx 10^{-315}$	$2^{1023} \approx 10^{315}$	2^{-52}
Расширенные действительные (*Extended*)	10	15	16383	64	$2^{-16382} \approx 10^{-4931}$	$2^{16383} \approx 10^{4932}$	2^{-64}

Например, внутреннее машинное представление выше рассмотренного числа в формате короткое действительное

$$[\mathrm{A}] = 1\ 10000011\ 00011000000000000000000_2 = \mathrm{C18C0000}_{16}.$$

Согласно стандарту IEEE для вычислений с плавающей точкой операции сложения, вычитания, умножения, деления и вычисления квадратного корня реализуются в процессорах с погрешностью не более аппроксимационной. Пусть a и b – машинные вещественные числа; знаком $*$ обозначим одну из четырех бинарных арифметических операций $+,-,\times,/$; результат выполнения операции $*$ с числами a и b в компьютере – $(a*b)_M$. Тогда при отсутствии переполнения

$$(a*b)_M \le \varepsilon_M \cdot |a*b|_M + \mathrm{X}_0\ ; \qquad \left(\sqrt{a}\right)_M \le \varepsilon_M \cdot a\ .$$

Операции вычисления математических функций (кроме операции извлечения квадратного корня) могут проводиться с ошибкой.

Важной характеристикой ВТ является также соотношение X_0, X_∞, ε_M. Точностные характеристики являются достаточно сбалансированными, если $X_0 < \varepsilon_M{}^2$, $X_\infty > \varepsilon_M{}^{-2}$.

Отметим следующие свойства машинных чисел:

1) множество машинных чисел конечно;

2) машинные операции не удовлетворяют законам ассоциативности и дистрибутивности $a*(b*c) \neq (a*b)*c$, $a(b+c) \neq ab+ac$;

3) возможны ситуации для ненулевых a и b: $a+b=a$, $ab=0$;

4) свойство коммутативности сохраняется: $a*b=b*a$;

5) свойство монотонности сохраняется: если $a \le b$, $c>0$, то $ac \le bc$, $a+c \le b+c$.

Основам теории погрешностей и машинной арифметики посвящена практическая работа № 2 [2, с. 8].

1.4. Статистический и технический подходы к учету погрешностей действий

Рассмотренный выше *аналитический* (или *классический*) *способ учета погрешностей действий*, предполагающий точное оценивание погрешностей, имеет два существенных недостатка:

1) способ чрезвычайно громоздок и не может быть рекомендован при массовых вычислениях;

2) способ учитывает крайние, наихудшие случаи взаимодействия погрешностей, которые допустимы, но маловероятны. Например, при суммировании нескольких приближенных чисел, полученных округлением как с избытком, так и с недостатком, произойдет частичная компенсация погрешностей. При больших количествах однотипных вычислений вступают в силу уже *вероятностные* или *статистические законы* формирования погрешностей результатов действий. Например, методами теории вероятностей показывается, что математическое ожидание абсолютной погрешности суммы n слагаемых с одинаковым уровнем абсолютных погрешностей при достаточно большом n пропорционально $\sqrt{n}$. В частности, если $n>10$ и все слагаемые округлены до m-го десятичного разряда, то для подсчета абсолютной погрешности суммы S применяют правило Чеботарева $\Delta(S) \approx \sqrt{3n} \cdot 0{,}5 \cdot 10^{-m}$.

Различие в результатах классического и статистического подходов к оцениванию погрешности суммы рассмотрим на примере оценки погрешности среднего арифметического нескольких приближенных чисел. При одинаковом уровне абсолютных погрешностей $\Delta(x_j^*) = 0{,}5 \cdot 10^{-m}$ исходных чисел классической оценкой абсолютной погрешности среднего арифметического $s^* = \frac{1}{n} \cdot \sum_{j=1}^{n} x_j^*$ будет величина

$$\Delta(s^*) = \frac{1}{n} \sum_{j=1}^{n} \Delta(x_j^*) = \frac{1}{n} \cdot n \cdot 0{,}5 \cdot 10^{-m} = 0{,}5 \cdot 10^{-m} = \Delta(x_j^*),$$

т.е. такая же, как и у исходных данных. В то же время по правилу Чеботарева имеем

$$\Delta(s^*) \approx \frac{1}{n}\sqrt{3n} \cdot 0{,}5 \cdot 10^{-m} = \sqrt{\frac{3}{n}} \cdot 0{,}5 \cdot 10^{-m} = \sqrt{\frac{3}{n}} \cdot \Delta(x_j^*) \xrightarrow[n \to \infty]{} 0.$$

Как видим, применение правила Чеботарева приводит к естественному выводу о том, что арифметическое усреднение результатов измерений или наблюдений увеличивает точность, чего нельзя сказать на основе классической теории погрешностей.

Прямое применение вероятностно-статистических оценок погрешностей также является достаточно сложным делом и вряд ли может быть рекомендовано при рядовых массовых вычислениях. Однако именно такие оценки подкрепляют практические правила работы с приближенными числами, составляющие основу так называемого *технического подхода*. Этот подход связывают с именем известного русского кораблестроителя, математика и механика академика Алексея Николаевича Крылова (1863 – 1945). Согласно ***принципу А.Н. Крылова*** *приближенное число должно записываться так, чтобы в нем все значащие цифры, кроме последней, были верными и лишь последняя была бы сомнительна, и при том в среднем не более чем на одну единицу.*

Чтобы результаты арифметических действий, совершаемых над приближенными числами, записанными в соответствии с принципом А.Н. Крылова, также соответствовали этому принципу, нужно придерживаться следующих простых правил:

1) при сложении и вычитании приближенных чисел в результате следует сохранять столько десятичных знаков, сколько их в приближенном данном с наименьшим количеством десятичных знаков;

2) при умножении и делении в результате следует сохранять столько значащих цифр, сколько их имеет приближенное данное с наименьшим числом значащих цифр;

3) результаты промежуточных вычислений должны иметь один-два запасных знака (которые затем должны быть отброшены).

Пример 1.7

$0.1245 + 3.52 = 3.6445 \approx 3.64$; $0.1245 \cdot 3.52 = 0.438240 \approx 0.438$.

Таким образом, при техническом подходе к учету погрешностей приближенных вычислений предполагается, что в самой записи приближенного числа содержится информация о его точности. И хотя прямая выгода от применения правил работы с приближенными числами может быть получена лишь при ручном счете (не нужно оперировать с цифрами, не влияющими на информативную часть приближенного результата), их знание и понимание помогает правильной интерпретации компьютерных расчетов, а иногда и самой организации таковых.

1.5. Неустойчивые задачи (корректные и некорректные задачи)

В силу неизбежного появления погрешностей в исходных данных задачи, а также погрешностей округления при ее решении следует иметь представление о том, насколько чувствительными могут оказаться сами задачи и методы их решения к таким погрешностям. Приведем два примера чрезмерной чувствительности.

1) Многочлен $P_{20}(x) = (x-1)(x-2)...(x-20) = x^{20} - 210x^{19} + ... + 20!$ имеет двадцать действительных корней: $x_1 = 1, \quad x_2 = 2, \ ... \ , \quad x_{20} = 20$.

Предположим, что только в одном коэффициенте при x^{19} сделана ошибка порядка ε_M: $-(210 + 2^{-23}) \approx -(210 + 10^{-7})$. Возмущенный многочлен будет иметь следующие корни:

$$x_1 \approx 1.000, \quad x_2 \approx 2.000, \quad ... \ , x_8 \approx 8.007, \quad x_9 \approx 8.917,$$
$$x_{10,11} \approx 10.095 \pm 0.644i, \quad ... \ , x_{20} \approx 20.847.$$

Корни меняются и количественно, и качественно даже при малом возмущении.

2) Линейная система

$$\begin{cases} x + 10y = 11 \\ 100x + 1001y = 1101 \end{cases}$$

имеет единственное решение $x=1,\quad y=1$. Допустив абсолютную погрешность в 0,01 в правой части 1-го уравнения, получим возмущенную систему

$$\begin{cases} x+10y=11.01 \\ 100x+1001y=1101 \end{cases}$$

с единственным решением $x=11{,}01,\quad y=0$, что далеко от решения исходной системы.

Кроме неустойчивых задач, существуют неустойчивые методы. Правда, им можно найти альтернативу. Для решения задач, требующих особого учета возмущений, разрабатываются особые подходы, приводящие к построению специальных устойчивых численных методов, называемых методами ***регуляризации***.

В начале XX в. при выяснении вопроса о соответствии математических и физических моделей задач естествознания было введено понятие корректности математической задачи. Любой объект или процесс в самом общем виде можно представить следующей схемой (рис. 1.1).

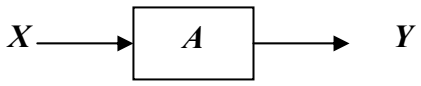


Рис. 1.1. Схема представления объекта

Имеем операторное уравнение $\boldsymbol{AX}=\boldsymbol{Y}$, где $\boldsymbol{A}:\boldsymbol{X}\to\boldsymbol{Y}$ – некоторый оператор, отображающий множество $\boldsymbol{X}$ на множество $\boldsymbol{Y}$. Если известны $\boldsymbol{A}$, $\boldsymbol{Y}$, а $\boldsymbol{X}$ неизвестен, то имеем *задачу восстановления сигнала*. Если известны $\boldsymbol{X}$, $\boldsymbol{Y}$, а $\boldsymbol{A}$ неизвестен, то имеем *задачу идентификации*. Оба случая относятся к так называемым ***обратным задачам***.

Определение. Задача нахождения элемента $x\in\boldsymbol{X}$ по заданному элементу $y\in\boldsymbol{Y}$ из уравнения $\boldsymbol{AX}=\boldsymbol{Y}$ называется ***корректной по Адамару*** (1865 – 1963, французский математик), если:

1) при каждом $y\in\boldsymbol{Y}$ существует решение $x\in\boldsymbol{X}$;

2) это решение x единственно в $\boldsymbol{X}$;

3) решение непрерывно зависит от правой части уравнения $\boldsymbol{AX}=\boldsymbol{Y}$.

При нарушении любого условия задача считается ***некорректной***.

Третье условие состоит в том, что в корректно поставленных задачах небольшие изменения в исходных данных (малые возмущения $\boldsymbol{A}$ и $\boldsymbol{Y}$) вызывают небольшие изменения решения.

Некорректные задачи встречаются при создании систем автоматической математической обработки результатов наблюдений и физических экспериментов. Это задачи восстановления сигнала, обратные задачи теплопроводности, геофизики, астрофизики и др.

Академиком А.Н. Тихоновым (1906 – 1993, математик и геофизик) разработана общая стратегия построения устойчивых методов решения некорректных задач с помощью так называемого ***регуляризующего оператора*** (алгоритма) – ***регуляризатора***.

1.6. Погрешности корней скалярных уравнений с приближенными коэффициентами

Коэффициенты алгебраических или трансцендентных уравнений вида $f(x)=0$, описывающих реальные объекты, как правило, точно не известны (грубость модели, неточность измерений, усечение чисел при вводе в ЭВМ и т.п.). Соответственно встает вопрос о том, как влияет погрешность коэффициентов на погрешности его корней. Иначе, с какой точностью имеет смысл решать данное уравнение, если известно, что его коэффициенты не точны, но имеется информация об уровне их погрешностей?

Реально можно оценить лишь главную часть погрешности, понимая под ней модуль дифференциала. Оценить безусловную абсолютную погрешность приближенного корня *x,* имея оценки абсолютных погрешностей коэффициентов Δ_{a_i}, можно по формуле:

$$|\Delta x| \le \frac{1}{\left|\frac{\partial f}{\partial x}\right|}\left(\left|\frac{\partial f}{\partial a_1}\right|\Delta_{a_1} + \left|\frac{\partial f}{\partial a_2}\right|\Delta_{a_2} + \ldots + \left|\frac{\partial f}{\partial a_m}\right|\Delta_{a_m}\right).$$

Модули частных производных функции f по коэффициентам называются *коэффициентами чувствительности*.

Определение погрешностей скалярных уравнений с приближенными коэффициентами выполняется в практической работе № 3 [2, с.15].

1.7. Обусловленность линейных алгебраических систем

Учитывая распространенность систем линейных алгебраических уравнений (СЛАУ), ибо часто именно к ним сводится на определенном

этапе процесс математического моделирования, рассмотрим подход к количественной оценке степени неопределенности этих задач. Знание таких характеристик позволяет обоснованно судить о корректности моделей, грамотно подбирать методы и строить алгоритмы, правильно трактовать полученные результаты.

Пусть рассматривается система линейных алгебраических уравнений

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + ... + a_{1m}x_m = b_1, \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + ... + a_{2m}x_m = b_2, \\ ....., \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + ... + a_{mm}x_m = b_m. \end{cases}$$

В матричной форме записи она имеет вид $\boldsymbol{Ax} = \boldsymbol{b}$. Будем предполагать, что матрица системы $\boldsymbol{A}$ задана и является невырожденной. Известно, что в этом случае решение системы существует, единственно и устойчиво по входным данным. Если $\boldsymbol{A}$ и $\boldsymbol{b}$ возмущены, то вместо точного решения СЛАУ $\boldsymbol{x} = (x_1, x_2, ..., x_m)^{\mathrm{T}}$ получим возмущенное (приближенное) $\boldsymbol{x}^* = (x_1^*, x_2^*, ..., x_m^*)^{\mathrm{T}}$: $\boldsymbol{A}^*\boldsymbol{x}^* = \boldsymbol{b}^*$.

Для количественной характеристики вектора погрешности $\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}^*$ введем понятие *нормы*. В линейной алгебре используются различные *векторные нормы*, которые ставят в соответствие вектору некоторую скалярную числовую характеристику. Норма вектора отражает порядок величины его элементов.

Нормой вектора $\boldsymbol{x}$ называется число $\|\boldsymbol{x}\|$, удовлетворяющее трем аксиомам:

1) $\|\boldsymbol{x}\| \geq 0$, причем $\|\boldsymbol{x}\| = 0$ тогда и только тогда, когда $\boldsymbol{x}$=0;

2) $\|\alpha\boldsymbol{x}\| = |\alpha| \cdot \|\boldsymbol{x}\|$ для любого вектора $\boldsymbol{x}$ и любого числа α;

3) $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$ для любых векторов $\boldsymbol{x}$ и $\boldsymbol{y}$.

В зависимости от специфики задачи применяют различные виды норм. Часто можно применить любую. Наиболее употребительными являются следующие три нормы:

$\|\boldsymbol{x}\|_1 = \sum_{i=1}^{m} |x_i|$ – норма-сумма;

$\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{m} |x_i|^2}$ – евклидова норма (модуль вектора);

$\|\boldsymbol{x}\|_\infty = \max_{1 \le i \le m} |x_i|$ – норма-максимум (max-норма, ∞-норма, infinity norm).

Матричные нормы ставят в соответствие матрице некоторую скалярную числовую характеристику. Норма матрицы отражает порядок величины матричных элементов. *Нормой матрицы* **A** называется величина $\|\boldsymbol{A}\| = \max_{x \ne 0} \frac{\|\boldsymbol{Ax}\|}{\|\boldsymbol{x}\|}$. Введенная норма обладает свойствами, аналогичными свойствам нормы вектора:

1) $\|\boldsymbol{A}\| \ge 0$, причем $\|\boldsymbol{A}\| = 0$ тогда и только тогда, когда $\boldsymbol{A} = 0$;

2) $\|\alpha \boldsymbol{A}\| = |\alpha| \cdot \|\boldsymbol{A}\|$ для любой матрицы $\boldsymbol{A}$ и любого числа α;

3) $\|\boldsymbol{A} + \boldsymbol{B}\| \le \|\boldsymbol{A}\| + \|\boldsymbol{B}\|$ для любых матриц $\boldsymbol{A}$ и $\boldsymbol{B}$;

4) $\|\boldsymbol{A} \cdot \boldsymbol{B}\| \le \|\boldsymbol{A}\| \cdot \|\boldsymbol{B}\|$.

Каждой из векторных норм соответствует своя подчиненная норма матрицы

$$\|\boldsymbol{A}\|_1 = \max_j \sum_{i=1}^{m} |a_{ij}|, \qquad \|\boldsymbol{A}\|_2 = \sqrt{\lambda_{\max}(\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}})}, \qquad \|\boldsymbol{A}\|_\infty = \max_i \sum_{j=1}^{m} |a_{ij}|.$$

В оценках вместо нормы $\|\boldsymbol{A}\|_2$ используется евклидова норма матрицы $\|\boldsymbol{A}\|_e = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{m} a_{ij}^2}$, так как $\|\boldsymbol{A}\|_2 \le \|\boldsymbol{A}\|_e$.

Пример 1.8. Вычисление норм вектора и матрицы [2, с.18].

Пример 1.9. Вычисление норм вектора в среде MathCAD представлено на рис. 1.2.

Под *абсолютной погрешностью приближенного вектора* понимают норму разности между точным и приближенным векторами $\Delta(\boldsymbol{x}^*) = \|\boldsymbol{x} - \boldsymbol{x}^*\|$, а под *относительной погрешностью вектора* – отношение абсолютной погрешности к норме вектора (точного или приближенного) $\delta(\boldsymbol{x}^*) = \frac{\|\boldsymbol{x} - \boldsymbol{x}^*\|}{\|\boldsymbol{x}\|}$.

$$b := \begin{pmatrix} 0 \\ 3 \\ -4 \end{pmatrix} \qquad i := 1..3 \qquad b1_i := |b_i| \qquad b1 = \begin{pmatrix} 0 \\ 3 \\ 4 \end{pmatrix}$$

Норма-сумма: $n1\_b := \sum b1$ $\qquad n1\_b = 7$

Евклидова норма: $ne\_b := |b1|$ $\qquad ne\_b = 5$

Норма-максимум (infiniti norm): $ni\_b := \max(b1)$ $\qquad ni\_b = 4$

Рис. 1.2. Вычисление норм вектора в среде MathCAD

Абсолютная и относительная погрешности матрицы вводятся аналогично погрешностям вектора с помощью формул

$$\Delta(\boldsymbol{A}^*) = \left\| \boldsymbol{A} - \boldsymbol{A}^* \right\| \quad \text{и} \quad \delta(\boldsymbol{A}^*) = \frac{\left\| \boldsymbol{A} - \boldsymbol{A}^* \right\|}{\|\boldsymbol{A}\|}.$$

Рассмотрим вопрос о количественной оценке степени неопределенности задачи решения СЛАУ

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \ . \tag{1.3}$$

Пусть правая часть (1.3) «возмущена» на $\Delta\boldsymbol{b}$, т.е. $\boldsymbol{A}\boldsymbol{x}^* = \boldsymbol{b}^*$, или

$$\boldsymbol{A}(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{b} + \Delta\boldsymbol{b} \ . \tag{1.4}$$

Выясним связь между относительными погрешностями вектора свободных членов и вектора-решения, т.е. получим оценку вида

$$\frac{\|\Delta\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \le (?) \frac{\|\Delta\boldsymbol{b}\|}{\|\boldsymbol{b}\|}.$$

Подставляя (1.3) в (1.4), видим:

$$\boldsymbol{A}\boldsymbol{x} + \boldsymbol{A}\Delta\boldsymbol{x} = \boldsymbol{A}\boldsymbol{x} + \Delta\boldsymbol{b},$$

$$\boldsymbol{A}\Delta\boldsymbol{x} = \Delta\boldsymbol{b},$$

$$\Delta\boldsymbol{x} = \boldsymbol{A}^{-1}\Delta\boldsymbol{b}. \tag{1.5}$$

Нормируя равенства (1.3) и (1.5), получим:

$$\|\boldsymbol{b}\| \le \|\boldsymbol{A}\| \cdot \|\boldsymbol{x}\| \quad \text{и} \quad \|\Delta\boldsymbol{x}\| \le \left\|\boldsymbol{A}^{-1}\right\| \cdot \|\Delta\boldsymbol{b}\|,$$

где матричная норма должна быть согласована с выбранной векторной нормой. Эти два числовых неравенства одинакового смысла можно перемножить

$$\|\boldsymbol{b}\| \cdot \|\Delta\boldsymbol{x}\| \le \|\boldsymbol{A}\| \cdot \|\boldsymbol{x}\| \cdot \left\|\boldsymbol{A}^{-1}\right\| \cdot \|\Delta\boldsymbol{b}\|.$$

Из последнего делением на $\|\boldsymbol{b}\| \cdot \|\boldsymbol{x}\|$ получаем искомую связь

$$\frac{\|\Delta \boldsymbol{x}\|}{\|\boldsymbol{x}\|} \le \|\boldsymbol{A}\| \cdot \left\|\boldsymbol{A}^{-1}\right\| \cdot \frac{\|\Delta \boldsymbol{b}\|}{\|\boldsymbol{b}\|}. \qquad (1.6)$$

Положительное число $\|\boldsymbol{A}\| \cdot \left\|\boldsymbol{A}^{-1}\right\|$ – коэффициент этой связи называют ***числом*** (***мерой***) ***обусловленности*** матрицы $\boldsymbol{A}$ и обозначают $\mathrm{cond}\,\boldsymbol{A}$ (от английского *conditioned* – «обусловленный»). Распространены также обозначения $\nu(\boldsymbol{A})$, $\mu(\boldsymbol{A})$, $\chi(\boldsymbol{A})$.

Аналогично зависит погрешность решения СЛАУ и от погрешности матрицы коэффициентов $\boldsymbol{A}$. Можно получить оценки для случая одновременного возмущения матрицы $\boldsymbol{A}$ и вектора $\boldsymbol{b}$.

Теорема об оценке погрешности решения по погрешностям входных данных. Пусть $\boldsymbol{x}$ – решение системы $\boldsymbol{Ax} = \boldsymbol{b}$, а $\boldsymbol{x}^*$ - решение системы $\boldsymbol{A}^* \boldsymbol{x}^* = \boldsymbol{b}^*$. Тогда $\delta(\boldsymbol{x}^*) \le cond(\boldsymbol{A}) \cdot (\delta(\boldsymbol{b}^*) + \delta(\boldsymbol{A}^*))$, где $cond(\boldsymbol{A}) = \|\boldsymbol{A}\| \cdot \left\|\boldsymbol{A}^{-1}\right\|$ – ***относительное число обусловленности системы***. Более точная зависимость описана в [1, с. 30].

Чем больше число обусловленности, тем сильнее сказывается на решении СЛАУ ошибка в исходных данных. Если $cond\ \boldsymbol{A} = \mathrm{O}(10^{p})$ и исходные данные имеют погрешность в q-м знаке после запятой, то независимо от способа решения СЛАУ в результате можно гарантировать не более $q - p$ знаков после запятой. $cond\ \boldsymbol{A}$ не может быть меньше 1

$$cond(\boldsymbol{A}) = \|\boldsymbol{A}\| \cdot \left\|\boldsymbol{A}^{-1}\right\| \ge \left\|\boldsymbol{A} \cdot \boldsymbol{A}^{-1}\right\| = \|\boldsymbol{E}\| = 1.$$

Если число $cond\ \boldsymbol{A}$ велико, то система считается плохо обусловленной, так как возможен сильный рост погрешности результата. Число обусловленности показывает, во сколько раз может возрасти относительная погрешность результата по сравнению с погрешностью исходных данных.

Решение СЛАУ на компьютере будет заведомо ложным (возможен неограниченный рост погрешности), если $cond(\boldsymbol{A}) \ge (macheps)^{-1}$ или даже $cond(\boldsymbol{A}) \ge (macheps)^{-0,5}$.

Пример 1.10. Вычисление норм матрицы и числа обусловленности матрицы в среде MathCAD представлено на рис. 1.3.

$$A := \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

$\|A\|_1 = \max(1+3, 2+4) = 6$ $\quad$ $\text{norm1}(A) = 6$

$\|A\|_e = \sqrt{1^2+2^2+3^2+4^2} = 5.477$ $\quad$ $\text{norme}(A) = 5.477$

$\|A\|_i = \max(1+2, 3+4) = 7$ $\quad$ $\text{normi}(A) = 7$

Число обусловленности связано с нормой матрицы и вычисляется по-разному для каждой из норм:

$$\text{norme}(A) \cdot \text{norme}\left(A^{-1}\right) = 15$$

$\text{conde}(A) = 15$ $\quad$ $\text{cond1}(A) = 21$ $\quad$ $\text{condi}(A) = 21$

Cond(A)=O(10^1) - хорошо обусловленная матрица.
Если элементы в матрице имеют q верных цифр после запятой, то при решении СЛАУ с такой матрицей верными будут q-1 цифра.

$$B := \begin{pmatrix} 1 & 2 \\ 3 & 6.01 \end{pmatrix}$$ $\quad$ $\text{conde}(B) = 5.012 \times 10^3$

$\text{cond1}(B) = 7.217 \times 10^3$ $\quad$ $\text{condi}(B) = 7.217 \times 10^3$

Cond(B)=O(10^3) - плохо обусловленная матрица. Ее строки определяют очень близкие уравнения. Если элементы в матрице имеют q верных цифр после запятой, то при решении СЛАУ с такой матрицей верными будут q-3 цифр.

Рис. 1.3. Вычисление норм матрицы и числа обусловленности матрицы в среде MathCAD

Определение погрешностей решения СЛАУ выполняется в практической работе № 4 [2, с.16].

Пример 1.11. Оценка числа обусловленности матрицы и определение погрешности решения СЛАУ в среде MathCAD представлено на рис. 1.4.

$$A := \begin{pmatrix} \sqrt{2} & 2 \\ 1.4 & 2 \end{pmatrix} \qquad b := \begin{pmatrix} 2-\sqrt{2} \\ 0.6 \end{pmatrix}$$

$x := \mathrm{lsolve}(A, b)$ -решение системы методом Гаусса $\qquad x = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$

Найдем относительное число обусловленности системы по формуле

$$\nu := \mathrm{norme}(A) \cdot \mathrm{norme}\left(A^{-1}\right) \qquad \nu = 420.725$$

Можно воспользоваться встроенной функцией: $\mathrm{conde}(A) = 420.725$

Число обусловленности показывает во сколько раз может возрасти погрешность решения.

Проведем вычислительный эксперимент.

Внесем погрешность в один элемент матрицы и в вектор правой части

$$A1 := \begin{pmatrix} \sqrt{2} & 2 \\ 1.41 & 2 \end{pmatrix} \qquad b1 := \begin{pmatrix} 2-\sqrt{2} \\ 0.61 \end{pmatrix}$$

Решим систему с возмущенной правой частью:

$x1 := \mathrm{lsolve}(A1, b1)$ $\qquad x1 = \begin{pmatrix} -5.747 \\ 4.356 \end{pmatrix}$

Найдем величины погрешностей

$$\delta A1 := \frac{\mathrm{norm2}(A - A1)}{\mathrm{norm2}(A)} \qquad \delta b1 := \frac{|b1 - b|}{|b|}$$

Величина относительной погрешности входных данных : $\delta A1 = 2.892 \times 10^{-3} \qquad \delta b1 = 0.012$

Теоретическая оценка погрешности: $(\delta A1 + \delta b1) \cdot \mathrm{conde}(A) = 6.234$

Практическая оценка относительной погрешности решения: $\delta 1 := \frac{|x - x1|}{|x|} \qquad \delta 1 = 4.111$

$\frac{\delta 1}{\delta A1 + \delta b1} = 277.426$ Погрешность возросла в 277 раз.

Рис. 1.4. Оценка числа обусловленности матрицы и погрешности решения СЛАУ

Контрольные вопросы

1. Типы погрешностей, возникающих при численном решении задач.

2. Какие погрешности для вычислителя являются безусловными (неустранимыми), а какие условными (устранимыми)?

3. Что называется абсолютной и относительной погрешностями приближенных чисел?

4. Сформулируйте правила округления приближенных чисел: по дополнению и усечением.

5. Дайте определение значащей цифры числа. Приведите примеры.

6. Сформулируйте определение верной цифры числа. Приведите примеры.

7. Какова оценка погрешностей арифметических действий?

8. Как оцениваются погрешности вычисления значения функции многих переменных?

9. Как в соответствии с принципом Крылова записывается приближенное число?

10. Приведите правила Крылова для выполнения арифметических действий над приближенными числами.

11. Что называется машинной бесконечностью, машинным нулем и машинным эпсилоном?

12. Какова оценка погрешностей арифметических действий над машинными числами с плавающей точкой?

13. Какие свойства машинных чисел вам известны?

14. Дайте определение корректной задачи по Адамару.

15. Как оценивается абсолютная погрешность корня скалярного уравнения с приближенными коэффициентами?

16. Что называется коэффициентами чувствительности при численном решении скалярных уравнений с приближенными коэффициентами?

17. Векторно-матричная форма записи СЛАУ.

18. Дайте понятие и вычисление векторных норм.

19. Приведите понятие и вычисление матричных норм.

20. Что понимается под абсолютной и относительной погрешностями приближенных вектора и матрицы?

21. Как оценивается точность решения СЛАУ с приближенными значениями свободных членов, матрицы коэффициентов?

22. Понятие и вычисление числа обусловленности матрицы.

23. Охарактеризуйте связь числа обусловленности матрицы коэффициентов с достоверностью и точностью решения СЛАУ.

2. ЧИСЛЕННЫЕ МЕТОДЫ ЛИНЕЙНОЙ АЛГЕБРЫ

По оценкам специалистов 75 % всех расчетных математических задач приходится на решение систем линейных алгебраических уравнений. Это не удивительно, так как математические модели тех или иных явлений или процессов либо сразу строятся как линейные алгебраические, либо сводятся к таковым посредством дискретизации и/или линеаризации. Поэтому трудно переоценить роль, которую играет выбор эффективного (в том или ином смысле) способа решения СЛАУ. Современная вычислительная математика располагает большим арсеналом методов, а математическое обеспечение ЭВМ – многими пакетами прикладных программ для решения СЛАУ. Чтобы ориентироваться среди методов и программ, делать оптимальный выбор, нужно разбираться в основах построения методов и алгоритмов, учитывающих специфику постановок задач, знать их сильные и слабые стороны и границы применимости.

Все методы решения линейных алгебраических задач (наряду с задачей решения СЛАУ, это и вычисление определителей, и обращение матриц, и задачи на собственные значения) можно разбить на два класса: прямые и итерационные.

Метод решения задачи называют ***прямым***, если он позволяет получить решение после выполнения конечного числа арифметических операций. Если операции реализуются точно (без округлений), то и решение также будет точным (в связи с чем к классу прямых методов применяют еще название ***точные методы***). Метод решения задачи называют ***итерационным***, если точное решение может быть получено лишь в результате бесконечного повторения единообразных (как правило, простых) действий.

2.1. Решение систем линейных алгебраических уравнений прямыми методами

Условимся говорить о численном решении таких СЛАУ, у которых число уравнений совпадает с числом вещественных неизвестных, а решение предполагается существующим и единственным.

Итак, изучается вопрос о численном решении систем вида

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \ldots + a_{1m}x_m = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \ldots + a_{2m}x_m = b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \ldots + a_{mm}x_m = b_{m,} \end{cases} \qquad (2.1)$$

или иначе векторно-матричных уравнений

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad (2.1a)$$

размерностью m, где $\boldsymbol{b} = (b_1, b_2, ..., b_m)^T$ – вектор свободных членов, $\boldsymbol{x} = (x_1, x_2, ..., x_m)^T$ – вектор неизвестных (вектор-решение) с вещественными координатами, $\boldsymbol{A} = (a_{ij})_{i,j=1,m}$ – вещественная $m \times m$-матрица коэффициентов данной системы.

Эффективность способов решения системы (2.1) во многом зависит от структуры и свойств матрицы $\boldsymbol{A}$: размера, обусловленности, симметричности, заполненности (т.е. соотношением между числом ненулевых и нулевых элементов), специфики расположения ненулевых элементов в матрице и др.

Будем предполагать, что матрица системы $\boldsymbol{A}$ задана и является невырожденной. Известно, что в этом случае решение системы существует единственно и устойчиво по входным данным. Решение может быть получено по *формулам Крамера*

$$x_i = \frac{\det \boldsymbol{A}_i}{\det \boldsymbol{A}} \quad (i = 1,2,...m),$$

где матрица $\boldsymbol{A}_i$ образуется из матрицы $\boldsymbol{A}$ заменой ее i-го столбца столбцом свободных членов.

Пример 2.1. Решение СЛАУ по формулам Крамера [2, с.19].

Если при реализации этих формул определители вычисляются понижением порядка на основе разложения по элементам какой-нибудь строки или столбца матрицы, то на вычисление определителя m-го порядка будет затрачиваться $m!$ операций умножения. Факториальный рост количества арифметических операций (и, вообще, очень быстрый рост) с увеличением размерности задачи называют «проклятьем размерности». Например, оценив величину $100! \approx 10^{158}$, приходим к выводу о том, что пока системы сотого порядка в принципе не могут быть решены по формулам Крамера (рис. 2.1). Заметим при этом, что, во-первых, метод Крамера будет неустойчив, т.е. погрешности округлений будут катастрофически нарастать, во-вторых, размерность $m = 100$ для современных задач не так и велика: довольно часто решаются системы с сотнями и с тысячами неизвестных.

$\frac{\frac{10^{158}}{\left(280.6 \cdot 10^{12}\right)}}{60 \cdot 60 \cdot 24 \cdot 365} = 1.13 \times 10^{136}$	лет понадобится суперкомпьютеру BlueGene/L производительностью 280.6 TFLOPS

Рис 2.1. «Проклятье размерности»

Нахождение решения векторно-матричного уравнения (2.1а) по формуле $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$, если осуществлять вычисление обратной матрицы с помощью союзной матрицы, т.е. через алгебраические дополнения, фактически равнозначно применению формул Крамера и также практически непригодно по упомянутым выше причинам для вычислительных целей.

К прямым методам решения, применяемым при компьютерных вычислениях, относятся метод Гаусса и его модификации, метод Холецкого и метод прогонки. На практике прямые методы используют для решения систем размерностью $m \le 200$.

2.1.1. Метод Гаусса

Наиболее известным и популярным способом решения линейных систем вида (2.1) является метод Гаусса. Суть его проста – это последовательное исключение неизвестных. Решение получают в два этапа: на первом осуществляется приведение исходной системы уравнений с помощью преобразований к эквивалентной системе с верхней треугольной матрицей (*прямой ход*); на втором, т.е. в *обратном ходе* (снизу вверх), находятся последовательно все неизвестные системы.

Прямой ход состоит из m-1 шагов исключения, сводящихся к умножению всех членов уравнения на постоянное число, сложению уравнений, выражению отдельных неизвестных через другие.

На первом шаге исключим неизвестное x_1 из уравнений с номерами $i = 2,3,\ldots m.$ Предположим, что $a_{11} \neq 0$. Будем называть его *ведущим элементом* 1-го шага. Найдем величины $\mu_{i1} = \frac{a_{i1}}{a_{11}}$, $i = 2,3,\ldots m.$, называемые множителями 1-го шага. Вычтем последовательно из второго, третьего, ..., m-го уравнений системы первое уравнение, умноженное соответственно на

$\mu_{21},\mu_{31},...,\mu_{m1}$. В результате 1-го шага получим эквивалентную систему уравнений

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + ... + a_{1m}x_m = b_1 \\ 0x_1 + a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 + ... + a_{2m}^{(1)}x_m = b_2^{(1)} \\ ..... \\ 0x_1 + a_{m2}^{(1)}x_2 + a_{m3}^{(1)}x_3 + ... + a_{mm}^{(1)}x_m = b_m^{(1)}. \end{cases}$$

Аналогично проводятся остальные шаги. Опишем очередной *k*-й шаг. Предположим, что ведущий элемент $a_{kk} \neq 0$. Вычислим множители *k*-го шага: $\mu_{ik} = \dfrac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}}$, $i = k+1,\ldots,m$ и вычтем последовательно из $(k+1)$-го, ..., *m*-го уравнений системы *k*-е уравнение, умноженное на соответствующие μ_{ik}.

После $(m-1)$-го шага исключения получим систему уравнений

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + ... + a_{1m}x_m = b_1 \\ 0x_1 + a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 + ... + a_{2m}^{(1)}x_m = b_2^{(1)} \\ ..... \\ 0x_1 + 0x_2 + 0x_3 + ... + a_{mm}^{(m-1)}x_m = b_m^{(m-1)}, \end{cases}$$

матрица которой является верхней треугольной. На этом вычисления прямого хода заканчиваются.

Обратный ход. Из последнего уравнения системы находим x_m. Подставляя найденное значение x_m в предпоследнее уравнение, получим x_{m-1}. Далее последовательно находим неизвестные $x_{m-2},...,x_2,x_1$.

Учитывая цикличность выполняемых операций, а также нецелесообразность хранения промежуточных результатов, запишем **простой алгоритм решения СЛАУ методом Гаусса**:

1. для $k = 1,2,\ldots,m-1$,
2. для $i = k+1,...,m$:
3. $\mu_{ik} := a_{ik} \big/ a_{kk}$,
4. $b_i := b_i - \mu_{ik} b_k$;
5. для $j = k+1,...,m$:

6. $a_{ij} := a_{ij} - \mu_{ik} a_{kj}$.
7. $x_m := b_m / a_{mm}$;
8. для $k = m-1,...,2,1$:
9. $x_k := \left(b_k - \sum_{j=k+1}^{m} a_{kj} x_j \right) / a_{kk}$.

Разработка собственных функций в среде MathCAD выполняется с помощью шаблонов операторов, расположенных на панели инструментов «Программирование» (рис. 2.2).

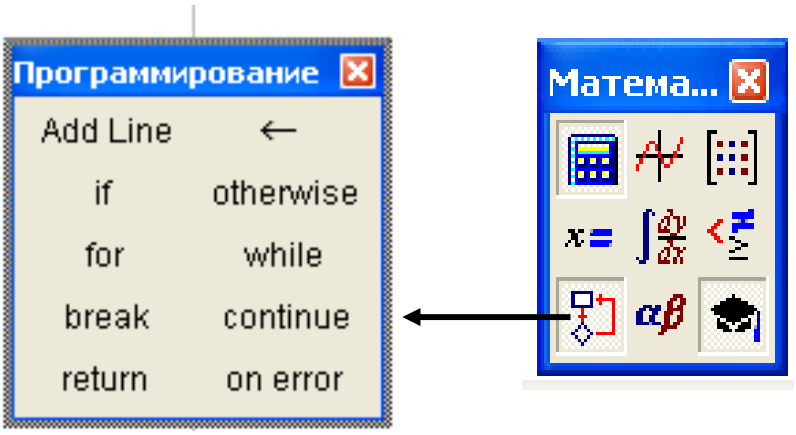


Рис 2.2. Панель шаблонов операторов

На рис. 2.3 приведена соответствующая вышеприведенному алгоритму функция и решение СЛАУ с ее помощью.

В методе Гаусса для вычисления масштабирующих множителей требуется делить на ведущие элементы каждого шага. Так как реальные машинные вычисления производятся не с точными, а с усеченными числами, т.е. неизбежны ошибки округления, то выполнение алгоритма может прекратиться или привести к неверным результатам, если элемент окажется равен нулю или близок к нулю. Другими словами, в этом случае возможен неконтролируемый рост погрешности. Поэтому часто применяют модификации метода Гаусса, обладающие лучшими вычислительными свойствами.

Метод Гаусса с выбором главного элемента по столбцу (схема частичного выбора). На каждом шаге прямого хода уравнения системы переставляются так, чтобы деление производилось на наибольший по модулю в данном столбце элемент. В этом случае все масштабирующие

множители по модулю меньше единицы и схема обладает вычислительной устойчивостью.

ORIGIN:= 1

$$A := \begin{pmatrix} 3 & 4 & -9 & 5 \\ -15 & -12 & 50 & -16 \\ -27 & -36 & 73 & 8 \\ 9 & 12 & -10 & -16 \end{pmatrix} \qquad b := \begin{pmatrix} -14 \\ 44 \\ 142 \\ -76 \end{pmatrix}$$

```
gs(A, b) := | m ← rows(A)
            | for k ∈ 1.. m − 1
            |   for i ∈ k + 1.. m
            |     μ_{i,k} ← A_{i,k} / A_{k,k}
            |     b_i ← b_i − μ_{i,k} · b_k
            |     for j ∈ k.. m
            |       A_{i,j} ← A_{i,j} − μ_{i,k} · A_{k,j}
            | x_m ← b_m / A_{m,m}
            | for k ∈ m − 1.. 1
            |   x_k ← ( b_k − Σ_{j=k+1}^{m} A_{k,j} · x_j ) / A_{k,j}
            | x
```

Вставка * Найти $n \ge k$ такое, что $|a_{nk}| = \max\limits_{i \ge k}\{|a_{ik}|\}$;
Если $|a_{nk}| = 0$,
остановить («однозначного решения нет»),
иначе поменять местами b_k и b_n, a_{kj} и a_{nj} при всех $j = k, \ldots, m$.

$$gs(A, b) = \begin{pmatrix} -8 \\ -2 \\ -2 \\ 0 \end{pmatrix}$$

Рис. 2.3. Решение СЛАУ методом Гаусса

Частичное упорядочивание по столбцам требует внесения в алгоритм следующих изменений: в начало тела цикла по *k* сделать **вставку *** между строками 1 и 2:

1а. Найти $n \ge k$ такое, что $|a_{nk}| = \max\limits_{i \ge k}\{|a_{ik}|\}$;

1б. Если $|a_{nk}| = 0$,

остановить работу алгоритма («однозначного решения нет»);

иначе поменять местами b_k и b_n, a_{kj} и a_{nj} при всех $j = k,...,m.$

Более разумным, наверное, является сравнение $|a_{nk}|$ не с нулем, а с некоторым малым $\varepsilon > 0$, задаваемым вычислителем в зависимости от различных априорных соображений. Счет останавливается или берется под особый контроль, если окажется $|a_{nk}| < \varepsilon$. Фактически в процессе решения проводится алгоритмическое исследование системы на однозначную разрешимость.

Устойчивость алгоритма к погрешностям исходных данных и результатов промежуточных вычислений можно еще усилить, если выполнять деление на каждом шаге на элемент, наибольший по модулю во всей матрице, преобразуемой на данном шаге подсистемы. Такая модификация метода Гаусса, называемая ***методом главных элементов***, применяется довольно редко, поскольку сильно усложняет алгоритм. Усложнение связано как с необходимостью осуществления двумерного поиска главных элементов, так и с необходимостью запоминать номера столбцов, откуда берутся эти элементы (перестановка столбцов означает как бы переобозначение неизвестных, в связи с чем требуется обратная замена).

Общее число операций при вычислении методом Гаусса пропорционально $\frac{2}{3}m^3$. Система из 1000 уравнений при быстродействии компьютера 1 MFLOPS (10^6 оп/с) будет решена за время от 10 мин до 1 ч.

Пример 2.2. Решение системы методом Гаусса с выбором главного элемента по столбцу [2, с.25].

Как уже было сказано ранее, решения СЛАУ можно получать с помощью определителей или обратных матриц. Но более эффективно поступать наоборот: вычислять определители и обращать матрицы в рамках метода Гаусса решения линейных систем. Выполняемое преобразование матрицы $\boldsymbol{A}$ к треугольному виду не изменяет ее определителя. Учитывая, что определитель треугольной матрицы равен произведению диагональных элементов, имеем: *определитель матрицы* $\boldsymbol{A}$ *равен произведению всех ведущих элементов при ее преобразовании методом Гаусса*.

2.1.2. LU-разложение матриц

Представим матрицу $\boldsymbol{A}$ в виде произведения нижней треугольной матрицы $\boldsymbol{L}$ и верхней треугольной $\boldsymbol{U}$: $\boldsymbol{A} = \boldsymbol{LU}$.

Введем в рассмотрение матрицы

$$\boldsymbol{L} = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 \\ \mu_{21} & 1 & \dots & 0 & 0 \\ \mu_{31} & \mu_{32} & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \mu_{m1} & \mu_{m2} & \mu_{m3} & \dots & 1 \end{pmatrix} \quad \text{и} \quad \boldsymbol{U} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m-1} & a_{1m} \\ 0 & a_{22}^{(1)} & \dots & a_{2m-1}^{(1)} & a_{2m}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & a_{3m-1}^{(2)} & a_{3m}^{(2)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & a_{mm}^{(m-1)} \end{pmatrix}$$

Это и есть разложение матрицы на множители, которое возможно, если все главные миноры квадратной матрицы $\boldsymbol{A}$ отличны от нуля. Если элементы диагонали одной из матриц $\boldsymbol{L}$ или $\boldsymbol{U}$ фиксированы (ненулевые), то такое разложение единственно. Элементы матриц $\boldsymbol{L}$ и $\boldsymbol{U}$ вычисляются по шагам аналогично алгоритму метода Гаусса.

Пример 2.3. Разложение матрицы **A** на множители. Решение примера в среде MathCAD представлено в файле *R5.mcd* электронного комплекса.

Если матрица $\boldsymbol{A}$ исходной системы разложена, то вместо $\boldsymbol{Ax} = \boldsymbol{b}$ можно записать эквивалентное уравнение $\boldsymbol{LUx} = \boldsymbol{b}$. Введя вектор вспомогательных переменных $\boldsymbol{y} = (y_1, y_2, ..., y_m)^T$, последнее можно переписать в виде системы

$$\begin{cases} \boldsymbol{Ly} = \boldsymbol{b} \\ \boldsymbol{Ux} = \boldsymbol{y}. \end{cases}$$

Таким образом, решение данной системы с квадратной матрицей коэффициентов свелось к последовательному решению двух систем с треугольными матрицами коэффициентов.

Пример 2.4. Решение системы уравнений с помощью $\boldsymbol{LU}$-разложения матрицы приведено в файле *R5.mcd* электронного комплекса.

При невыполнении условия об отличии главных миноров квадратной матрицы $\boldsymbol{A}$ от нуля для невырожденных матриц применяют перестановку строк таким образом, что матрица $\boldsymbol{PA}$ имеет $\boldsymbol{LU}$-разложение:

$$\boldsymbol{PA} = \boldsymbol{LU},$$

где $\boldsymbol{P}$ - матрица перестановок.

Например, для системы размерностью 3 при перестановке 1-го и 3-го уравнений матрица перестановок будет

$$\boldsymbol{P} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

2.1.3. Метод Холецкого

Если матрица системы является симметричной $(a_{ij} = a_{ji})$ и положительно определенной, то для решения системы применяют ***метод Холецкого (метод квадратных корней)***. В основе метода лежит алгоритм специального $\boldsymbol{LU}$-разложения матрицы $\boldsymbol{A}$, в результате чего она приводится к виду $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$. Если разложение получено, то как и в методе $\boldsymbol{LU}$-разложения, решение системы $\boldsymbol{L}\boldsymbol{L}^T \cdot \boldsymbol{x} = \boldsymbol{b}$ сводится к последовательному решению двух систем с треугольными матрицами:

$$\begin{cases} \boldsymbol{L}\boldsymbol{y} = \boldsymbol{b} \\ \boldsymbol{L}^T \boldsymbol{x} = \boldsymbol{y}. \end{cases}$$

Для нахождения коэффициентов матрицы $\boldsymbol{L}$ неизвестные коэффициенты матрицы $\boldsymbol{L}\boldsymbol{L}^T$ приравнивают соответствующим элементам матрицы $\boldsymbol{A}$. Затем последовательно находят требуемые коэффициенты по формулам

$$l_{11} = \sqrt{a_{11}}, \quad l_{i1} = a_{i1}/l_{11} \quad \text{для} \quad i = 2,3,\ldots,m,$$

$$l_{22} = \sqrt{a_{22} - l_{21}^2}, \quad l_{i2} = (a_{i2} - l_{i1}l_{21})/l_{22} \quad \text{для} \quad i = 3,4,..,m,$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots...$$

$$l_{kk} = \sqrt{a_{kk} - l_{k1}^2 - l_{k2}^2 - \ldots - l_{kk-1}^2}, \quad l_{ik} = (a_{ik} - l_{i1}l_{k1} - l_{i2}l_{k2} - \ldots - l_{ik-1}l_{kk-1})/l_{kk}$$

$$\text{для} \quad i = k+1,\ldots,m,$$

$$l_{mm} = \sqrt{a_{mm} - l_{m1}^2 - l_{m2}^2 - \ldots - l_{mm-1}^2}.$$

Пример 2.5. Решение системы методом Холецкого.

Пример 2.6. Разложение матриц на множители встроенными функциями MathCAD.

Пример 2.7. Решение СЛАУ **LU**-разложением встроенными функциями MathCAD.

Решение примеров в среде MathCAD представлено в файле *R5.mcd* электронного комплекса.

2.1.4. Метод прогонки

Если матрица системы является разреженной, то есть содержит большое число нулевых элементов, то применяют еще одну модификацию метода Гаусса – метод прогонки. Среди таких систем выделим системы с матрицами ленточной структуры, в которых ненулевые элементы располагаются на главной диагонали и на нескольких побочных диагоналях.

Рассмотрим наиболее простой случай *ленточных систем*, к которым сводится решение задач сплайн-интерполяции функций, дискретизации краевых задач для дифференциальных уравнений методами конечных разностей, конечных элементов и др. А именно, будем искать решение такой системы, каждое уравнение которой связывает три «соседних» неизвестных

$$a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i,$$

где $i = 1,2,...,m;\ \ a_1 = 0,\ \ c_m = 0$. Такие уравнения называют *трехточечными разностными уравнениями второго порядка*, а матрицу коэффициентов называют *трехдиагональной матрицей*

$$\begin{cases} b_1 x_1 + c_1 x_2 = d_1 \\ a_2 x_1 + b_2 x_2 + c_2 x_3 = d_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m-1} x_{m-2} + b_{m-1} x_{m-1} + c_{m-1} x_m = d_{m-1} \\ a_m x_{m-1} + b_m x_m = d_m \end{cases}$$

Цель первого этапа решения (*прямой прогонки*) – избавиться от ненулевых элементов в поддиагональной части матрицы. Преобразуем первое уравнение системы к виду

$$x_1 = \alpha_1 x_2 + \beta_1,$$

где $\alpha_1 = -c_1/b_1$, $\beta_1 = d_1/b_1$ – *прогоночные коэффициенты* 1-го шага.

Подставим полученное выражение во второе уравнение системы и преобразуем его к виду $x_2 = \alpha_2 x_3 + \beta_2$ и т.д. На *i*-м шаге уравнение преобразуется к виду $x_i = \alpha_i x_{i+1} + \beta_i$, где $\alpha_i = -c_i/(b_i + a_i\alpha_{i-1})$, $\beta_i = (d_i - a_i\beta_{i-1})/(b_i + a_i\alpha_{i-1})$. На *m*-м шаге подстановка в последнее уравнение выражения $x_{m-1} = \alpha_{m-1} x_m + \beta_{m-1}$ дает возможность определить значение x_m (начало второго этапа – *обратной прогонки*):

$$x_m = \beta_m = (d_m - \alpha_m\beta_{m-1})/(b_m + a_m\alpha_{m-1}).$$

Значения остальных неизвестных находятся по формулам $x_i = \alpha_i x_{i+1} + \beta_i, \quad i = m-1, m-2, ..., 1.$

Прогонка *корректна*, если знаменатели прогоночных коэффициентов α_i, β_i не обращаются в нуль, и *устойчива*, если $|\alpha_i| < 1$ при всех $i \in \{1,2,...,m\}$. Для этого коэффициенты a_i и c_i при $i = 2,3,...,m-1$ должны быть отличны от нуля; $|b_i| > |a_i| + |c_i|$ для всех $i = 1,2,...,m$.

Пример 2.8. Решение системы методом прогонки. Решение примера в среде MathCAD представлено в файле *R5.mcd* электронного комплекса.

2.1.5. Замечания к применению прямых методов

Прямые методы приводят к точному решению СЛАУ при точном выполнении предусматриваемых соответствующими алгоритмами арифметических операций (без округлений). Реальные же вычисления базируются на арифметике машинных (т.е. усеченных до определенного количества разрядов) чисел. Поэтому практически вместо точного решения СЛАУ прямой метод дает приближенное решение. 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. Системы с плохо обусловленными матрицами коэффициентов нецелесообразно решать прямыми методами вследствие возможности появления очень больших ошибок.

Одним из факторов, предопределяющих выбор метода решения конкретной задачи, является вычислительная эффективность метода. Особенностью прямых методов является то, что здесь можно точно подсчитать требуемое количество арифметических операций. *Арифметическую сложность метода* можно оценить вычислительными затратами на операции умножения и деления, которые составляют величину $O\left(\frac{m^3}{3}\right)$ в методе Гаусса, $O\left(\frac{m^3}{6}\right)$ в методе Холецкого, $O(5m)$ в методе прогонки.

Ознакомление с решением систем линейных алгебраических уравнений прямыми методами предполагается при выполнении практической работы № 5 [2, с. 24] .

2.2. Решение систем линейных алгебраических уравнений итерационными методами

Метод решения задачи называют ***итерационным***, если точное решение может быть получено лишь в результате бесконечного повторения единообразных (как правило, простых) действий. Решение системы ищется как предел бесконечного вычислительного процесса, позволяющего по уже найденным приближениям к решению построить следующее, более точное приближение (поэтому еще эти методы называют ***приближенными***). Если эта бесконечная последовательность действий сходится к решению задачи, то говорят, что *итерационный процесс сходится*.

Важной чертой таких методов является их самоисправляемость и простота реализации. Если в прямых 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.

2.2.1. Метод простых итераций

В этом методе исходную систему уравнений $\boldsymbol{Ax} = \boldsymbol{b}$ предварительно приводят к виду

$$\boldsymbol{x} = \boldsymbol{Bx} + \boldsymbol{c}, \qquad (2.2)$$

где $\boldsymbol{x}$ – тот же вектор неизвестных, $\boldsymbol{B}$ и $\boldsymbol{c}$ – некоторые новые матрица и вектор соответственно, $\det \boldsymbol{B} \neq 0$.

Выбрав начальное приближение к решению системы уравнений $\boldsymbol{x}^{(0)} = \left(x_1^{(0)},...,x_m^{(0)}\right)^T$ (верхним индексом обозначен номер итерации), подставляем его в (2.2) и находим значения неизвестных системы на 1-й итерации: $\boldsymbol{x}^{(1)} = \boldsymbol{Bx}^{(0)} + \boldsymbol{c}$. Затем аналогично на 2-й, 3-й и т.д. В общем случае алгоритм записывается следующим образом: $\boldsymbol{x}^{(k)} = \boldsymbol{Bx}^{(k-1)} + \boldsymbol{c}$. Последовательность $\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)},...\boldsymbol{x}^{(k)},...$ будет сходиться к решению $\boldsymbol{x}^*$, если $\|\boldsymbol{B}\|<1$.

Теорема 2.1. Пусть $\|\boldsymbol{B}\| \le q < 1$. Тогда при любом начальном векторе $\boldsymbol{x}^{(0)}$ метод простых итераций сходится к единственному решению $\boldsymbol{x}^*$ задачи (2.2) и при всех $k \in N$ справедливы оценки погрешности:

1) $\left\|\boldsymbol{x}^* - \boldsymbol{x}^{(k)}\right\| \le \dfrac{q}{1-q}\left\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\right\|$ - ***апостериорная***, т.е полученная «из опыта», ею можно пользоваться лишь после проведения k-й итерации;

2) $\left\|\boldsymbol{x}^* - \boldsymbol{x}^{(k)}\right\| \le \dfrac{q^k}{1-q}\left\|\boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)}\right\|$ - ***априорная***, т.е полученная «до опыта», ею можно пользоваться до начала счета.

Априорная оценка позволяет подсчитывать заранее число итераций k, достаточное для получения решения $\boldsymbol{x}^*$ с заданной точностью ε (в смысле допустимого уровня абсолютных погрешностей) при выбранном начальном векторе $\boldsymbol{x}^{(0)}$. Для этого нужно найти наименьшее целое решение неравенства $\dfrac{q^k}{1-q}\left\|\boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)}\right\| \le \varepsilon$ относительно переменной k. Апостериорной же оценкой удобно пользоваться непосредственно в процессе вычислений и останавливать этот процесс, как только выполнится неравенство $\left\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\right\| \le \dfrac{1-q}{q}\varepsilon$.

Неравенство $\left\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\right\| \le \varepsilon$ будет гарантией выполнения неравенства $\left\|\boldsymbol{x}^* - \boldsymbol{x}^{(k)}\right\| \le \varepsilon$ только в том случае, когда $q \le \dfrac{1}{2}$.

Как выбрать начальное приближение? Очевидно, итераций потребуется тем меньше, чем ближе $\boldsymbol{x}^{(0)}$ к $\boldsymbol{x}^*$. Если нет никакой дополнительной информации о решении задачи (2.2) (например, может быть известным решение близкой задачи или грубое решение данной задачи), то обычно в качестве $\boldsymbol{x}^{(0)}$ выбирают нулевые значения или столбец свободных членов $\boldsymbol{c}$. Во втором случае погрешность после k-го приближения будет оцениваться величиной $\left\|\boldsymbol{x}^* - \boldsymbol{x}^{(k)}\right\| \le \dfrac{\|\boldsymbol{c}\|}{1-q} q^{k+1}$. Покажем это, подставив нормы матрицы $\boldsymbol{B}$ и вектора $\boldsymbol{c}$ в выражение априорной погрешности $\left\|\boldsymbol{x}^* - \boldsymbol{x}^{(k)}\right\| \le \dfrac{q^k}{1-q}\left\|\boldsymbol{x}^{(1)} - \boldsymbol{x}^{(0)}\right\|$:

$$\frac{q^k}{1-q}\left\|\boldsymbol{x}^{(1)}-\boldsymbol{x}^{(0)}\right\|=\frac{\|\boldsymbol{B}\|^k}{1-\|\boldsymbol{B}\|}\left\|\boldsymbol{B}\boldsymbol{x}^{(0)}+\boldsymbol{c}-\boldsymbol{x}^{(0)}\right\|=\frac{\|\boldsymbol{B}\|^k}{1-\|\boldsymbol{B}\|}\|\boldsymbol{B}\boldsymbol{c}+\boldsymbol{c}-\boldsymbol{c}\|=$$

$$=\frac{\|\boldsymbol{B}\|^k}{1-\|\boldsymbol{B}\|}\|\boldsymbol{B}\boldsymbol{c}\|=\frac{\|\boldsymbol{B}\|^{k+1}\cdot\|\boldsymbol{c}\|}{1-\|\boldsymbol{B}\|}=\frac{q^{k+1}\cdot\|\boldsymbol{c}\|}{1-q}.$$

Теперь можно подсчитывать число итераций k, достаточное для получения решения $\boldsymbol{x}^*$ с заданной точностью ε (в смысле допустимого уровня абсолютных погрешностей) при выбранном начальном векторе $\boldsymbol{x}^{(0)}=\boldsymbol{c}$. Для этого нужно найти наименьшее целое решение неравенства $\frac{q^{k+1}\cdot\|\boldsymbol{c}\|}{1-q}\le\varepsilon$ относительно переменной k

$$q^{k+1}\le\varepsilon\frac{1-q}{\|\boldsymbol{c}\|};$$

$$k=\log_q\frac{\varepsilon(1-q)}{\|\boldsymbol{c}\|}=\frac{\ln\frac{\varepsilon(1-q)}{\|\boldsymbol{c}\|}}{\ln q}-1.$$

Пример 2.9. Решение системы линейных уравнений методом простых итераций [2, с. 29].

2.2.2. Метод Якоби

Самый простой способ приведения системы к виду, удобному для итерации, состоит в следующем: из первого уравнения системы выразим неизвестное x_1, из второго уравнения системы выразим x_2, и т. д. Основанный на таком приведении системы $\boldsymbol{Ax}=\boldsymbol{b}$ к виду (2.2) метод простых итераций называют ***методом Якоби***. В результате получим систему уравнений с матрицей $\boldsymbol{B}$, в которой на главной диагонали стоят нулевые элементы, а остальные элементы вычисляются по формулам

$$B_{ij}=-a_{ij}/a_{ii},\ i,j=1,2,...,m.$$

Компоненты вектора $\boldsymbol{c}$ вычисляются по формулам

$$c_i=b_i/a_{ii},\ i=1,2,...,m.$$

Расчетная формула метода простой итерации имеет вид

$$\boldsymbol{x}^{(k)}=\boldsymbol{B}\boldsymbol{x}^{(k-1)}+\boldsymbol{c}$$

или в покоординатной форме записи выглядит так:

$$x_i^{(k)} = B_{i1}x_1^{(k-1)} + B_{i2}x_2^{(k-1)} + ...B_{ii-1}x_{i-1}^{(k-1)} + B_{ii+1}x_{i+1}^{(k-1)} + ... + B_{im}x_m^{(k-1)} + c_i,$$

$i = 1, 2, ..., m$.

Критерий окончания итераций в методе Якоби имеет вид

$$\left\| \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)} \right\| \le \varepsilon_1,$$

где $\varepsilon_1 = \dfrac{1-q}{q}\varepsilon$. Если $q \le \dfrac{1}{2}$, то можно применять более простой критерий окончания итераций $\left\| \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)} \right\| \le \varepsilon$.

Приведем решение системы из трех линейных уравнений методом Якоби в общем виде. Исходная система

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3. \end{cases}$$

Выразим неизвестные из соответствующих уравнений

$$\begin{cases} x_1 = -\dfrac{a_{12}}{a_{11}}x_2 - \dfrac{a_{13}}{a_{11}}x_3 + \dfrac{b_1}{a_{11}} \\ x_2 = -\dfrac{a_{21}}{a_{22}}x_2 - \dfrac{a_{23}}{a_{22}}x_3 + \dfrac{b_2}{a_{22}} \\ x_3 = -\dfrac{a_{31}}{a_{33}}x_2 - \dfrac{a_{32}}{a_{33}}x_3 + \dfrac{b_1}{a_{33}}. \end{cases}$$

Получили вычислительную схему

$$\begin{cases} x_1^{(k)} = 0 \cdot x_1^{(k-1)} + B_{12}x_2^{(k-1)} + B_{13}x_3^{(k-1)} + c_1 \\ x_2^{(k)} = B_{21}x_1^{(k-1)} + 0 \cdot x_2^{(k-1)} + B_{23}x_3^{(k-1)} + c_2 \\ x_3^{(k)} = B_{31}x_1^{(k-1)} + B_{32}x_2^{(k-1)} + 0 \cdot x_3^{(k-1)} + c_3. \end{cases}$$

Пример 2.10. Решение СЛАУ методом Якоби [2, с. 31].

2.2.3. Метод Зейделя

Метод отличается от метода простых итераций прежде всего в немедленном вводе в вычисления каждого из полученных исправленных значений неизвестных. Метод можно рассматривать как модификацию метода

Якоби. Основная идея состоит в том, что при вычислении очередного k-го приближения к неизвестному x_i при i >1 используют уже найденные k-е приближения к неизвестным $x_1, x_2, ..., x_{i-1}$, а не (k -1)-е приближение, как в методе Якоби. Расчетная формула метода в покоординатной форме записи выглядит так:

$$x_i^k = B_{i1}x_1^{(k)} + B_{i2}x_2^{(k)} + ... + B_{i,i-1}x_{i-1}^{(k)} + B_{i,i+1}x_{i+1}^{(k-1)} + ... + B_{im}x_m^{(k-1)} + c_i,$$

i = 1, 2, ..., m.

Для рассмотренной в п. 2.2.3 системы вычислительная схема будет следующей:

$$\begin{cases} x_1^{(k)} = 0 \cdot x_1^{(k-1)} + B_{12}x_2^{(k-1)} + B_{13}x_3^{(k-1)} + c_1 \\ x_2^{(k)} = B_{21}x_1^{(k)} + 0 \cdot x_2^{(k-1)} + B_{23}x_3^{(k-1)} + c_2 \\ x_3^{(k)} = B_{31}x_1^{(k)} + B_{32}x_2^{(k)} + 0 \cdot x_3^{(k-1)} + c_3. \end{cases}$$

Условия сходимости и критерий окончания итераций можно взять такими же, как в методе Якоби.

Пример 2.11. Решение систем линейных уравнений методом Зейделя [2, с. 32].

Пусть матрица системы уравнений ***A*** – симметричная и положительно определенная. Тогда при любом выборе начального приближения метод Зейделя сходится. Дополнительных условий на малость нормы некоторой матрицы здесь не накладывается.

2.2.4. О других итерационных методах решения СЛАУ

В случаях невыполнения условия $q < 1$ вышеперечисленные методы неэффективны и даже малонадежны ввиду медленной сходимости. Кроме того, разрабатываются свои методы для систем, обладающих какими-либо особенностями. Для иллюстрации широких возможностей вычислителей по выбору итерационных методов перечислим их названия:

1. Методы *релаксации* (как обобщение метода Зейделя):
   1.1. полной;
   1.2. последовательной верхней;
   1.3. последовательной нижней.
2. Методы *установки*:
   2.1. двухслойные итерационные методы.

2.1.1. стационарные;

2.1.2. нестационарные;

2.2. трехслойные итерационные методы.

3. Методы *вариационного типа*:
метод *сопряженных градиентов* (МСГ).

4. Метод *минимальных невязок*.

5. Явный итерационный метод *с чебышевским набором параметров*.

6. Методы *расщепления*.

К группе стационарных двухслойных методов относится *метод простой итерации* (МПИ) с расчетной формулой

$$\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} - \tau(\boldsymbol{A}\boldsymbol{x}^{(k-1)} - \boldsymbol{b}).$$

Этот метод обычно применяют в случае симметричной и положительно определенной матрицы $\boldsymbol{A}$. Параметр $\tau>0$ выбирают из соображений лучшей сходимости, минимизируя норму матрицы переходов $\|\boldsymbol{E} - \tau\boldsymbol{A}\|_2$. Оптимальным является выбор параметра

$$\tau = \frac{2}{\lambda_{\min} + \lambda_{\max}},$$

где $\lambda_{\min}, \lambda_{\max}$ - минимальное и максимальное собственные значения матрицы $\boldsymbol{A}$. В этом случае $\|\boldsymbol{E} - \tau\boldsymbol{A}\|_2$ принимает минимальное значение, равное $\frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\min} + \lambda_{\max}}$.

Пример 2.12. Решение систем линейных уравнений методом простой итерации.

Пример 2.13. Решение систем линейных уравнений с помощью вычислительного блока MathCAD.

Решение примеров в среде MathCAD представлено в файле *R6.mcd* электронного комплекса.

Рассмотренные методы широко применяются при вычислении определителей, обращении матриц, различных преобразований матриц, заменяя сложные прямые методы.

2.2.5. О роли ошибок округления в итерационных методах

Утверждения о сходимости итерационных процессов говорят о том, что решение задачи при определенных условиях может быть найдено этим

процессом сколь угодно точно, причем погрешность каждого приближения может быть эффективно проконтролирована (теорема 2.1). Это справедливо до тех пор, пока к погрешности метода (остаточной погрешности) не добавится вычислительная погрешность (погрешность округлений), неизбежная при любых реальных компьютерных расчетах. При значениях q, приближающихся к единице, роль ошибок округлений в образовании общей погрешности тем сильнее, чем медленнее сходимость итерационного процесса.

Ознакомление с решением систем линейных алгебраических уравнений итерационными методами предполагается при выполнении практической работы № 6 [2, с. 28].

Контрольные вопросы

1. Дайте определение и примеры прямых методов решения СЛАУ.
2. Дайте определение и примеры итерационных методов решения СЛАУ.
3. В чем основное отличие точных и приближенных методов решения систем линейных уравнений?
4. Каким методом лучше всего решать систему уравнений невысокого порядка, например третьего?
5. Сущность решения СЛАУ методом Гаусса .
6. С какой целью применяют модификацию метода Гаусса - схему частичного выбора?
7. Модификации метода Гаусса, обладающие лучшими вычислительными свойствами.
8. Использование $\boldsymbol{LU}$-разложения матрицы для решения СЛАУ.
9. Запишите формулы для нахождения решения после приведения системы к виду $\boldsymbol{LUx} = \boldsymbol{b}$.
10. Для каких систем уравнений применяют метод Холецкого?
11. Использование метода Холецкого для решения СЛАУ.
12. Для каких СЛАУ применяется метод прогонки?
13. Сформулируйте алгоритм метода прогонки.
14. Арифметическая сложность прямых методов решения СЛАУ.
15. Можно ли заранее оценить число итераций для получения решения с заданной погрешностью?

16. Что позволяет определить априорная оценка погрешности решения СЛАУ методом простых итераций?

17. Что позволяет определить апостериорная оценка погрешности решения СЛАУ методом простых итераций?

18. В каких случаях предпочтительны итерационные методы решения систем линейных уравнений?

19. Сущность и расчетная формула метода Якоби.

20. Сущность и расчетная формула метода Зейделя.

21. Расчетная формула стационарного двухслойного метода простой итерации.

22. От чего зависит скорость сходимости метода итераций?

23. При каком условии будет сходиться метод итераций?

24. Как влияет вычислительная ошибка на точность решения системы уравнений методом итераций?

3. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И СИСТЕМ

3.1 Методы решения нелинейных скалярных уравнений

3.1.1. Постановка задачи приближенного решения нелинейного уравнения

Многие задачи исследования различных объектов с помощью математических моделей, применения их для прогноза или расчета приводят к необходимости решения нелинейных уравнений.

Пусть рассматривается уравнение

$$f(x) = 0. \qquad (3.1)$$

Если функция представляет многочлен, то уравнение (3.1) называется *алгебраическим*. Если x находится под знаком трансцендентной функции (показательной, логарифмической, тригонометрической и т.п.), уравнение (3.1) будет *трансцендентным*. *Корнем* уравнения называется значение x^*, при котором $f(x^*) = 0$. Корень x^* считается *простым*, если $f'(x^*) \neq 0$, в противном случае корень называется *кратным*. Целое число m называется кратностью корня x^*, если $f^{(k)}(x^*) = 0$ для $k = 1, 2, 3, \ldots, m\text{-}1$ и $f^{(m)}(x^*) \neq 0$.

Постановка задачи вычисления приближенного значения корня с точностью ε: найти такое значение x, что $\left|x - x^*\right| < \varepsilon$.

Решение задачи разбивается на два этапа: на первом этапе осуществляют ***локализацию*** корней, на втором этапе производят ***итерационное уточнение*** корней. На этапе локализации корней находят достаточно узкие отрезки (или отрезок, если корень единственный), которые содержат один и только один корень уравнения $f(x) = 0$. На втором этапе вычисляют приближенное значение корня с заданной точностью. Часто вместо отрезка локализации достаточно указать начальное приближение к корню.

Локализация корней может производиться *графически* или *аналитически*. В первом случае строят график функции $f(x)$ и выделяют те промежутки оси абсцисс, где график пересекает ось *Ox*. Если построение графика $y = f(x)$ затруднительно, но исходное уравнение (3.1) очевидным образом представляется в виде $f_1(x) = f_2(x)$, то строят графики $y = f_1(x)$, $y = f_2(x)$ и выделяют на оси абсцисс промежутки пересечения этих графиков.

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 $f(x_i)$. Затем строят ряд знаков функции в критических точках, включая в рассмотрение и крайние точки числовой оси $-\infty$ и $+\infty$. Анализируя этот ряд, по числу смен знаков определяют количество корней и интервалы локализации корней: на левой и правой границах такого интервала функция $f(x)$ должна иметь разные знаки. Одна из проблем, в которую упирается решение задачи локализации корней, – это практическая невозможность точного вычисления значения функций.

Пример 3.1. Локализация корней графическим способом. Решение представлено в среде MathCAD в файле *R7_ex1.mcd* электронного комплекса.

Идеи методов этапа **уточнения корней** можно сгруппировать по трем основным направлениям:

1) поиск корня с заданной погрешностью путем перебора всех возможных значений аргумента с проверкой наличия решения;

2) замена нелинейной функции более простой функцией (линейной, параболической), близкой к исходной, и поиск ее корня итерационными процедурами;

3) сведение нелинейного уравнения $f(x)=0$ к $f_1(x)=f_2(x)$ и обеспечение в нем равенства итерационными процедурами.

Условием окончания процесса решения уравнения (т.е. получения корня x^* с заданной погрешностью) может быть одно из двух возможных:

1) $f\left(x^*\right)\le \delta$;

2) $\left|x^*-x^{(k)}\right|\le \varepsilon$,

где δ,ε – 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 на $\left|x^{(k+1)}-x^{(k)}\right|\le \varepsilon$.

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 $f(x)$. Пусть $[a,b]$ – любой отрезок локализации, называемый еще ***промежутком существования корня***. Предположим, что функция $f(x)$ непрерывна на $[a,b]$ и на концах принимает значения разных знаков $f(a)\cdot f(b)<0$.

3.1.2. Метод сканирования

Относится к методам первой группы. Метод предусматривает разделение всего интервала $[a,b]$ на маленькие отрезки, равные заданной погрешности ε, с последующим вычислением функции $f(x)$ на концах этих отрезков. Выбирается отрезок, где функция меняет знак или не превосходит ε. В качестве решения можно взять любую точку этого отрезка. В любом случае погрешность решения не будет превышать заданную.

Для повышения эффективности метода можно уточнение производить в несколько этапов. На первом этапе задать большое значение ε_1,

найти отрезок, где функция меняет знак (грубо найти корень), затем найденный отрезок еще раз разделить с более мелким шагом ε_2, более точно найти корень и т.д. еще несколько этапов (обычно 3…5), после чего удается найти корень с заданной погрешностью в целом за меньшее число раз вычисления $f(x)$ (рис. 3.1).

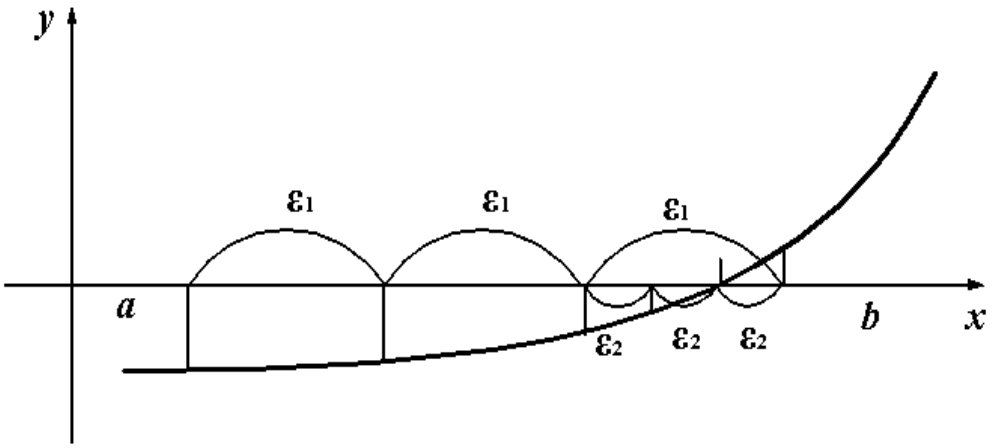


Рис. 3.1. Геометрическая интерпретация метода сканирования

3.1.3. Метод дихотомии (бисекции)

Относится к методам первой группы. Алгоритм метода ***дихотомии*** состоит в построении последовательности вложенных отрезков, на концах которых функция принимает значения разных знаков. Берется произвольная точка $c \in (a,b)$, называемая ***пробной точкой***. Вычисление значения $f(c)$ приведет к какой-либо одной из следующих взаимоисключающих ситуаций:

а) $f(a) \cdot f(c) < 0$ – корень находится на интервале (a,c);

б) $f(c) \cdot f(b) < 0$ – корень находится на интервале (c,b);

в) $f(c) = 0$ – точка c является искомым корнем.

Таким образом, одно вычисление значения функции позволяет уменьшить промежуток $[a,b]$ существования корня (ситуации а), б)) или указать его значение (ситуация в)), маловероятная в смысле «прямого попадания» пробной точки c в корень, но вполне реальная в смысле выполнения приближенного равенства $f(c) \approx 0$, когда длина промежутка существования корня близка к ε. Для ситуаций а) или б) описанная процедура одного шага сужения промежутка существования корня может быть при-

менена к промежутку $[a,c]\subset[a,b]$ или к $[c,b]\subset[a,b]$ соответственно и далее повторяться циклически (рис. 3.2).

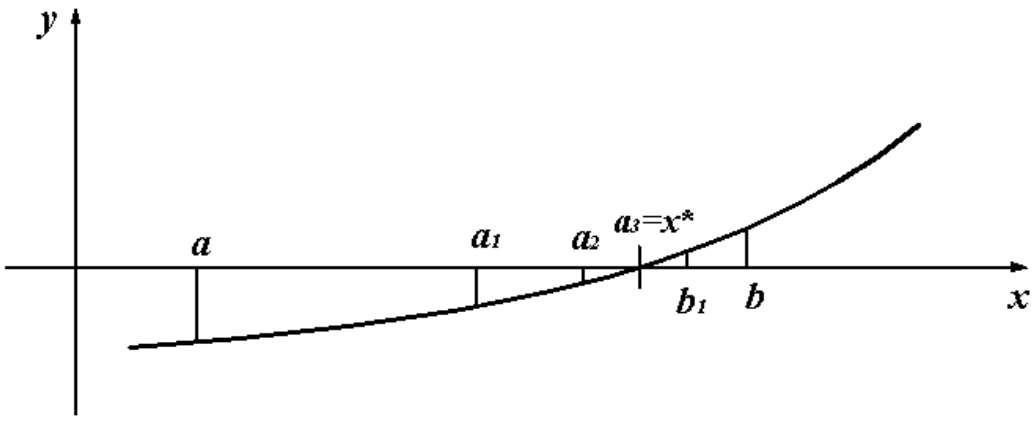


Рис. 3.2. Геометрическая интерпретация метода бисекции

Наиболее употребительным частным случаем метода дихотомии является ***метод половинного деления***, реализующий самый простой способ выбора пробной точки – деление промежутка существования корня пополам. Опишем его алгоритм:

Шаг 0. Задать концы отрезка $[a,b]$, функцию $f(x)$, допустимую абсолютную погрешность корня ε; вычислить $f(a)$.

Шаг 1. Вычислить $c := 0{,}5(a+b)$.

Шаг 2. Если $b-a<2\varepsilon$, положить $x :\approx c$ и остановиться.

Шаг 3. Вычислить $f(c)$.

Шаг 4. Если $f(c)=0$, положить $x :\approx c$ и остановиться.

Шаг 5. Если $f(a)\cdot f(c)<0$, положить $b := c$, $f(b) := f(c)$ и вернуться к шагу 1; иначе положить $a := c$, $f(a) := f(c)$ и вернуться к шагу 1.

За один шаг метода половинного деления промежуток существования корня сокращается ровно вдвое. Поэтому оценка погрешности корня x после выполнения k-й итерации

$$\left|x-x^{(k)}\right|<\frac{b-a}{2^k}\quad\forall\ k\in N$$

дает возможность подсчитать число шагов (итераций) метода, достаточное для получения корня x с заданной точностью ε. Для этого нужно найти наименьшее натуральное k, удовлетворяющее неравенству

$$\frac{b-a}{2^k}<\varepsilon.$$

Пример 3.2. Решение уравнения методом бисекции. Пример представлен в среде MathCAD в файле *R7_ex2.mcd* электронного комплекса.

3.1.4. Метод хорд

Относится к методам второй группы. Используемый в методе половинного деления способ фиксирования пробной точки можно охарактеризовать как пассивный, ибо он осуществляется по заранее жестко заданному плану и никак не учитывает вычисляемые на каждом шаге значения функции. В семействе методов дихотомии можно достичь лучших результатов, если отрезок $[a,b]$ делить точкой c на части не пополам, а пропорционально величинам ординат $f(a)$ и $f(b)$ графика данной функции $f(x)$. Это означает, что точку c есть смысл находить как абсциссу точки пересечения оси Ox с прямой, проходящей через точки $A(a; f(a))$ и $B(b; f(b))$, иначе с хордой AB дуги $A\bar{x}B$.

Запишем уравнение прямой, проходящей через две данные точки *A* и *B:*

$$\frac{y-f(a)}{f(b)-f(a)}=\frac{x-a}{b-a}.$$

Отсюда, полагая $y=0$ (уравнение оси Ox), $x=c$, находим:

$$c=a-\frac{f(a)(b-a)}{f(b)-f(a)}. \qquad (3.2)$$

Существует несколько версий реализации метода хорд. Одна из них – подсчет значений c по формуле (3.2) в рамках алгоритма типа рассмотренного выше алгоритма половинного деления, где следует положить $b:=c$ при $f(a)\cdot f(c)<0$. Счет ведется до совпадения значений c на двух соседних итерациях с точностью ε: $\left|c^{(k)}-c^{k-1}\right|\le\varepsilon$. Хорда может быть с «закрепленным» правым (рис. 3.3) или левым концом – соответственно приближение к корню слева или справа.

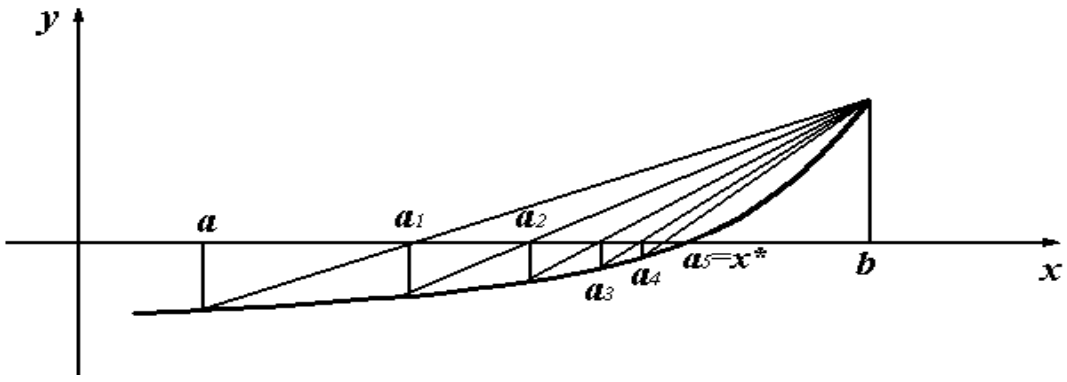


Рис. 3.3. Геометрическая интерпретация метода хорд

Метод быстро сходится для функций, близких к линейным, и может проигрывать в скорости методу половинного деления при экспоненциальном характере функции.

В целом методы семейства дихотомии являются ***глобально сходящимися*** со скоростью геометрической прогрессии (обладают ***линейной сходимостью***, сходимостью первого порядка).

3.1.5. Метод Ньютона (касательных)

Относится к методам второй группы. ***Метод Ньютона*** (в зарубежной литературе – ***метод Ньютона-Рафсона***) является одним из популярнейших итерационных методов решения нелинейных уравнений, что связано с его идейной простотой и быстрой сходимостью.

Аналогичен методу 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 $f(a) \cdot f''(a) > 0$), или $x^{(0)} = b$ (если $f(b) \cdot f''(b) > 0$). Расчетная формула метода Ньютона имеет вид

$$x^{(k+1)} = x^{(k)} - \frac{f\left(x^{(k)}\right)}{f'\left(x^{(k)}\right)}.$$

Геометрически метод Ньютона означает, что следующее приближение к корню $x^{(k+1)}$ есть точка пересечения с осью Ox касательной, проведенной к графику функции $y = f(x)$ в точке $\left(x^{(k)}, f\left(x^{(k)}\right)\right)$ (рис. 3.4).

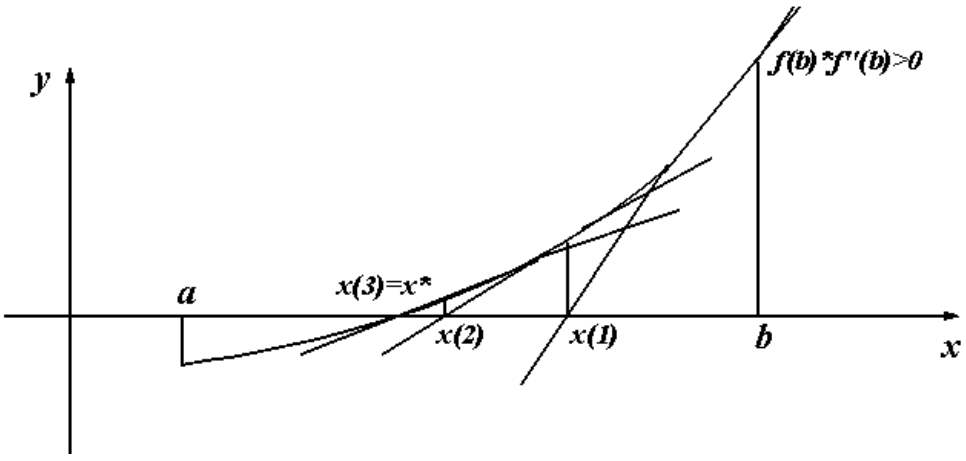


Рис. 3.4. Геометрическая интерпретация метода Ньютона

Теорема о сходимости метода Ньютона. Пусть x^* – простой корень уравнения $f(x)=0$, в некоторой окрестности которого функция дважды непрерывно дифференцируема. Тогда найдется такая малая δ - окрестность корня x^*, что при произвольном выборе начального приближения $x^{(0)}$ из этой окрестности итерационная последовательность метода Ньютона не выходит за пределы окрестности и справедлива оценка

$$\left|x^{(k+1)}-x^*\right| \le \frac{1}{\delta}\left|x^{(k+1)}-x^*\right|^2.$$

Критерий окончания итерационного процесса. При заданной точности $\varepsilon>0$ вычисления следует вести до тех пор, пока не окажется выполненным неравенство $\left|x^{(k)}-x^{(k-1)}\right|<\varepsilon$.

Пример 3.3. Решение уравнения методом Ньютона. Пример представлен в среде MathCAD в файле *R7_ex3.mcd* электронного комплекса.

Метод Ньютона обладает высокой скоростью сходимости – ***второго порядка (квадратичной)***, но сходимость является ***локальной***. Неудачный выбор начального приближения может дать расходящуюся итерационную последовательность. Поэтому методу Ньютона часто предшествует какой-либо надежно сходящийся алгоритм (например метод половинного деления), а метод Ньютона используют на завершающей стадии решения уравнения.

Пример 3.4. Чувствительность метода Ньютона к выбору начального приближения. Пример представлен в среде MathCAD в файле *R7_ex4.mcd* электронного комплекса.

3.1.6. Модификации метода Ньютона

Трудностью метода Ньютона является необходимость вычисления производной на каждом итерационном шаге. Далеко не всегда бывает удобно находить аналитическое выражение для производной функции. Однако в этом и нет особой необходимости: поскольку на каждом шаге мы получаем *приближенное* значение корня, можно для его вычисления использовать *приближенное* значение производной, например, найденное в начальной точке $x^{(0)}$. Такую модификацию метода называют ***упрощенным методом Ньютона***

$$x^{(k+1)} = x^{(k)} - \frac{f\left(x^{(k)}\right)}{f'\left(x^{(0)}\right)}.$$

Можно найти приближенное значение производной из соотношения

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x} \approx \frac{f(x+\Delta x) - f(x)}{\Delta x}.$$

В качестве малой величины Δx можно взять, например, заданную точность вычислений ε, тогда расчетная формула примет вид

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})\varepsilon}{f(x^{(k)} + \varepsilon) - f(x^{(k)})}, \tag{3.3}$$

что составляет основу ***разностного метода Ньютона***.

С другой стороны, для вычисления производной можно воспользоваться значениями функции, полученными на двух предыдущих шагах,

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})(x^{(k-1)} - x^{(k)})}{f(x^{(k-1)}) - f(x^{(k)})}, \tag{3.4}$$

где $k = 1,2,3,...$, а $x^{(0)}$ и $x^{(1)}$ должны задаваться. В таком виде метод называется ***методом секущих (secant method)***. При этом, однако, возникает проблема с вычислением первого приближения. Обычно полагают, что $x^{(1)} = x^{(0)} + \varepsilon$, то есть первый шаг вычислений проводится с использованием формулы (3.3), а все последующие – с использованием формулы (3.4).

Формула (3.4) определяет новый метод как ***двухшаговый***. Его геометрическая интерпретация: x_{k+1} есть абсцисса точки пересечения с осью Ox прямой, проведенной через точки $(x_{k-1}; f(x_{k-1}))$ и $(x_k; f(x_k))$, т.е. секущей (рис. 3.5).

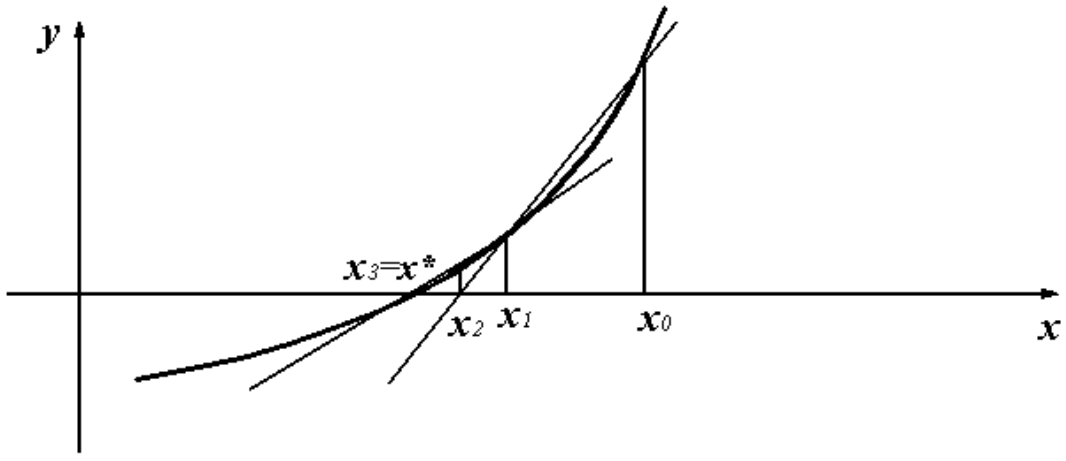


Рис. 3.5. Геометрическая интерпретация метода секущих

Используя метод секущих, мы не можем гарантировать, что корень находится между двумя последними приближениями. Метод секущих и

метод хорд определяются однотипными формулами, но порождающие их идеологии различны, что сказывается на свойствах и скорости сходимости генерируемой ими последовательности приближений. Метод секущих обладает квадратичной сходимостью.

Высокий порядок скорости сходимости метода секущих в сочетании с минимальными вычислительными затратами – одно вычисление функции на один итерационный шаг – выводит этот метод на первое место по эффективности решения скалярных нелинейных уравнений среди прочих итерационных методов. Именно эта вычислительная схема реализована в пакете ***Mathcad***.

Идея метода секущих развивается в ***методах Мюллера***, ***Риддера (Ridder's method)***, ***Брента (Brent method)***. Формулы методов достаточно громоздки и в данном курсе не приводятся.

3.1.7. Комбинированный метод

Относится к методам второй группы. Так же как и предыдущие, данный метод 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 обусловливает другое окончание поиска. Поиск можно прекратить, как только разница между правым и левым концами интервала станет меньше предварительно заданной погрешности ε.

3.1.8. Метод параболической аппроксимации

Относится к методам второй группы. 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 отрезка локализации корня $[a,b]$. По полученному уравнению параболы $y = c_2x^2 + c_1x + c_0$ находят приближенный корень, 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 сторо-

ны оси x. Процедура повторяется до тех пор, пока величина отрезка локализации корня не будет меньше ε – предварительно заданной погрешности.

3.1.9. Метод простой итерации (последовательных приближений)

Относится к методам третьей группы. Для применения метода простой итерации следует исходное уравнение $f(x)=0$ *преобразовать к виду, удобному для итерации* $x=\varphi(x)$. Это преобразование можно выполнить различными способами. Функция $\varphi(x)$ называется итерационной функцией. Расчетная формула метода простой итерации имеет вид

$$x^{(k+1)}=\varphi\left(x^{(k)}\right).$$

Теорема о сходимости метода простой итерации. Пусть в некоторой δ- окрестности корня x^* функция $\varphi(x)$ дифференцируема и удовлетворяет неравенству $|\varphi'(x)|\le q$, где $0\le q<1$ – постоянная. Тогда независимо от выбора начального приближения из указанной δ-окрестности итерационная последовательность не выходит из этой окрестности, метод сходится со скоростью геометрической последовательности и справедлива оценка погрешности

$$\left|x^{(k)}-x^*\right|\le\frac{q}{1-q}\left|x^{(k)}-x^{(k-1)}\right|.$$

Геометрическая интерпретация метода простой итерации показана на рис. 3.6.

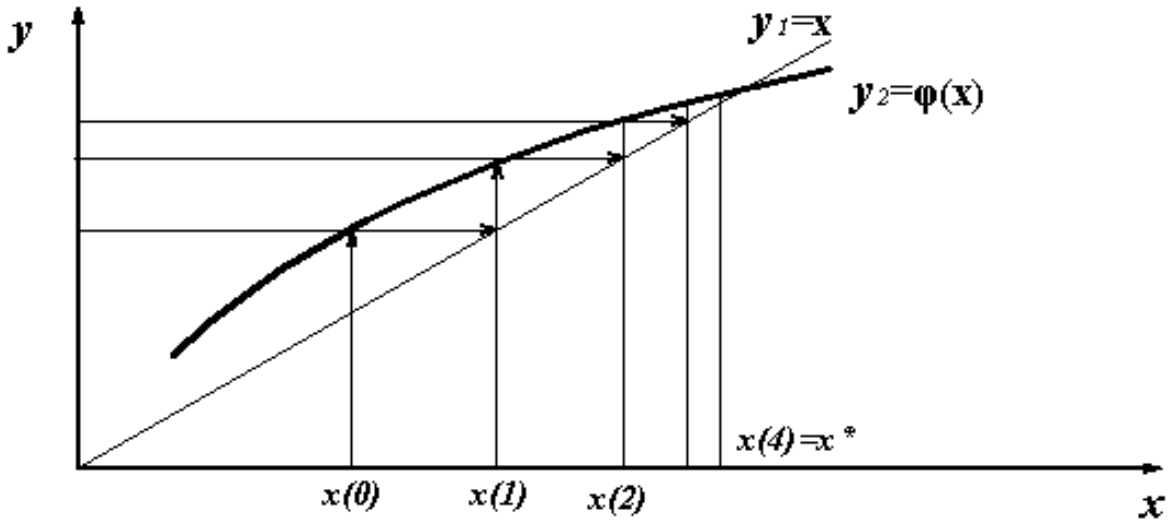


Рис. 3.6. Геометрическая интерпретация метода простой итерации

Критерий окончания итерационного процесса. При заданной точности $\varepsilon > 0$ вычисления следует вести до тех пор, пока не окажется выполненным неравенство $\left|x^{(k)} - x^{(k-1)}\right| < \frac{1-q}{q}\varepsilon$. Если величина $0 < q \le 0{,}5$, то можно использовать более простой критерий окончания итераций $\left|x^{(k)} - x^{(k-1)}\right| < \varepsilon$.

Ключевой момент в применении метода простой итерации состоит в эквивалентном преобразовании уравнения. Способ, при котором выполнено условие сходимости метода простой итерации, состоит в следующем: исходное уравнение приводится к виду $x = x - \alpha f(x)$, где α – итерационный параметр. Предположим дополнительно, что производная f' знакопостоянна и $m \le f'(x) \le M$ на отрезке $[a,b]$. Тогда при выборе итерационного параметра $\alpha = \frac{2}{m+M}$ метод сходится и $q = \left|\frac{M-m}{M+m}\right| < 1$.

Метод последовательных приближений обладает ***локальной сходимостью*** со скоростью ***первого порядка (линейной)***, то есть областью его сходимости является малая окрестность корня x^*. Достоинство метода заключается в том, что не накапливаются ошибки вычислений. Ошибка вычислений эквивалентна ухудшению очередного приближения, что может отразиться только на числе итераций, но не на точности результата. Метод устойчив даже по отношению к грубым ошибкам.

Пример 3.5. Приведение уравнения к виду, удобному для итераций. Решение примера в среде MathCAD представлено в файле *R7_ex5.mcd* электронного комплекса.

3.2. Решение алгебраических уравнений

Рассмотрим решение алгебраического уравнения

$$P_n(x) = 0, \tag{3.5}$$

т.е. нелинейного уравнения, где функция $P_n(x)$ имеет вид многочлена (полинома)

$$P_n(x) \equiv a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0. \tag{3.6}$$

Всего уравнение (3.5) порядка n имеет n корней (действительных и комплексных). При действительных коэффициентах уравнения комплексные корни будут попарно сопряженными. Правило Декарта оценивает количество положительных и отрицательных действительных корней: коли-

чество положительных корней равно числу перемен знака в последовательности коэффициентов уравнения либо на четное число меньше; количество отрицательных корней равно числу перемен знака в последовательности коэффициентов уравнения $P_n(-x)=0$ либо на четное число меньше (равные нулю коэффициенты в обоих случаях не учитываются).

Пример 3.6. $(x-3)(x-2)(x-1)=0$.

$P_3(x)=0$: $x^3-6x^2+11x-6=0$

\+ 1) – 2) + 3) – : три положительных корня.

$P_3(-x)=0$: $-x^3-6x^2-11x-6=0$.

Нет смены знака: нет отрицательных корней.

Если требуется найти только действительные корни и не все из существующих, то целесообразно использовать для решения алгебраического уравнения (3.5) какие-либо из рассмотренных выше методов решения нелинейных скалярных уравнений. Многократное вычисление значения многочлена (3.5), необходимое при этом, можно ускорить почти вдвое использованием ***схемы Горнера***.

Во многих случаях достаточно знать предельное (максимальное) значение корня (например при оценке устойчивости систем управления):

1) ***Метод Лагранжа*** позволяет определить верхнюю границу положительных корней по формуле

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

где m – номер первого по порядку отрицательного коэффициента в полиноме $P_n(x)$, B – наибольшая из абсолютных величин отрицательных коэффициентов $P_n(x)$ в предположении, что $a_n>0$.

Верхняя граница положительных корней для примера 3.6: $R=1+\sqrt[2]{6/1}\approx 3{,}45$.

2) ***Метод Ньютона*** заключается в переборе наугад взятых точек: если при каком-то значении $x=\beta$ $P_n(x)>0$ и все производные $P_n'(x)>0,\ P_n''(x)>0,\ldots,\ P_n^{(n)}(x)>0$, то значение β является предельным значением положительных корней.

3) ***Метод кольца*** позволяет находить область существования всех корней алгебраического уравнения, в том числе и комплексных. Действительные корни находятся в интервалах ($–R$, $–r$) и (r, R) (соответственно отрицательные и положительные). Величины r и R вычисляются по формулам

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

где $A = \max\{|a_{n-1}|,\ |a_{n-2}|,...,|a_0|\}; \qquad B = \max\{|a_n|,\ |a_{n-1}|,...,|a_1|\}.$

4) ***Метод предельных значений*** находит область существования корней несколько точнее. 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 $(1/R_3, R_1)$ [4].

Методы уточнения корней алгебраического уравнения:

1) ***метод понижения порядка*** (***метод Лина***) – 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. [13].

2) На использовании квадратичного множителя $x^2 + bx + c$ основан и ***метод Берстоу*** [13].

3) Одним из наиболее эффективных методов нахождения всех или почти всех корней алгебраического уравнения как действительных, так и комплексных, является ***метод Лобачевского***. Основная идея метода заключается в последовательном применении операции квадрирования корней [1].

4) ***Метод Лагерра (Laguerre's method)*** основывается на следующих соотношениях для полиномов:

$P_n(x) = (x - x_1)(x - x_2)\ldots(x - x_n)$,

$$\ln\left|P_{n(x)}\right| = \ln|x - x_1| + \ln|x - x_2| \ldots \ln|x - x_n|,$$

$$\frac{d \ln|P_n(x)|}{dx} = \frac{1}{x - x_1} + \frac{1}{x - x_2} \ldots \frac{1}{x - x_n} = \frac{P_n'}{P_n} \equiv G,$$

$$-\frac{d^2 \ln|P_n(x)|}{dx^2} = \frac{1}{(x - x_1)^2} + \frac{1}{(x - x_2)^2} \ldots \frac{1}{(x - x_n)^2} = \left(\frac{P_n'}{P_n}\right)^2 - \frac{P_n''}{P_n} \equiv H.$$

Предполагают, что корень x_1 находится на расстоянии a от текущего приближения, в то время как все другие корни находятся на расстоянии b: $x - x_1 = a; x - x_i = b,$ если $i = 2,3,\ldots,n.$ Тогда

$$\frac{1}{a} + \frac{n-1}{b} = G, \quad \frac{1}{a^2} + \frac{n-1}{b^2} = H,$$

откуда

$$a = \frac{n}{G \pm \sqrt{(n-1)(nH - G^2)}}.$$

Знак перед корнем выбирают с таким расчетом, чтобы получить наибольшее значение знаменателя.

5) Еще один метод, который применяют для поиска корней полиномов, – ***метод сопровождающей матрицы*** (***companion matrix***). Можно доказать, что матрица

$$A = \begin{pmatrix} -\frac{a_{n-1}}{a_n} & -\frac{a_{n-2}}{a_n} & \ldots & -\frac{a_1}{a_n} & -\frac{a_0}{a_n} \\ 1 & 0 & \ldots & 0 & 0 \\ 0 & 1 & \ldots & 0 & 0 \\ \vdots & & \ldots & & \vdots \\ 0 & 0 & \ldots & 1 & 0 \end{pmatrix},$$

называемая сопровождающей матрицей для полинома $P(x) = \sum_{i=0}^{n} a_i x^i$, имеет собственные значения, равные корням полинома. Напомним, что собственными значениями матрицы называются такие числа, для которых выполняется равенство $\boldsymbol{Ax} = \lambda \boldsymbol{x}$ или $P(x) = \det(\boldsymbol{A} - \lambda \boldsymbol{I}) = 0$. Существуют весьма эффективные методы поиска собственных значений. Таким образом, задачу поиска корней полинома можно свести к задаче поиска собственных значений сопровождающей матрицы.

3.3. Методы решения систем нелинейных уравнений

Формально задача поиска решения системы нелинейных уравнений

$$f_1(x_1, x_2, \ldots, x_n) = 0,$$
$$f_2(x_1, x_2, \ldots, x_n) = 0,$$
$$\ldots$$
$$f_3(x_1, x_2, \ldots, x_n) = 0$$

может быть записана точно так же, как и задача поиска корня одного уравнения $\boldsymbol{f}(\boldsymbol{x}) = 0$, где $\boldsymbol{f} = (f_1, f_2, \ldots, f_n)$ и $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ – векторная функция и вектор неизвестных соответственно.

В основном для решения систем нелинейных уравнений используют два метода: метод Ньютона–Рафсона и метод простых итераций. Реже используются поисковые методы оптимизации, которыми ищут минимум суммы квадратов невязок $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{\varepsilon}$ всех уравнений системы, подбирая переменные x_i:

$$R = \sum_i \varepsilon_i \to \min_{x_i}.$$

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

В ***методе простых итераций*** (***МПИ***), как и при решении одного уравнения, предварительно все уравнения приводятся к специальному виду $x_i = \varphi_i(x_1, x_2, \ldots x_n)$ или в векторном виде $\boldsymbol{x} = \boldsymbol{\Phi}(\boldsymbol{x})$. Алгоритм метода аналогичен случаю решения одного уравнения. Метод сходится, если $\sum_{j=1}^{n} \left| \frac{\partial \varphi_i}{x_j} \right| < 1, \ i = \overline{1,n}$ или $\sum_{i=1}^{n} \left| \frac{\partial \varphi_i}{x_j} \right| < 1, \ j = \overline{1,n}$. Сходимость метода очень чувствительна к степени близости начального приближения к истинным значениям корней, и тем чувствительней, чем выше порядок системы. Кроме того, преобразование системы может потребовать значительных вычислений.

Метод Ньютона обеспечивает лучшую сходимость и является более популярным. Вблизи точки x каждая из функций f_i может быть разложена в ряд Тейлора $f_i(\boldsymbol{x} + \Delta\boldsymbol{x}) = f_i(\boldsymbol{x}) + \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j} \Delta x_j + \ldots$ или в векторной форме $\boldsymbol{f}(\boldsymbol{x} + \Delta\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{J}\Delta\boldsymbol{x} + \ldots$, где $\boldsymbol{J}$ – матрица Якоби с элементами

$$J_{i,j} = \frac{\partial f_i}{\partial x_j}.$$

Ограничиваясь только первыми двумя членами разложения и полагая, что $\boldsymbol{f}(\boldsymbol{x}+\Delta\boldsymbol{x})=0$ (т.е. приращение $\Delta\boldsymbol{x}$ ведет от текущего приближения к истинным корням), получаем уравнение $\Delta\boldsymbol{x}=-\boldsymbol{J}^{-1}\boldsymbol{f}(\boldsymbol{x})$. Подставляя $\Delta\boldsymbol{x}=\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^{(k)}$, мы получаем схему для уточнения решения системы уравнений, аналогичную методу Ньютона для случая одного уравнения

$$\boldsymbol{x}^{(k+1)}=\boldsymbol{x}^{(k)}-\boldsymbol{J}^{-1}\boldsymbol{f}(\boldsymbol{x}^{(k)}).$$

Условие окончания итерационного процесса $\left\|\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^{(k)}\right\|\le\varepsilon$.

Поскольку вычислять матрицу Якоби на каждом шаге достаточно трудоемко, то обычно ее элементы вычисляют приближенно или используют одни и те же значения на нескольких шагах. Одну из разновидностей метода Ньютона – ***метод Левенберга-Маркардта*** – использует Mathcad.

Решение нелинейных уравнений и систем рассматривается в практической работе № 7 [2, с. 36].

Контрольные вопросы

1. Сформулируйте постановку задачи приближенного решения нелинейного уравнения.
2. Этапы решения нелинейных уравнений.
3. Какие методы решения нелинейных уравнений вы можете перечислить?
4. Сущность и геометрическая интерпретация метода сканирования решения нелинейных уравнений.
5. В чем сущность и геометрическая интерпретация метода бисекции решения нелинейных уравнений?
6. Можно ли найти кратный корень с помощью метода бисекции?
7. В чем сущность и геометрическая интерпретация метода хорд решения нелинейных уравнений?
8. Каковы сущность и геометрическая интерпретация метода Ньютона решения нелинейных уравнений?
9. Модификации метода Ньютона решения нелинейных уравнений.
10. Сущность и геометрическая интерпретация комбинированного метода решения нелинейных уравнений.
11. В чем сущность и геометрическая интерпретация метода параболической аппроксимации решения нелинейных уравнений?
12. Сущность и геометрическая интерпретация метода простой итерации решения нелинейных уравнений.

13. Критерий окончания итераций для метода простой итерации.

14. Особенности решения алгебраических уравнений.

15. Оценки количества и «качества» корней при решении алгебраических уравнений.

16. Перечислите методы уточнения корней алгебраических уравнений.

4. ПРИБЛИЖЕНИЕ ФУНКЦИЙ. ИНТЕРПОЛЯЦИЯ

4.1. Виды приближений функции. Постановка задачи интерполяции

Приближением (***аппроксимацией в широком смысле***) функции $f(x)$ называется нахождение такой функции $g(x)$ (***аппроксимирующей функции***), которая была бы близка исходной (рис. 4. 1). При этом $f(x)$ может быть известной, неизвестной или частично известной. Функция $g(x)$ должна обладать «хорошими» свойствами, позволяющими легко производить над нею те или иные аналитические или вычислительные операции. ***Критерии близости*** (***согласия***) функций $f(x)$ и $g(x)$ могут быть различные.

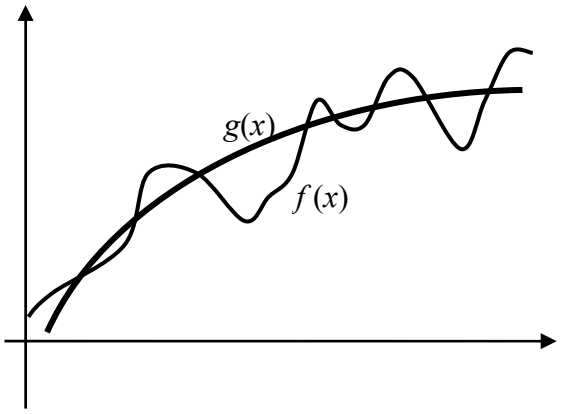


Рис. 4.1. Геометрическая интерпретация аппроксимации

В том случае, когда аппроксимация проводится на непрерывном множестве точек (отрезке), аппроксимация называется ***непрерывной***, или ***интегральной***. Примером такой аппроксимации может служить разложение функции в ряд Тейлора, то есть замена некоторой функции степенным многочленом.

При разработке математических моделей объектов и процессов обрабатываемые экспериментальные данные чаще всего представляются в виде массивов независимых и зависимых переменных – x_i, y_i, т.е. приближение строится на дискретном наборе точек. Такую аппроксимацию называют ***точечной***, или ***дискретной*** (рис 4.2). Аппроксимирующая функция $g(x)$ в зависимости от специфики задачи может отвечать различным требованиям:

1) $g(x)$ должна проходить через точки (x_i, y_i), т.е. $g(x_i) = y_i$, $i = 1...n$ (функция $g_1(x)$ на рис. 4.2). В этом случае говорят об ***интерполяции*** данных функцией $g(x)$ – нахождении значений таблично заданной функции в тех точках внутри данного интервала, где она не задана; или ***экстраполяции*** – восстановлению функции в точках за пределами заданного интервала. В обоих случаях исходные 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);

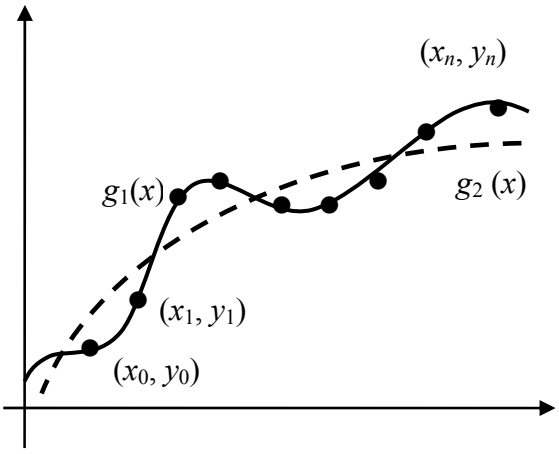


Рис. 4.2. Дискретное приближение

2) $g(x)$ должна некоторым образом (например в виде определенной аналитической зависимости) приближать $f(x)$, не обязательно проходя через точки (x_i, y_i) (функция $g_2(x)$ на рис. 4.2). Такова постановка задачи ***регрессии*** (***сглаживания*** данных), которую часто называют ***аппроксимацией в узком смысле***;

3) $g(x)$ должна приближать экспериментальную зависимость $y(x_i)$, учитывая к тому же, что данные (x_i, y_i) получены с некоторой погрешностью, выражающей шумовую компоненту измерений. При этом функция $g(x)$ с помощью того или иного алгоритма уменьшает погрешность, присутствующую в данных (x_i, y_i). Такого типа задачи называют задачами ***фильтрации***. Сглаживание – частный случай фильтрации.

Различные виды построения аппроксимирующей функции $g(x)$ показаны на рис. 4.3.

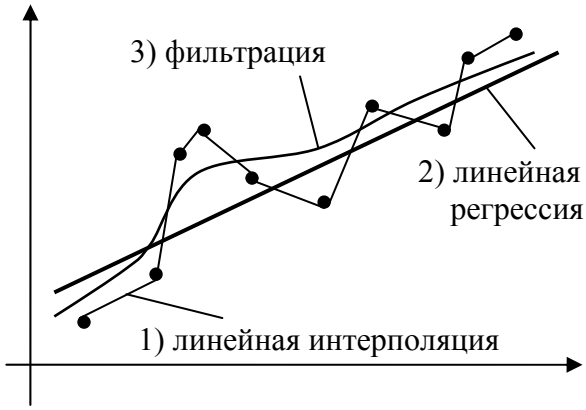


Рис. 4.3. Виды аппроксимации

Как в целях подавления шума, так и для решения других про-

блем обработки данных широко применяются различные интегральные преобразования. Они ставят в соответствие всей совокупности данных $y(x_i)$ некоторую функцию другой координаты (или координат) $F(\omega)$. Примерами интегральных преобразований являются преобразование Фурье и вейвлет-преобразование. В MathCAD некоторые преобразования, например Фурье и Лапласа, можно осуществить в режиме символьных вычислений.

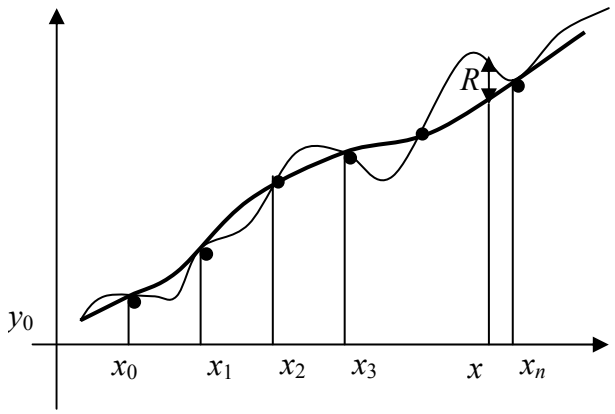


Рис. 4.4. Интерполяция

Рассмотрим подходы к решению задачи интерполяции. Пусть задан дискретный набор точек $x_i (i = 0,1,\ldots,n)$, называемых ***узлами интерполяции***, причем среди этих точек нет совпадающих, а также значения функции y_i в этих точках. Требуется построить функцию $g(x)$, проходящую через все заданные узлы. Таким образом, критерием близости функции является $g(x_i) = y_i$ (рис. 4.4).

Имеется три проблемы интерполяции:

1) выбор интерполяционной функции $g(x)$;

2) оценка погрешности $R(x)$;

3) размещение узлов интерполяции для обеспечения наивысшей возможной точности восстановления исходной функции.

В качестве функции $g(x)$ обычно выбирается полином, который называют ***интерполяционным полиномом***, а интерполяцию – ***полиномиальной***.

В том случае, когда полином един для всей области интерполяции, говорят, что интерполяция ***глобальная.***

В тех случаях, когда между различными узлами полиномы различны, говорят о ***кусочной***, или ***локальной интерполяции.***

Найдя интерполяционный полином, мы можем вычислить значения функции $f(x)$ между узлами (провести ***интерполяцию в узком смысле слова***), а также определить значение функции $f(x)$ даже за пределами заданного интервала (провести ***экстраполяцию***). Следует иметь в виду, что точность экстраполяции обычно очень невелика.

4.2. Интерполяционный многочлен Лагранжа

Пусть функция $y = f(x)$ задана таблицей своих $n+1$ значений (i=0, 1, ..., n):

i	0	1	……..	n
x_i	x_0	x_1	……..	x_n
y_i	y_0	y_1	……..	y_n

Требуется найти многочлен степени n такой, что значения функции и многочлена в точках таблицы совпадают. Справедлива теорема о существовании и единственности интерполяционного многочлена

$$P_n(x) = a_0 + a_1 x + a_2 x^2 + ... + a_{n-1} x^{n-1} + a_n x^n .$$

Его можно построить различными методами с различными вычислительными затратами. Все методы построения интерполяционной функции в виде полинома дают один и тот же результат.

Для нахождения коэффициентов интерполяционного многочлена $a_0, a_1, ... a_n$ как раз имеется $n+1$ условие

$$\begin{cases} a_0 + a_1 x_0 + a_2 x_0^2 + ... + a_n x_0^n = y_0, \\ a_0 + a_1 x_1 + a_2 x_1^2 + ... + a_n x_1^n = y_1, \\ ..... \\ a_0 + a_1 x_n + a_2 x_n^2 + ... + a_n x_n^n = y_n. \end{cases}$$

Решить получившуюся систему линейных алгебраических уравнений можно одним из рассмотренных ранее методов, однако это малоэффективно.

Одна из форм записи интерполяционного многочлена для нахождения функции в какой-либо точке x из отрезка $[x_0, x_n]$ – ***многочлен Лагранжа***:

$$L_n(x) = \sum_{i=0}^{n} y_i l_i(x),$$

где $l_i(x) = \dfrac{(x-x_0)(x-x_1)...(x-x_{i-1})(x-x_{i+1})...(x-x_n)}{(x_i-x_0)(x_i-x_1)...(x_i-x_{i-1})(x_i-x_{i+1})...(x_i-x_n)}$ – базисные многочлены Лагранжа.

Если раскрыть все скобки в числителе, привести подобные члены, вычислить значения знаменателей (в знаменателе – числа), то получим полином n-го порядка от x.

При $n = 1$ по двухточечной таблице получаем формулу ***линейной интерполяции***

$$L_1(x) = \frac{(x - x_1)}{(x_0 - x_1)} y_0 + \frac{(x - x_0)}{(x_1 - x_0)} y_1,$$

а при $n = 2$ по трехточечной таблице получаем формулу ***квадратичной интерполяции***

$$L_2(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} y_0 + \frac{(x - x_1)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} y_1 + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} y_2.$$

Оценить погрешность интерполяции в точке x из $[x_0, x_n]$ можно величиной ***остаточного члена***

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

где $M_{n+1} = \max \left| f^{(n+1)}(x) \right|$ – максимальное значение $(n+1)$-й производной исходной функции $f(x)$ на отрезке $[x_0, x_n]$.

Пример 4.1. Построение многочлена Лагранжа [2, с. 62]. В среде MathCAD пример представлен в файле *R8_ex1.mcd* электронного комплекса.

Непосредственное использование многочлена Лагранжа, во-первых, неудобно из-за его громоздкости и соответственно больших вычислительных затрат. Во-вторых, заранее неизвестна степень полинома, необходимая для интерполирования функции с требуемой точностью. В-третьих, при больших значениях n возрастает влияние на результат ошибок в исходных данных, как правило, полученных с приближением. Избавиться от этих недостатков позволяет итерационная схема Эйткена [1].

4.3. Интерполяция методом Ньютона

Другая форма записи интерполяционного многочлена - ***интерполяционный многочлен Ньютона***. В интерполяционном многочлене Лагранжа все слагаемые однотипны и играют одинаковую роль в образовании результата. Построив интерполяционный многочлен подобным многочлену Тейлора образом, т.е. расположив слагаемые в порядке убывания их зна-

чимости, можно было бы более просто перестраивать его степень, добавляя или отбрасывая удаленные от начала его записи члены.

Рассмотрим сначала случай ***равноотстоящих*** (равномерно расположенных) узлов интерполяции, т.е. таких, что

$$x_i = x_0 + ih,$$

где $i = 0, 1, \ldots, n$, а $h > 0$ – постоянная величина, называемая ***шагом сетки*** (таблицы).

Метод Ньютона использует понятие ***конечных разностей***. Рассмотрим ***элементы теории конечных разностей***.

Вычитая из каждого последующего члена последовательности из n+1 чисел $y_0, y_1, ..., y_n$ предыдущий, образуем n ***конечных разностей первого порядка*** (***первых разностей***)

$$\Delta y_0 := y_1 - y_0, \quad \Delta y_1 := y_2 - y_1, \ \ldots, \quad \Delta y_{n-1} := y_n - y_{n-1}.$$

Из них таким же образом можно получить n-1 конечных разностей второго порядка (вторых разностей)

$$\Delta^2 y_0 := \Delta y_1 - \Delta y_0, \quad \Delta^2 y_1 := \Delta y_2 - \Delta y_1, \ \ldots, \quad \Delta^2 y_{n-2} := \Delta y_{n-1} - \Delta y_{n-2}.$$

Рекуррентное выражение конечной разности k-го порядка через конечные разности более низкого порядка

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

Привлекая определение производной, можно обнаружить прямую связь между конечными разностями и производными:

$$f^{(k)}(x_i) \approx \frac{\Delta^k y_i}{h^k}.$$

Используя конечные разности и разложение функции в ряд Тейлора

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k,$$

интерполяционный многочлен Ньютона можно записать в виде, называемом ***первым интерполяционным многочленом Ньютона***:

$$P_n(x) = y_0 + \frac{\Delta y_0}{h}(x - x_0) + \frac{\Delta^2 y_0}{2! h^2}(x - x_0)(x - x_1) + \ldots$$

$$\ldots + \frac{\Delta^n y_0}{n! h^n}(x - x_0)(x - x_1)\ldots(x - x_{n-1}).$$

Нетрудно заметить, что, как и в методе Лагранжа, это выражение есть не что иное, как обычный полином n-го порядка от x, только записан-

ный в другой форме. В этом можно убедиться, раскрыв все скобки и приведя подобные члены. Конечные разности в выражении – это числовые коэффициенты, найденные по заданным точкам.

Часто вводят безразмерную переменную q, показывающую, сколько содержится шагов от ***базового узла*** x_0 до заданной точки x: $q = (x - x_0)/h$. Так как

$$x - x_i = x_0 + qh - x_0 - ih = h(q - i),$$

то приходим к ***первой интерполяционной формуле Ньютона***

$$P_n(x) = y_0 + q\Delta y_0 + \frac{\Delta^2 y_0}{2!} q(q-1) + \ldots + \frac{\Delta^n y_0}{n!} q(q-1)\ldots(q-n+1).$$

В таком виде многочлен наиболее приспособлен для интерполяции в окрестности узла x_0, т.е. при $|q| < 1$, а именно, для *интерполирования вперед* (в сторону увеличения x) и *экстраполирования назад* (левее x_0).

Пример 4.2. Интерполяция многочленом Ньютона для равноотстоящих узлов [2, с.62]. В среде MathCAD пример представлен в файле *R8_ex2.mcd* электронного комплекса.

Так как реально степени интерполяционных многочленов бывают не так велики, в то время как таблицы значений функций достаточно обширны, то за базовый узел можно принимать узел, ближайший к заданной фиксированной точке x, если за ним имеется достаточное число узлов. При расположении x в конце таблицы, т.е. в окрестности узла x_n, для интерполяции назад (левее x_n) и экстраполяции вперед (правее x_n) применяют ***второй интерполяционный многочлен Ньютона***

$$P_n(x) = y_n + \frac{\Delta y_{n-1}}{h}(x - x_n) + \frac{\Delta^2 y_{n-2}}{2!h^2}(x - x_n)(x - x_{n-1}) + \ldots$$

$$\ldots + \frac{\Delta^n y_0}{n!h^n}(x - x_n)(x - x_{n-1})\ldots(x - x_1).$$

Введя переменную $q = (x - x_n)/h$, получим ***вторую интерполяционную формулу Ньютона***:

$$P_n(x) = y_n + q\Delta y_{n-1} + \frac{\Delta^2 y_{n-2}}{2!} q(q+1) + \ldots + \frac{\Delta^n y_0}{n!} q(q+1)\ldots(q+n-1).$$

Для работы в центральной части таблицы применяются ***центральные интерполяционные формулы*** с ***центральными разностями – формулы Гаусса, Стирлинга, Бесселя*** [1].

Погрешность интерполяции (величина остаточного члена) оценивается по формуле, аналогичной методу Лагранжа, или

$$R(x) = \left| f(x) - P_n(x) \right| \le \frac{\max\left\{ \left| \Delta^{(n+1)} y_i \right| \right\}}{(n+1)!} q(q-1)(q-2)...(q-n).$$

В случае ***неравноотстоящих узлов*** для построения интерполяционных формул используют ***разделенные разности*** (разностные отношения). Через значения функции $f(x_0),\ f(x_1), \ldots, f(x_n)$ сначала определяют ***разделенные разности первого порядка***:

$$f(x_0; x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0},$$

$$f(x_1; x_2) = \frac{f(x_2) - f(x_1)}{x_2 - x_1}, \ldots$$

$$f(x_{n-1}; x_n) = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}.$$

На этих разностях базируются ***разделенные разности второго порядка***

$$f(x_0; x_1; x_2) = \frac{f(x_1; x_2) - f(x_0; x_1)}{x_2 - x_0}, \ldots$$

$$f(x_{n-2}; x_{n-1}; x_n) = \frac{f(x_{n-1}; x_n) - f(x_{n-2}; x_{n-1})}{x_n - x_{n-2}},$$

и т.д. Таким образом, если определены k-е разностные отношения $f(x_i; x_{i+1}; \ldots; x_{i+k})$, то ($k$+1)-е определяются через них равенством

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

Интерполяционная формула Ньютона для неравноотстоящих узлов имеет вид

$$P_n(x) = f(x_0) + f(x_0; x_1)(x - x_0) + f(x_0; x_1; x_2)(x - x_0)(x - x_1) + \ldots$$
$$\ldots + f(x_0; x_1; \ldots; x_n)(x - x_0)(x - x_1)\ldots(x - x_{n-1}).$$

Пример 4.3. Построение интерполяционного многочлена Ньютона с конечными разностями по формуле неравноотстоящих узлов [2, с. 63].

Если функция n+1 раз на отрезке $[x_0, x_n]$, содержащем узлы интерполяции x_i, i=0,1,...n, дифференцируема, то для погрешности интерполяции (величины остаточного члена) справедлива оценка:

$$R(x) = |f(x) - P_n(x)| \le \frac{M_{n+1}}{4(n+1)!} h_{\max}^{n+1},$$

где $M_{n+1} = \max_{[x_0, x_n]} \left| f^{(n+1)}(x) \right|$ – максимальное значение $(n+1)$-й производной исходной функции $f(x)$ на отрезке $[x_0, x_n]$; $h_{\max} = \max_{1 \le i \le n} h_i$; $h_i = x_i - x_{i-1}$.

Эта оценка показывает, что для достаточно гладкой функции при фиксированной степени интерполяционного многочлена погрешность интерполяции стремится к нулю не медленнее, чем величина, пропорциональная $h_{\max}^{n+1}$. Этот факт формулируют так: интерполяционный многочлен степени n аппроксимирует функцию с $(n+1)$ порядком точности относительно $h_{\max}^{n+1}$.

В практическом плане формула Ньютона обладает преимуществами перед формулой Лагранжа. Предположим, что необходимо увеличить степень многочлена на единицу, добавив в таблицу еще один узел x_{n+1}. При использовании формулы Лагранжа это приводит к необходимости вычислять каждое слагаемое заново. Для вычисления $P_{n+1}(x)$ достаточно добавить к $P_n(x)$ лишь очередное слагаемое $f(x_0; x_1; ...; x_n; x_{n+1})(x - x_0)(x - x_1)...(x - x_{n-1})(x - x_n)$. Если функция f достаточно гладкая, то справедливо приближенное равенство $f(x) - P_n(x) \approx P_{n+1}(x) - P_n(x)$. Это равенство можно использовать для практической оценки погрешности интерполяции $\varepsilon_n = |P_{n+1}(x) - P_n(x)|$. Вообще же, точность результата интерполирования принципиально не может быть выше, чем точность исходных данных.

Учитывая то, что многочлен k-й степени имеет k-е разности постоянными, а все последующие – нулевыми, приходим к заключению, что *если k-е разности таблицы конечных разностей некоторой функции практически постоянны, то эта функция ведет себя в рассматриваемой области как многочлен k-й степени*; эту степень и следует применять для интерполирования с наибольшей для данных реалий точностью. Дальнейшее увеличение степени полинома может только ухудшить ситуацию из-за нарастания ошибок округления.

Пример 4.4. Использование остаточного члена интерполяции [2, с. 63].

4.4. Метод Чебышева

Этот метод иллюстрирует решение вопроса о выборе узлов интерполяции для получения при заданном числе узлов минимально возможной погрешности. Предварительно переменная x преобразуется в переменную, 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:

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

где z_i – корни многочленов Чебышева $T_n(z) = \cos(n \arccos z)$.

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

4.5. Сплайн-интерполяция

Теорема Фабера. Какова бы ни была стратегия выбора узлов интерполяции, найдется непрерывная на отрезке $[a,b]$ функция f, для которой $\max_{[a,b]} |f(x) - P_n(x)| \to \infty$ при $n \to \infty$.

Таким образом, теорема Фабера отрицает существование единой для всех непрерывных функций стратегии выбора узлов. Проиллюстрируем сказанное примером.

Предположим, что выбираем равноотстоящие узлы. Покажем, что для функции Рунге такая стратегия является неудачной.

Пример 4.5. Глобальная интерполяция функции Рунге. В среде MathCAD пример представлен в файле *R8_ex5.mcd* электронного комплекса.

В тех случаях, когда промежуток $[a, b]$, на котором нужно проинтерполировать недостаточно гладкую функцию $f(x)$ велик, нет смысла пытаться повышать качество ее полиномиальной интерполяции за счет использования многочленов высоких степеней. Более перспективным в этих условиях является ***кусочно-полиномиальная интерполяция***. Интерполирующая функция в этом случае составляется из отдельных мно-

гочленов, как правило, одинаковой небольшой степени, определенных каждый на своей части отрезка $[a, b]$.

Система Mathcad предоставляет возможность аппроксимации двумя важными классами функций: кусочно-линейной и сплайнами.

При кусочно-линейной интерполяции узловые точки соединяются отрезками прямых, то есть через каждые две точки $(x_i, f(x_i))$, $(x_{i+1}, f(x_{i+1}))$ проводится полином первой степени.

Пример 4.6. Кусочно-линейная интерполяция функции Рунге. В среде MathCAD пример представлен в файле *R8_ex6.mcd* электронного комплекса.

Как видно из приведенного примера этот способ приближения также имеет недостаток: в точках "стыка" двух соседних многочленов производная, как правило, имеет разрыв.

Если исходная функция была гладкой и требуется, чтобы и аппроксимирующая функция была гладкой, то кусочно-линейная интерполяция неприемлема. В этом случае применяют сплайны – специальным образом построенные гладкие кусочно-многочленные функции.

Пусть отрезок $[a,b]$ разбит точками на n частичных отрезков $[x_i, x_{i+1}]$. ***Сплайном*** степени m называется функция $S_m(x)$, обладающая следующими свойствами:

1) функция $S_m(x)$ непрерывна на отрезке $[a,b]$ вместе со своими производными до некоторого порядка p;

2) на каждом частичном отрезке $[x_{i-1}, x_i]$ функция совпадает с некоторым алгебраическим многочленом $P_{m,i}(x)$ степени m.

Разность $m - p$ между степенью сплайна и наивысшим порядком непрерывной на отрезке $[a,b]$ производной называют ***дефектом сплайна***. Кусочно-линейная функция является сплайном первой степени с дефектом, равным единице. Действительно, на отрезке $[a,b]$ сама функция $S_1(x)$ (нулевая производная) непрерывна. В то же время на каждом частичном отрезке $S_1(x)$ совпадает с некоторым многочленом первой степени.

Пример 4.7. Построение параболического сплайна [2, с. 64].

Наиболее широкое распространение получили сплайны 3-й степени (кубические сплайны) $S_3(x)$ с дефектом, равным 1 или 2. Система MathCAD для осуществления сплайн-интерполяции кубическими полиномами предусматривает несколько встроенных функций. Одна из них рассмотрена в примере.

Пример 4.8. Построение сплайн-интерполяции. В среде MathCAD пример представлен в файле *R8_ex8.mcd* электронного комплекса.

Погрешность приближения кубическими сплайнами. Пусть функция f имеет на отрезке $[a,b]$ непрерывную производную четвертого порядка и $M_4 = \max_{[a,b]} \left| f^{(4)}(x) \right|$. Тогда для интерполяционного кубического сплайна справедлива оценка погрешности $\max\limits_{[a,b]} \left| f(x) - S_3(x) \right| \le C M_4 h_{\max}^4$.

Достоинства метода: минимальная возможная кривизна интерполяционной функции. Недостаток: вычислительные затраты.

Различные методы интерполяции, реализованные в MathCAD, приведены в [2, с. 53 – 62].

Методы интерполяции рассматриваются в практической работе № 8 [2, с.50].

Контрольные вопросы

1. Сформулируйте постановку задачи приближения функции в широком смысле.
2. В чем сущность непрерывного и дискретного приближения функции?
3. Какие виды приближений функции вам известны?
4. Сформулируйте постановку задачи приближения функции по методу интерполяции.
5. Поясните понятие экстраполяции функции.
6. Назовите основные проблемы интерполяции.
7. Поясните сущность глобальной и локальной интерполяции.
8. Общий вид интерполяционного многочлена Лагранжа.
9. Приведите интерполяционные многочлены Лагранжа первой, второй степени.
10. Поясните случай равноотстоящих узлов интерполяции.
11. Дайте понятие конечных разностей.
12. Приведите интерполяционный многочлен Ньютона.
13. Запишите первую и вторую интерполяционные формулы Ньютона.
14. Дайте понятие разделенных разностей (разностных отношений).
15. В каком случае применяются разделенные конечные разности при записи интерполяционного многочлена Ньютона?

16. Какие преимущества имеет запись интерполяционного многочлена по формуле Ньютона перед формулой Лагранжа?

17. Как используется метод Чебышева для выбора узлов интерполяции?

18. Объясните разницу между глобальной и кусочно-полиномиальной интерполяцией.

19. Дайте определение интерполяционного сплайна m-й степени.

20. Что такое дефект сплайна?

21. Запишите формулу сплайна первой степени с дефектом единица.

5. ПРИБЛИЖЕНИЕ ФУНКЦИЙ. АППРОКСИМАЦИЯ

5.1. Постановка задачи аппроксимации

На практике часто возникает необходимость найти функциональную зависимость между величинами x и y, которые получены в результате эксперимента. В отличие от интерполяции не требуется, чтобы аппроксимирующая функция проходила через все заданные точки, что особенно важно при аппроксимации данных, заведомо содержащих погрешности. Выполняют аппроксимацию и при желании получить упрощенное математическое описание сложной зависимости. Часто вид эмпирической зависимости известен, но числовые параметры неизвестны.

Выделяют две основные задачи аппроксимации:

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.

Близость исходной и аппроксимирующей функций определяется числовой мерой – *критерием аппроксимации* (*близости*). В зависимости от применяемого критерия различают методы аппроксимации.

5.2. Метод наименьших квадратов

Наиболее распространенным методом аппроксимации экспериментальных данных является ***метод наименьших квадратов***. Метод позволяет

использовать аппроксимирующие функции произвольного вида и относится к группе глобальных методов. Критерием близости в методе наименьших квадратов является требование минимальности суммы квадратов отклонений аппроксимирующей функции $g(x)$ от экспериментальных точек y_i

$$\Phi = \sum_{i=0}^{n} (y_i - g(x_i))^2 \to \min .$$

При заданной структуре аппроксимирующей функции $g(x)$ необходимо таким образом подобрать параметры этой функции, чтобы получить наименьшее значение критерия близости, т.е. наилучшую аппроксимацию.

Важной особенностью метода является то, что аппроксимирующая функция может быть произвольной. Ее вид определяется особенностями решаемой задачи, например физическими соображениями, если проводится аппроксимация результатов физического эксперимента. Наиболее часто встречаются аппроксимация прямой линией (***линейная регрессия***), аппроксимация полиномом (***полиномиальная регрессия***), ***аппроксимация линейной комбинацией произвольных функций***. Кроме того, часто возможно путем замены переменных свести задачу к линейной (провести линеаризацию). Например, пусть аппроксимирующая функция ищется в виде $y = A\exp(kx)$. Прологарифмируем это выражение и введем обозначения $z = \ln(y)$, $a = \ln(A)$. Тогда в новых обозначениях задача сводится к отысканию коэффициентов линейной функции $z = a + kx$.

Постановка задачи приближения функции по методу наименьших квадратов для полиномиальной регрессии. Пусть функция *y=f(x)* задана таблицей своих значений: $y_i = f(x_i)$, *i*=0, 1,…, *n*. Требуется найти многочлен фиксированной степени *m*, для которого среднеквадратичное отклонение (СКО, или стандартное отклонение) $\sigma = \sqrt{\frac{1}{n+1}\sum_{i=0}^{n}(y_i - P_m(x_i))^2}$ минимально.

Так как многочлен $P_m(x) = a_0 + a_1 x + a_2 x^2 + ... + a_m x^m$ определяется своими коэффициентами, то фактически нужно подобрать набор коэффициентов $a_0, a_1, ..., a_m$, минимизирующий функцию

$$\Phi(a_0, a_1, ..., a_m) = \sum_{i=0}^{n}(y_i - P_m(x_i))^2 = \sum_{i=0}^{n}\left(y_i - \sum_{j=0}^{m} a_j x_i^j\right)^2 .$$

Искомые переменные $a_0, a_1, ..., a_m$ можно найти из необходимого условия минимума Φ по этим переменным $\frac{\partial \Phi}{\partial a_k} = 0$, k=0,1,…, m.

$$\frac{\partial \Phi}{\partial a_k} = 2\sum_{i=0}^{n}\left(y_i - \sum_{j=0}^{m} a_j x_i^j \right)\left(-x_i^k\right) = 0, \quad i = 0,1,...,n.$$

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

$$\frac{\partial \Phi}{\partial a_k} = \sum_{i=0}^{n}\left(y_i\left(-x_i^k\right)\right) + \sum_{i=0}^{n}\sum_{j=0}^{m} a_j x_i^j x_i^k = \sum_{i=0}^{n} y_i\left(-x_i^k\right) + \sum_{j=0}^{m}\left(a_j \sum_{i=0}^{n} x_i^{j+k} \right).$$

Перепишем последние равенства

$$\sum_{j=0}^{m}\left(\sum_{i=0}^{n} x_i^{j+k} \right) a_j = \sum_{i=0}^{n} y_i x_i^k, \quad k = 0,1,...,m.$$

Полученная система – есть система алгебраических уравнений относительно неизвестных $a_0, a_1, ..., a_m$. Ее называют ***нормальной системой метода наименьших квадратов.*** Можно показать, что определитель этой системы отличен от нуля, то есть решение существует и единственно. Однако при высоких степенях m система является плохо обусловленной. Т.е. при изменении даже одного значения исходных данных все коэффициенты изменят в общем случае свои значения. Кроме того, коэффициенты полинома связаны между собой, т.к. находятся из решения СЛАУ. Это приводит к тому, что если какой-то коэффициент вследствие его малости захочется отбросить, придется пересчитывать заново оставшиеся. Существует специальная теория планирования экспериментов, которая позволяет обосновать и рассчитать значения x_i, используемые для аппроксимации, чтобы получить заданные свойства коэффициентов или аппроксимирующей функции. В основном метод наименьших квадратов применяют для нахождения многочленов, степень которых не выше 5. Решение нормальной системы можно найти, например, методом Гаусса.

Запишем ***нормальную систему наименьших квадратов*** для двух простых случаев: m=0 и m=2. При m=0 многочлен примет вид $P_0(x) = a_0$. Для нахождения неизвестного коэффициента a_0 имеем уравнение

$$(n+1)a_0 = \sum_{i=0}^{n} y_i \, .$$

Получаем, что коэффициент a_0 есть среднее арифметическое значений функции в заданных точках

$$a_0 = \frac{1}{n+1}\sum_{i=0}^{n} y_i \ .$$

Если же используется многочлен второй степени $P_2(x) = a_0 + a_1 x + a_2 x^2$, то нормальная система уравнений примет вид

$$\begin{cases} (n+1)a_0 + \left(\sum_{i=0}^{n} x_i\right)a_1 + \left(\sum_{i=0}^{n} x_i^2\right)a_2 = \sum_{i=0}^{n} y_i; \\ \left(\sum_{i=0}^{n} x_i\right)a_0 + \left(\sum_{i=0}^{n} x_i^2\right)a_1 + \left(\sum_{i=0}^{n} x_i^3\right)a_2 = \sum_{i=0}^{n} y_i x_i; \\ \left(\sum_{i=0}^{n} x_i^2\right)a_0 + \left(\sum_{i=0}^{n} x_i^3\right)a_1 + \left(\sum_{i=0}^{n} x_i^4\right)a_2 = \sum_{i=0}^{n} y_i x_i^2. \end{cases}$$

Пример 5.1. Приближение функции по методу наименьших квадратов [2, с. 68].

Предположим, что функцию *f* можно с высокой точностью аппроксимировать многочленом $P_m(x)$ некоторой степени *m*. Если эта степень заранее не известна, то возникает проблема выбора оптимальной степени аппроксимирующего многочлена в условиях, когда исходные данные y_i содержат случайные ошибки. Для решения этой задачи можно принять следующий алгоритм: для каждого *m*=0,1,2,.. вычисляется величина

$$\sigma_m = \sqrt{\frac{1}{n-m}\sum_{i=0}^{n}(y_i - P_m(x_i))^2} \ .$$

За оптимальное значение степени многочлена следует принять то значение *m*, начиная с которого величина σ_m стабилизируется или начинает возрастать.

Пример 5.2. Нахождение оптимальной степени многочлена. В среде MathCAD пример представлен в файле *R9_ex2.mcd* электронного комплекса.

Определение параметров эмпирической зависимости. Часто из физических соображений следует, что зависимость $y = f(x)$ между величинами хорошо описывается моделью вида $y = g(x, a_0, a_1, ..., a_m)$, где вид

зависимости g известен. Тогда применение критерия наименьших квадратов приводит к задаче определения искомых параметров $a_0, a_1, ..., a_m$ из условия минимума функции

$$\Phi(a_0, a_1, ..., a_m) = \sum_{i=0}^{n} (g(x_i, a_0, a_1, ..., a_m) - y_i)^2 .$$

Пример 5.3. Вывод нормальной системы уравнений для нахождения параметров эмпирической зависимости [2, с. 69].

Если зависимость от параметров $a_0, a_1, ..., a_m$ нелинейна, то экстремум функции $\Phi(a_0, a_1, ..., a_m) = \sum_{i=0}^{n} (g(x_i, a_0, a_1, ..., a_m) - y_i)^2$ ищут методами минимизации функций нескольких переменных.

5.3. 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.

$$E = \max|Q_m(x_i) - y_i| \to \min .$$

Доказано, что многочлен $Q_m(x)$, называемый ***многочленом наилучшего равномерного приближения***, является единственным для данного значения m.

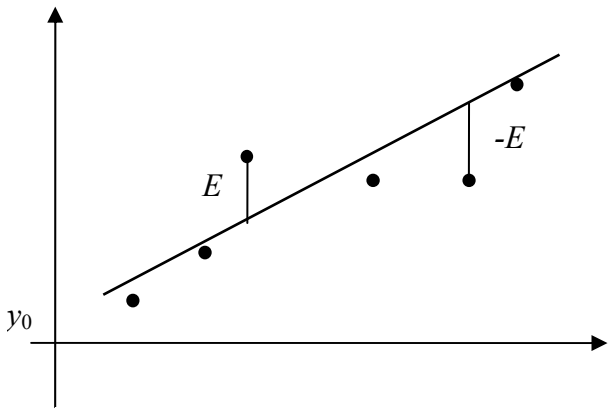


Рис. 5.1. Многочлен наилучшего равномерного приближения

Теорема Чебышева. Многочлен $Q_m(x)$ является многочленом наилучшего равномерного приближения для функции $f(x)$ на $[a, b]$ тогда и только тогда, когда на $[a, b]$ существует не менее $m + 2$ точек x_i таких, что в них поочередно принимаются наибольшие положительные и отрицательные отклонения, т.е. поочередно разность $Q_m(x_i) - f(x_i)$ равна E или $-E$ (рис. 5.1).

К сожалению, неизвестны ни общий вид многочленов равномерных приближений, ни способы их построения. Имеются лишь методики построения многочленов, близких к наилучшим равномерным, а также способы построения равномерных приближений невысокого порядка для нескольких весьма узких классов функций.

Многочлен наилучшего равномерного приближения нулевой степени

$$Q_0(x) = a_0 = \frac{m+M}{2},$$

где $m = \min f(x)$, $M = \max f(x)$ на $[a,b]$ (рис. 5.2).

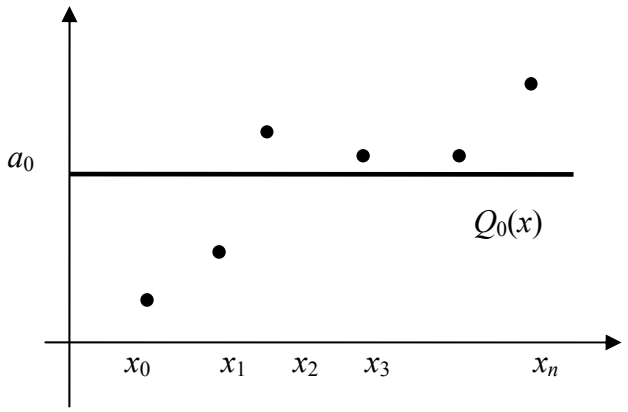


Рис. 5.2. Многочлен наилучшего равномерного приближения нулевой степени

5.4. Аппроксимация в MathCAD

MathCAD содержит достаточное количество встроенных средств для анализа данных:

– ***line*** (x, y) – линейная регрессия;

– ***regress*** (x, y, m), ***interp*** (vs, x, y, z) – полиномиальная регрессия;

– ***expfit*** (x, y, g) – регрессия экспоненциальной функцией $f(x) = ae^{bx} + c$;

– ***lgsfit*** (x, y, g) – регрессия логистической функцией $f(x) = a/(1 + be^{-cx})$;

– ***sinfit*** (x, y, g) – регрессия синусоидой $f(x) = a\sin(x + b) + c$;

– ***pwfit*** (x, y, g) – регрессия степенной функцией $f(x) = ax^b + c$;

– ***logfit*** (x, y, g) – регрессия логарифмической функцией $f(x) = a\ln(x + b) + c$;

– ***lnfit*** (x, y, g) – регрессия двухпараметрической логарифмической функцией $f(x) = a\ln(x) + b$;

– ***linfit*** – аппроксимация линейной комбинацией функций;

– ***genfit*** – аппроксимация функцией произвольного вида (регрессия общего вида);

– ***predict*** (*y, m, n*) – функция предсказания вектора, экстраполирующего выборку данных *y* размером *n*, взятых через равные промежутки значений аргумента; результат вставляется в «хвост» исходных данных;

– ***medsmooth*** (*y, b*) – сглаживание исходного вектора *y* алгоритмом «бегущих медиан» с шириной окна сглаживания *b*;

– ***ksmooth*** (*x, y, b*) – сглаживание на основе функции Гаусса;

– ***supsmooth*** (*x, y*) – локальное сглаживание адаптивным алгоритмом, основанное на анализе ближайших соседей каждой пары данных.

Методы приближения и аппроксимации функций рассматриваются в практической работе № 9 [2, с. 68].

Контрольные вопросы

1. Постановка задачи приближения функции по методу аппроксимации.
2. Какие разновидности задачи аппроксимации функции вы можете назвать?
3. В чем сущность аппроксимации функций методом наименьших квадратов?
4. Каков вывод нормальной системы метода наименьших квадратов для полиномиальной регрессии?
5. Как осуществляется запись нормальной системы метода наименьших квадратов для полиномиальной регрессии нулевой, первой и второй степени?
6. Выбор оптимальной степени аппроксимирующего многочлена.
7. Что такое многочлен наилучшего равномерного приближения?
8. Аппроксимация в MathCAD.

6. ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ

6.1. Квадратурные формулы

Вычисление интегралов встречается достаточно часто, например, при моделировании. Далеко не всегда задача может быть решена аналитически. Численные методы обычно применяются при взятии неберущихся интегралов от достаточно сложных функций, которые предварительно табулируются, или при интегрировании таблично заданных функций, что часто встречается в экономических приложениях.

Пусть требуется вычислить определенный интеграл на интервале $[a;b]$: $\int_a^b f(x)dx$.

Для численного интегрирования подынтегральную функцию аппроксимируют какой-либо более простой функцией, интеграл от которой может быть вычислен. Обычно в качестве аппроксимирующей функции используют полином, вычисляется интеграл по так называемым ***квадратурным формулам Ньютона-Котеса***. В них интеграл заменяется взвешенной суммой ординат подынтегральной функции в отдельных точках

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

Название (квадратурные формулы) связано с геометрической интерпретацией определенного интеграла: вычисление $\int_a^b f(x)dx$ при $f(x) \geq 0$ равносильно построению квадрата, равновеликого криволинейной трапеции с основанием $[a; b]$ и «крышей» $f(x)$ (рис. 6.1). В случае полинома нулевой степени метод численного интегрирования называют ***методом прямоугольников*** (подынтегральная функция заменяется горизонтальной прямой), в случае полинома первой степени – ***методом трапеций*** (подынтегральная функция заменяется наклонной прямой), в случае полинома второй степени – ***методом Симпсона*** (подынтегральная функция заменяется параболой 2-го порядка).

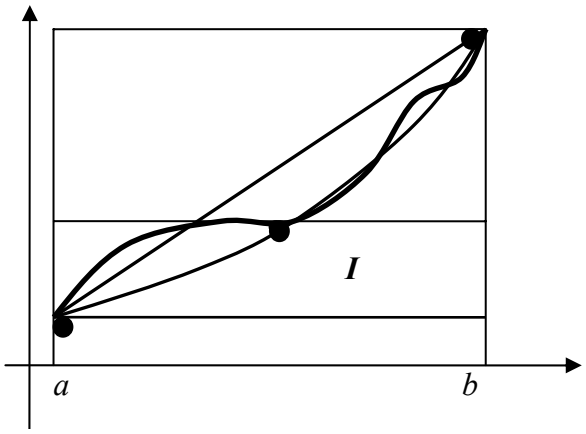


Рис. 6.1. Геометрическая интерпретация квадратурных формул

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 (рис. 6.2). Все методы различаются значениями ординат x_i и весов A_i. Погрешность можно оценить величиной остаточного члена R.

Приведем простейшие квадратурные формулы и оценки погрешностей (остаточный член R) для вычисления $\int_a^b f(x)dx$ для элементарного отрезка интегрирования $[a, b]$ и при равномерном делении отрезка $[a; b]$ на n интервалов с шагом $h=(b-a)/n$.

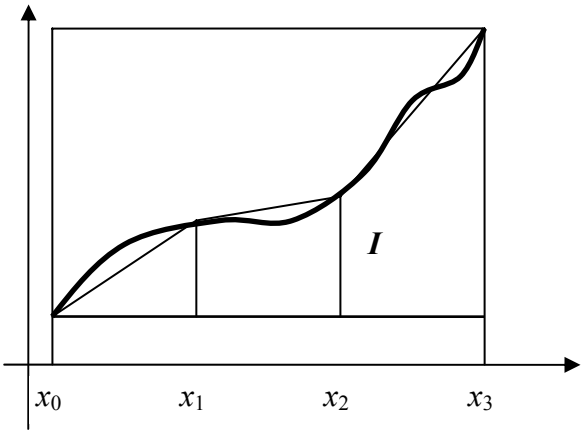


Рис. 6.2. Метод трапеций для трех интервалов

Формула левых прямоугольников

$$S_э = f(a)(b-a), \qquad S = h\sum_{i=0}^{n-1} f(x_i);$$

$$R = \frac{M_1(b-a)h}{2}.$$

Формула правых прямоугольников

$$S_э = f(b)(b-a); \qquad S = h\sum_{i=1}^{n} f(x_i); \qquad R = \frac{M_1(b-a)h}{2}.$$

Формула центральных (средних) прямоугольников

$$S_э = f\left(\frac{a+b}{2}\right)(b-a); \quad S = h\sum_{i=1}^{n} f\left(x_{i-1}+\frac{h}{2}\right) = h\sum_{i=1}^{n} f\left(x_i-\frac{h}{2}\right);$$

$$R = \frac{M_2(b-a)h^2}{24}.$$

Формула трапеций

$$S_э = \frac{f(a)+f(b)}{2}(b-a); \quad S = h\left(\frac{f(x_0)+f(x_n)}{2} + \sum_{i=1}^{n-1} f(x_i)\right);$$

$$R = \frac{M_2(b-a)h^2}{12}.$$

Формула Симпсона

$$S_э = \frac{b-a}{2\cdot 3}(f(a)+4f\left(\frac{b+a}{2}\right)+f(b));$$

$$S = \frac{h}{3}\left(f(x_0)+4\sum_{i=1}^{n/2} f(x_{2i-1})+2\sum_{i=1}^{n/2} f(x_{2i})+f(x_n)\right);$$

$$R = \frac{M_4 (b-a) h^4}{2880}.$$

Аналитические выражения для определения погрешности требуют знания производных до четвертого порядка (M_1, M_2, M_4) от подынтегральной функции. В большинстве случаев погрешность численного интегрирования определяется путем двойного интегрирования: с исходным шагом и с шагом, увеличенным в два раза. Разница вычисленных значений интегралов определяет погрешность $R \le \frac{|I_n - I_{n/2}|}{3}$ – для формул прямоугольников и трапеций; $R \le \frac{|I_n - I_{n/2}|}{15}$ – для формулы Симпсона. 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).

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

6.2. Другие методы численного интегрирования

Метод Ромберга. В MathCAD метод Ромберга используется при реализации встроенного оператора интегрирования для большинства функций без особенностей.

В методе трапеций подынтегральную функцию аппроксимируют полиномом первой степени, то есть прямой линией. Это значит, что вместо площади криволинейной трапеции мы будем искать площадь прямоугольной трапеции. Приближенное значение интеграла

$$\int_a^b f(x)dx \approx \frac{f(b)-f(a)}{2}(b-a).$$

Погрешность этой формулы равна $O(h^3 f'')$.

Обозначим $R(1;1) = \frac{h}{2}(f(a)-f(b))$, где $h = b-a$. Смысл введенного обозначения станет ясен несколько позже.

Оценку значения интеграла можно сделать более точной, если разбить интервал на n частей и применить формулу трапеций для каждого такого интервала

$$h(\frac{1}{2}(f(a)+f(b))+f(a+h)+f(a+2h)+\ldots+f(a+(n-1)h))).$$

Если разбить интервал на две части, то есть уменьшить шаг в два раза $h=(b-a)/2$, то оценка для величины интеграла будет иметь вид

$$R(2;1)=\frac{h}{2}(f(a)+f(b))+hf(a+h)=\frac{1}{2}R(1;1)+h\sum_{i=1}^{2^*-1}f(a+ih).$$

В данном случае суммирование включает только один элемент. Обратите внимание, в новую оценку вошла старая оценка. Нам потребовалось определять значение функции только в новых узлах.

Если имеется 2^n подынтервалов, то

$$R(n+1;1)=\frac{h}{2}(f(a)+f(b))+h\sum_{i=1}^{2^n-1}f(a+ih),\qquad h=\frac{b-a}{2^n}.$$

Если n=0, то $R(1;1)=\frac{h}{2}(f(a)+f(b))$.

Если n=1, то $R(2;1)=\frac{h}{2}(f(a)+f(b))+hf(a+h)=\frac{1}{2}R(1;1)+hf(a+h)$.

Если n=2, то $R(3;1)=\frac{h}{2}(f(a)+f(b))+hf(a+h)+hf(a+2h)+hf(a+3h)=$

$=\frac{1}{2}R(2;1)+hf(a+h)+hf(a+3h).$

Вообще, справедливо рекуррентное соотношение

$$R(n+1;1)=\frac{1}{2}R(n;1)+h\sum_{i=1}^{2^n-1}f(a+(2i-1)h).$$

Полученное соотношение называют рекурсивной формулой трапеций и часто применяют для вычисления определенных интегралов. Преимущество этой формулы состоит в том, что при увеличении числа подынтервалов функцию нужно вычислять только во вновь добавленных точках. К сожалению, с помощью этой формулы нельзя получить сколь угодно точное значение интеграла. Во-первых, при увеличении числа разбиений объем вычислений стремительно возрастает; во-вторых, на каждом шаге накапливается ошибка округлений. Для дальнейшего уточнения значения интеграла можно сделать следующий шаг – ***экстраполировать*** полученную последовательность значений на случай бесконечного числа точек или, что то же самое, на случай нулевого шага. Такой подход называется ***методом Ромберга***.

Метод Ромберга заключается в том, что полученные оценки значения интеграла экстраполируют на случай бесконечного числа разбиений (величины шага равной нулю) по рекуррентной формуле

$$R(n+1,m+1) = R(n+1,m) + \frac{R(n+1,m) - R(n,m)}{4^m - 1}.$$

То есть строится следующий треугольник:

$R(1,1)$

$R(2,1)$ $R(2,2)$

$R(3,1)$ $R(3,2)$ $R(3,3)$

$R(4,1)$ $R(4,2)$ $R(4,3)$ $R(4,4)$

$R(5,1)$ $R(5,2)$ $R(5,3)$ $R(5,4)$ $R(5,5)$,

в котором первый столбец состоит из значений интеграла, полученных при последовательном удвоении числа интервалов. Второй столбец – результат уточнения значений первого столбца по рекуррентной формуле. Третий столбец – уточненные значения интеграла на основе второго столбца и т.д.

Формула Ромберга может быть получена различными способами. Можно, например, воспользоваться методом Невиля. Пусть имеется набор точек (x_i, y_i). Обозначим P_i полином нулевой степени, проходящий через i-ю точку. Обозначим $P_{i(i+1)}$ полином первой степени, проходящий через точки i и i+1. Совершенно аналогично будет означать $P_{12\ldots(n-1)n}$ полином n–1 степени, проходящий через все n точек. Легко убедиться, что

$$P_{i(i+1)\ldots(i+m)} = \frac{(x - x_{i+m})P_{i(i+1)\ldots(i+m-1)} + (x_i - x)P_{(i+1)(i+2)\ldots(i+m)}}{x_i - x_{i+m}}.$$

В нашем случае $x_i = \frac{b-a}{2^{i-1}}$. В качестве y_i выступают $R(i, m)$. Мы хотим получить значение интеграла в пределе $h \to 0$, поэтому x=0.

Метод Чебышева. Все предыдущие методы имели следующую особенность: значения x располагались равномерно, а весовые коэффициенты были разными (в общем случае). В методе Чебышева приняты все весовые коэффициенты A_i одинаковыми, а x_i – разными. Переменная интегрирования предварительно приводится к диапазону [–1, 1]:

$$x_i = \frac{a+b}{2} + \frac{b-a}{2} t_i.$$

Метод Гаусса. В методе Гаусса различны и значения узлов t_i и веса A_i. Местоположение и длина интервалов разбиения отрезка [*a, b*] подбираются таким образом, чтобы достичь наивысшей точности при заданном числе интервалов.

Значения узлов t_i *и весов* A_i *квадратурной формулы Гаусса* $\int_a^b f(x)dx \approx \frac{b-a}{2}\sum_{i=0}^{N} A_i f\left(\frac{a+b}{2}+\frac{b-a}{2}t_i\right)$ с числом узлов от 1 до 4 даны в [2, с.77].

В этих методах вследствие неравномерного шага интегрирования нельзя оценить погрешность интегрирования двойным просчетом. Для этой цели применяются другие достаточно сложные алгоритмы.

Метод Монте-Карло.

Пусть $f(x)$ целиком лежит внутри прямоугольника с площадью $S = d \cdot (b-a)$. Сгенерируем N пар случайных чисел, равномерно распределенных в данном прямоугольнике: $a \le x_i \le b$, $0 \le y_i \le d$. Оценка интеграла: $I = S\frac{n}{N}$, где n – количество точек, для которых $y_i \le f(x_i)$.

Пример 6.1. Вычисление интеграла методом Монте-Карло в среде MathCAD приведено на рис. 6.3, где ***runif*** **(*N, a, b*)** – встроенная функция MathCAD генерирует вектор из N случайных чисел, равномерно распределенных на отрезке $[a,b]$.

Применение метода Монте-Карло для вычисления кратных интегралов: хотя Mathcad позволяет вычислять кратные интегралы непосредственно, однако в большинстве случаев при кратности интегралов 3 и более применение метода Монте-Карло предпочтительнее. Дело в том, что при одинаковой точности метод Монте-Карло дает существенный выигрыш во времени (в десятки и сотни раз), особенно при большой кратности интегралов. Заключим область интегрирования внутрь прямоугольной области, "набросаем" внутрь полученной области N случайных точек. Тогда интеграл найдем из соотношения $I = V \cdot \frac{n}{N}$, где N – общее число точек, n – число точек, лежащих внутри области интегрирования, V – объем области, включающей область интегрирования.

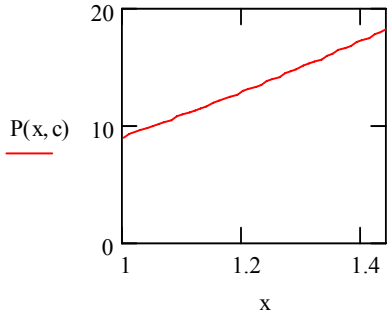
а) Многочлен $P(x,c) := c_0 + c_1 \cdot x + c_2 \cdot x^2 + c_3 \cdot x^3$

Коэффициенты многочлена: $c_0 := -1 \quad c_1 := 1 \quad c_2 := 10 \quad c_3 := -1$

Концы отрезка интегрирования: $a := 1 \quad b := 1.44$

$x := a, a + 0.01 .. b$

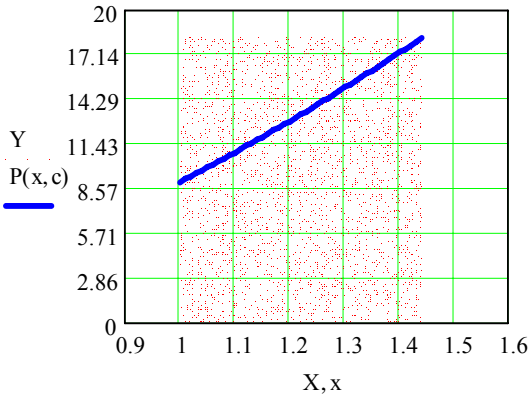
$\int_a^b P(x,c)\,dx \to 5.8917924266666666666$



$\int_a^b P(x,c)\,dx = 5.89179243$

Значение интеграла, вычисленное аналитически: $I := 5.891792$

$N := 3000 \quad i := 0 .. N - 1 \quad X := runif(N, a, b) \quad Y := runif(N, 0, P(b,c))$



$s := P(b,c) \cdot (b - a) \qquad s = 8.00360704$

$n := \sum_i if\left(Y_i \le P\left(X_i, c\right), 1, 0\right) \qquad I1 := s \cdot \frac{n}{N} \qquad I1 = 5.90932986$

$R := |I - I1| \qquad R = 0.01753786$

Рис. 6.3. Вычисление интеграла методом Монте-Карло

Пример 6.2. Вычисление кратного интеграла методом Монте-Карло (рис. 6.4). Максимальное значение подынтегральной функции в области интегрирования не превосходит 125, следовательно, мы может заключить всю область интегрирования внутрь четырехмерного цилиндроида высотой 125 и объемом V=125. Сгенерируем N четверок случайных чисел и подсчитаем, сколько из них лежит под поверхностью $f(x, y, z)$.

б) Функция $f(x,y,z) := 125 - x^2 - y^2 - z^2$ $\int_0^1 \int_0^1 \int_0^1 f(x,y,z)\,dx\,dy\,dz \to 124$ $I := 124$

$$\int_0^1 \int_0^1 \int_0^1 f(x,y,z)\,dx\,dy\,dz = 124$$

$N := 3000$ $i := 0..N-1$ $X := runif(N,0,1)$ $Y := runif(N,0,1)$ $Z := runif(N,0,1)$ $F := runif(N,0,125)$

$n := \sum_i if(F_i \le f(X_i, Y_i, Z_i), 1, 0)$ $V := 1 \cdot 1 \cdot 1 \cdot 125$ $I1 := V \cdot \frac{n}{N}$ $I1 = 123.91666667$

$R1 := |I - I1|$ $R1 = 0.08333333$

Рис. 6.4. Вычисление кратного интеграла методом Монте-Карло

Методы численного интегрирования рассматриваются в практической работе № 10 [2, с. 76].

Контрольные вопросы

1. В чем сущность численного интегрирования?
2. Геометрическая интерпретация и формулы численного интегрирования методом прямоугольников.
3. Как в методе прямоугольников уменьшить погрешность нахождения интеграла?
4. В каких случаях находит применение метод прямоугольников?
5. Геометрическая интерпретация и формулы численного интегрирования методом трапеций.
6. Как уменьшить в методе трапеций погрешность нахождения интеграла?
7. В каких случаях находит применение метод трапеций?
8. Какой аппроксимирующей заменяется подынтегральная функция в методе Симпсона?
9. Определение погрешности численного интегрирования.
10. В чем сущность методов Чебышева и Гаусса?
11. Численное интегрирование методом Монте-Карло.

7. РЕШЕНИЕ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

7.1. Концепция численного решения дифференциальных уравнений

Дифференциальные уравнения (ДУ) часто встречаются при построении моделей динамики объектов исследования. Они описывают, как правило, изменение параметров объекта во времени. Результатом решения ДУ являются функции, а не числа, как при решении конечных уравнений, вследствие чего методы решения их более трудоемки. Особенно это касается ДУ в частных производных.

Пусть необходимо найти решение обыкновенного дифференциального уравнения (ОДУ)

$$\frac{dy}{dx} = f(x, y) \quad \text{или} \quad y' = f(x, y) \tag{7.1}$$

с начальным условием $y(x_0) = y_0$. Такая задача называется ***задачей Коши***.

Методы решения ДУ подразделяются на три группы:

1) аналитические;
2) графические;
3) численные.

Численное решение задачи Коши состоит в построении таблицы приближенных значений $y_1, y_2, ..., y_n$ в точках $x_1, x_2, ..., x_n$. Точки $x_i = x_0 + ih$, $i = 0,1,..., n$ называются узлами сетки, а величина h – шагом сетки.

Решение носит шаговый характер, т.е. по одной или по нескольким начальным точкам (x, y) за один шаг находят следующую точку и т.д.

7.2. Метод Эйлера и модифицированный метод Эйлера

В основе построения дискретной задачи Коши лежит тот или иной способ замены дифференциального уравнения его дискретным аналогом. Простейший метод основан на замене левой части уравнения правой разностной производной

$$\frac{y_{i+1} - y_i}{h} = f(x_i, y_i).$$

Разрешая уравнение относительно y_{i+1}, получаем расчетную формулу ***метода Эйлера***:

$$y_{i+1} = y_i + f(x_i, y_i)h, , \quad i = 0,1,..., n. \tag{7.2}$$

Ту же формулу можно получить с помощью разложения искомой функции $y(x)$ в ряд Тейлора вблизи точки x_0. Ограничимся первыми двумя членами разложения $y(x) = y(x_0) + y'(x)(x - x_0) + \ldots$. Учтя уравнение (7.1) и обозначив $x - x_0 = \Delta x$, получаем $y(x) = y(x_0) + f(x_0, y_0)\Delta x$. Эту формулу можно применять многократно, находя значения функции во все новых и новых точках.

Геометрически метод Эйлера означает, что на каждом шаге мы аппроксимируем решение (интегральную кривую) отрезком касательной, проведенной к графику решения в начале интервала (рис. 7.1). Отсюда происходит другое название метода – ***метод ломаных***.

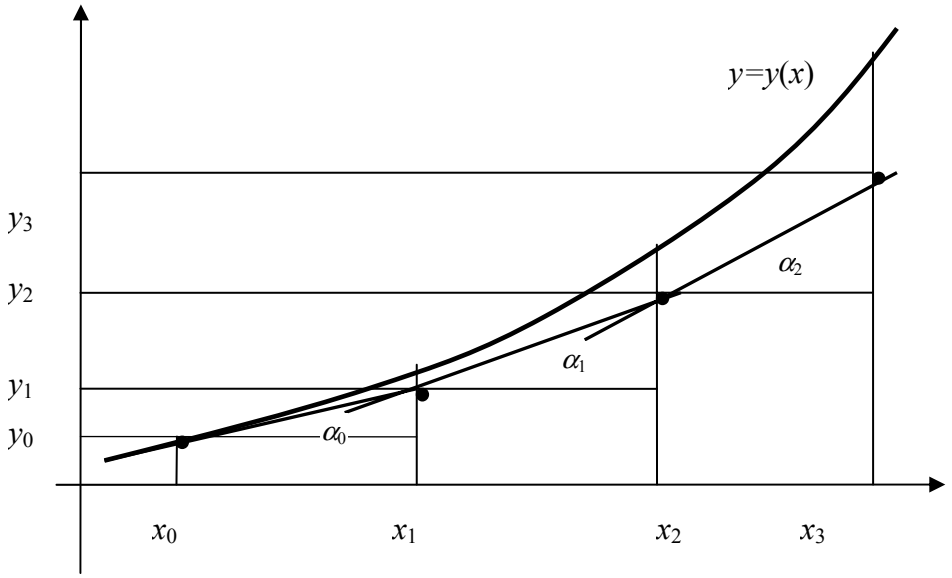


Рис. 7.1. Геометрическая интерпретация метода Эйлера

В соответствии с вычислительной схемой метода и его геометрической интерпретацией имеем

$$y_1 = y_0 + f(x_0, y_0)h = y_0 + y'(x_0)h = y_0 + tg\alpha_0 \cdot h\,.$$

Точность метода невелика и имеет порядок h. Говорят, что метод Эйлера – метод первого порядка, то есть его точность растет линейно с уменьшением шага h.

Пример 7.1. Решение задачи методом Эйлера [2, с. 82].

Численный метод называется ***явным***, если вычисление решения в следующей точке y_{i+1} осуществляется по явной формуле. Метод называется ***одношаговым***, если решение вычисляется в следующей точке y_{i+1} с

использованием только одного предыдущего значения y_i. Метод Эйлера является явным одношаговым методом.

Оценка погрешности метода Эйлера

Локальной погрешностью метода называется величина $l_i = y(x_{i+1}) - y_{i+1}$. Найдем величину локальной погрешности метода Эйлера

$$l_i = y(x_{i+1}) - y_{i+1} = y(x_i) + hy'(x_i) + O(h^2) - y_i - hf(x_i, y_i) = O(h^2)$$

при условии, что $y(x_i) = y_i$. Другими словами, l_i – погрешность, которую допускает за один шаг метод, стартующий с точного решения.

Глобальной погрешностью (или просто погрешностью) численного метода называют сеточную функцию δ^h со значениями $\delta_i = y(x_i) - y_i$ в узлах. В качестве меры абсолютной погрешности метода примем величину $E(h) = \max\limits_{0 \le i \le N} |y(x_i) - y_i|$. Можно показать, что для явных одношаговых методов из того, что локальная погрешность имеет вид $l_i = Ch^{p+1}$, следует, что $E(h) = Mh^p$, где C и M – некоторые константы. Таким образом, метод Эйлера является методом первого порядка точности. Для нахождения решения задачи Коши с заданной точностью ε требуется найти такое приближенное решение y^h, для которого величина глобальной погрешности $E(h) \le \varepsilon$. Так как точное решение задачи неизвестно, погрешность оценивают с помощью ***правила Рунге***.

Правило Рунге оценки погрешностей. Для практической оценки погрешности проводят вычисления с шагами h и $h/2$. За оценку погрешности решения, полученного с шагом $h/2$, принимают величину, равную $\max\limits_i \left| \dfrac{y_i^h - y_{2i}^{h/2}}{2^p - 1} \right|$, где p – порядок метода.

Пример 7.2. Оценка погрешности по правилу Рунге. В среде MathCAD пример представлен в файле *R11_ex2.mcd* электронного комплекса.

Модификации метода Эйлера

Метод Эйлера обладает медленной сходимостью, поэтому чаще применяют методы более высокого порядка точности. Существуют различные модификации метода Эйлера, позволяющие увеличить его точ-

ность. Все они основаны на том, что производную, вычисленную в начале интервала, заменяют средним значением производной на данном интервале. Среднее значение производной можно получить (конечно, только приближенно) различными способами. Можно, например, оценить значение производной в середине интервала $[x_i, x_{i+1}]$ и использовать его для аппроксимации решения на всем интервале

$$x_{i+\frac{1}{2}} = x_i + \frac{h}{2},$$

$$y_{i+\frac{1}{2}} = y_i + \frac{h}{2} f(x_i, y_i),$$

$$y_{i+1} = y_i + hf(x_{i+\frac{1}{2}}, y_{i+\frac{1}{2}}).$$

Усовершенствованный метод Эйлера можно представить одной вычислительной формулой

$$y_{i+1} = y_i + hf(x_i + \frac{h}{2}, y_i + \frac{h}{2} f(x_i, y_i)).$$

Этот метод имеет простую геометрическую интерпретацию. Метод Эйлера называют методом ломаных, так как интегральная кривая на отрезке $[x_i, x_{i+1}]$ заменяется ломаной с угловым коэффициентом $f(x_i, y_i)$. В усовершенствованном методе Эйлера интегральная кривая на отрезке $[x_i, x_{i+1}]$ заменяется ломаной с угловым коэффициентом, вычисленным в средней точке отрезка $x_{i+1/2} = x_i + h/2$ (рис. 7.2). Так как значение $y_{i+1/2}$ в этой точке неизвестно, для его нахождения используют метод Эйлера с шагом $h/2$.

Пример 7.3. Решение задачи усовершенствованным методом Эйлера [2, с.83] (см. рис. 7.2).

Можно также оценить среднее значение производной на интервале

$$\widetilde{y}_{i+1} = y_i + hf(x_i, y_i),$$

$$y_{i+1} = y_i + h\frac{f(x_i, y_i) + f(x_{i+1}, \widetilde{y}_{i+1})}{2}.$$

Так получают еще одну модификацию метода Эйлера второго порядка – метод Эйлера-Коши

$$y_{i+1} = y_i + \frac{h}{2}(f(x_i, y_i) + f(x_{i+1}, y_i + hf(x_i, y_i))).$$

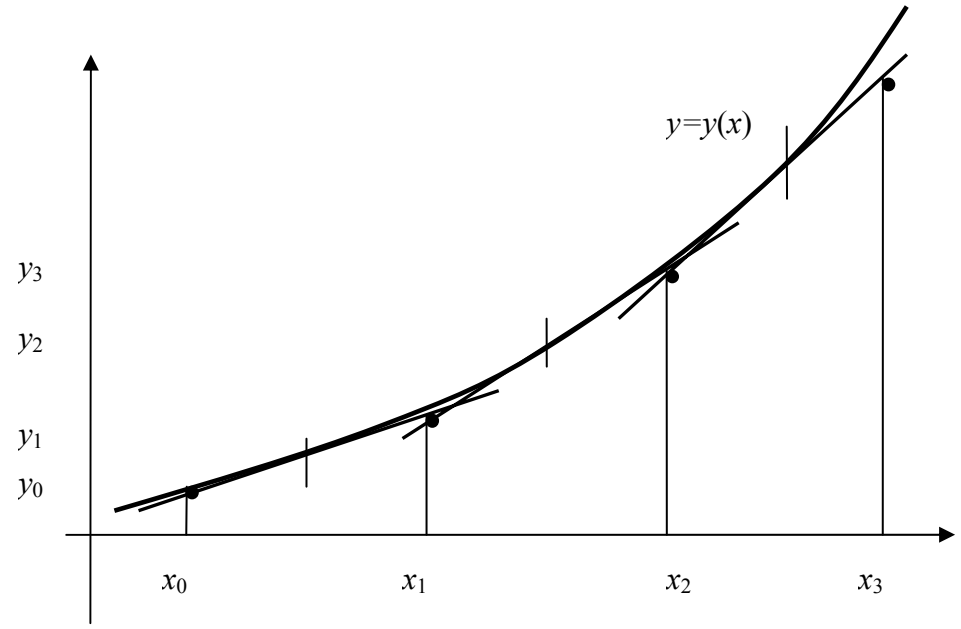


Рис. 7.2. Усовершенствованный метод Эйлера

Решение систем дифференциальных уравнений методом Эйлера

Пусть требуется решить нормальную систему дифференциальных уравнений

$$\begin{cases} y_1' = f_1(x, y_1, ..., y_n), \\ ..... \\ y_n' = f_n(x, y_1, ..., y_n) \end{cases}$$

с начальными условиями $y_1(x_0) = y_{10}$, $y_2(x_0) = y_{20}, ...,$ $y_n(x_0) = y_{n0}$.

Эту систему в векторной форме можно записать в виде

$$\boldsymbol{Y}' = \boldsymbol{f}(x, \boldsymbol{Y}), \qquad \boldsymbol{Y}(x_0) = \boldsymbol{Y}_0,$$

где $\boldsymbol{Y}(x) = \begin{pmatrix} y_1(x) \\ ... \\ y_n(x) \end{pmatrix}$, $\boldsymbol{f} = \begin{pmatrix} f_1(x, y_1, ..., y_n) \\ ... \\ f_n(x, y_1, ..., y_n) \end{pmatrix}$, $\boldsymbol{Y}_0 = \begin{pmatrix} y_{10} \\ ... \\ y_{n0} \end{pmatrix}$.

Расчетная формула метода Эйлера имеет вид

$$\boldsymbol{Y}_{i+1} = \boldsymbol{Y}_i + h\boldsymbol{f}(x_i, \boldsymbol{Y}_i), \qquad i = 0, 1, ..., n.$$

7.3. Метод Рунге-Кутты

Оценку значения производной можно улучшить, увеличивая число вспомогательных шагов. На практике наиболее распространенным методом решения обыкновенных дифференциальных уравнений ***является метод Рунге-Кутты*** четвертого порядка. Следовательно, он более точен,

чем метод Эйлера, который является методом первого порядка, и чем усовершенствованный метод Эйлера (второго порядка). Для оценки значения производной в этом методе используются четыре вспомогательных вычислений функции, но метод все равно остается одношаговым. Формулы метода Рунге-Кутты следующие:

$$k_1^i = hf(x_i, y_i),$$

$$k_2^i = hf(x_i + \frac{h}{2}, y_i + \frac{k_1^i}{2}),$$

$$k_3^i = hf(x_i + \frac{h}{2}, y_i + \frac{k_2^i}{2}),$$

$$k_4^i = hf(x_i + h, y_i + k_3^i),$$

$$\Delta y_i = \frac{1}{6}(k_1^i + 2k_2^i + 2k_3^i + k_4^i),$$

$$y_{i+1} = y_i + \Delta y_i.$$

Перечисленные методы можно применять и для решения систем дифференциальных уравнений. Поскольку многие дифференциальные уравнения высших порядков могут быть сведены заменой переменных к системе дифференциальных уравнений первого порядка, рассмотренные методы могут быть использованы и для решения дифференциальных уравнений порядка выше первого.

7.4. Многошаговые методы

Многошаговые методы требуют для нахождения следующего значения функции нескольких предыдущих. Поэтому они не обладают свойством «самостартования», начинают решать задачу Коши всегда одношаговыми методами. Кроме того, невозможно изменять в процессе решения величину шага для повышения эффективности решения.

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: $y^{\text{п}}_{i+1} = y_{i-3} + 4\frac{h}{3}(2f_{i-2} - f_{i-1} + 2f_i)$, где для компактности записи использовано следующее обозначение $f_i = f(x_i, y_i)$;

– этап коррекции: $y_{i+1} = y_{i-1} + \frac{h}{3}(f_{i+1} + 4f_i + y^{\text{п}}_{i+1})$.

Абсолютная погрешность определяется по формуле $\varepsilon \approx \left| y_{i+1} - y^{\text{п}}_{i+1} \right| / 29$.

Метод требует несколько меньшего количества вычислений (например, достаточно только два раза вычислить $f(x, y)$, остальные выполняются как на предыдущих этапах), но требует дополнительного «расхода» памяти. Кроме этого невозможно «запустить метод»: для этого необходимо предварительно получить одношаговыми методами первые три точки.

7.5. Решение ДУ в MathCAD

Решение встроенными методами приведено в [2, с. 83].

Решение обыкновенных дифференциальных уравнений рассматривается в практической работе № 11 [2, с. 81].

Контрольные вопросы

1. Поясните постановку задачи численного решения обыкновенных дифференциальных уравнений.
2. Что является решением дифференциального уравнения?
3. Одношаговые и многошаговые методы численного решения обыкновенных дифференциальных уравнений.
4. Формулы численного решения обыкновенных дифференциальных уравнений методом Эйлера и усовершенствованным методом Эйлера.
5. Правило Рунге оценки погрешностей численного решения.
6. Можно ли методом Эйлера решать системы дифференциальных уравнений?
7. Формулы численного решения обыкновенных дифференциальных уравнений методом Рунге-Кутта.
8. В чем сущность многошаговых методов?
9. За сколько этапов реализуется метод Милна? Что выполняется на каждом этапе?
10. Решение дифференциальных уравнений в MathCAD.

8. ЧИСЛЕННОЕ ДИФФЕРЕНЦИРОВАНИЕ

Численное дифференцирование в отличие от численного интегрирования – не столь актуальная задача в связи с отсутствием принципиальных трудностей аналитического нахождения производных. Зачастую возникает необходимость решения задач численного дифференцирования, например, когда:

1. Аналитический вид функции $f(x)$ может быть неизвестен.

2. Функция $f(x)$ при аналитическом дифференцировании может сильно усложняться.

3. Желательно получить значения производных с помощью однотипных вычислительных процессов без аналитических выкладок.

4. Есть потребность в простых формулах, с помощью которых производные в заданных точках могут аппроксимировать несколькими значениями функции (быть может, неизвестной) в этих близких к ним точках.

Источником формул численного дифференцирования, как и большинства квадратурных формул, является полиномиальная интерполяция.

В точках $x_i = x_0 + ih \quad (i = 0,1,...,n)$, зная $y_i = f(x_i)$, можно найти конечные разности $\Delta^k y_i$ и записать интерполяционный многочлен Ньютона $P_n(x)$, который продифференцировать. Таким образом получают ***конечно-разностную формулу численного дифференцирования***

$$f(x) \approx P_n(x) = P_n(x_0 + qh) = y_0 + q\Delta y_0 + \frac{q(q-1)}{2!}\Delta^2 y_0 + ...$$

$$+ \frac{q(q-1)...(q-n+1)}{n!}\Delta^n y;$$

$$f'(x) \approx q'_x\left[P_n(x_0 + qh)\right]'_q = \frac{1}{h}\left(\Delta y_0 + \frac{2q-1}{2}\Delta^2 y_0 + \frac{3q^2 - 6q + 2}{6}\Delta^3 y_0 + ...\right).$$

Можно вывести и другие формулы численного дифференцирования на основе других интерполяционных формул.

При ограничении степени интерполяционного многочлена можно получить следующие формулы численного дифференцирования:

1) на основе линейной интерполяции, $n = 1$:

$$f'(x) \approx \frac{\Delta y_0}{h} \quad \text{для} \quad x \in (x_0 - \delta, x_1 + \delta);$$

2) на основе квадратичной интерполяции, $n = 2$:

$$f'(x) \approx \frac{1}{h}\left(\Delta y_0 + \frac{2q-1}{2}\Delta^2 y_0\right) \text{ для } x \in (x_0 - \delta, x_2 + \delta),$$

где $\delta > 0$ определяет промежуток экстраполяции приемлемого качества.

Раскрывая конечные разности, получим:

1) для $n = 1$:

$$f'(x_0) \approx y'_0 = \frac{y_1 - y_0}{h},$$

$$f'(x_1) \approx y'_1 = \frac{y_2 - y_1}{h} \text{ и т.д.;}$$

для произвольной точки x $f'(x) \approx \frac{f(x+h) - f(x)}{h}$;

для таблично заданной функции $f'(x_i) \approx \frac{f(x_{i+1}) - f(x_i)}{h}$;

2) для $n = 2$

$$f'(x_0) \approx y'_0 = \frac{1}{2h}(-3y_0 + 4y_1 - y_2),$$

$$f'(x_1) \approx y'_1 = \frac{1}{2h}(y_2 - y_0),$$

$$f'(x_2) \approx y'_2 = \frac{1}{2h}(y_0 - 4y_1 + 3y_2).$$

Повторное дифференцирование полученных формул даст выражения для производных более высоких порядков: $f''(x)$, $f'''(x)$ и т.д.

Задача численного дифференцирования некорректна, так как погрешность производной полинома может существенно превышать погрешность интерполяции.

Остаточные члены приближенных формул находятся с помощью формулы Тейлора и дифференцированием соответствующих остаточных членов интерполяционных формул.

Численное дифференцирование в MathCAD

MathCAD вычисляет производные скалярных функций любого количества аргументов от нулевого до пятого порядка. Функции и аргументы могут быть действительными и комплексными. Производная с хорошей

точностью вычисляется численно и успешно символьно для подавляющего большинства аналитически заданных функций.

Пример 8.1. Численное и символьное дифференцирование средствами MathCAD показано на рис. 8.1.

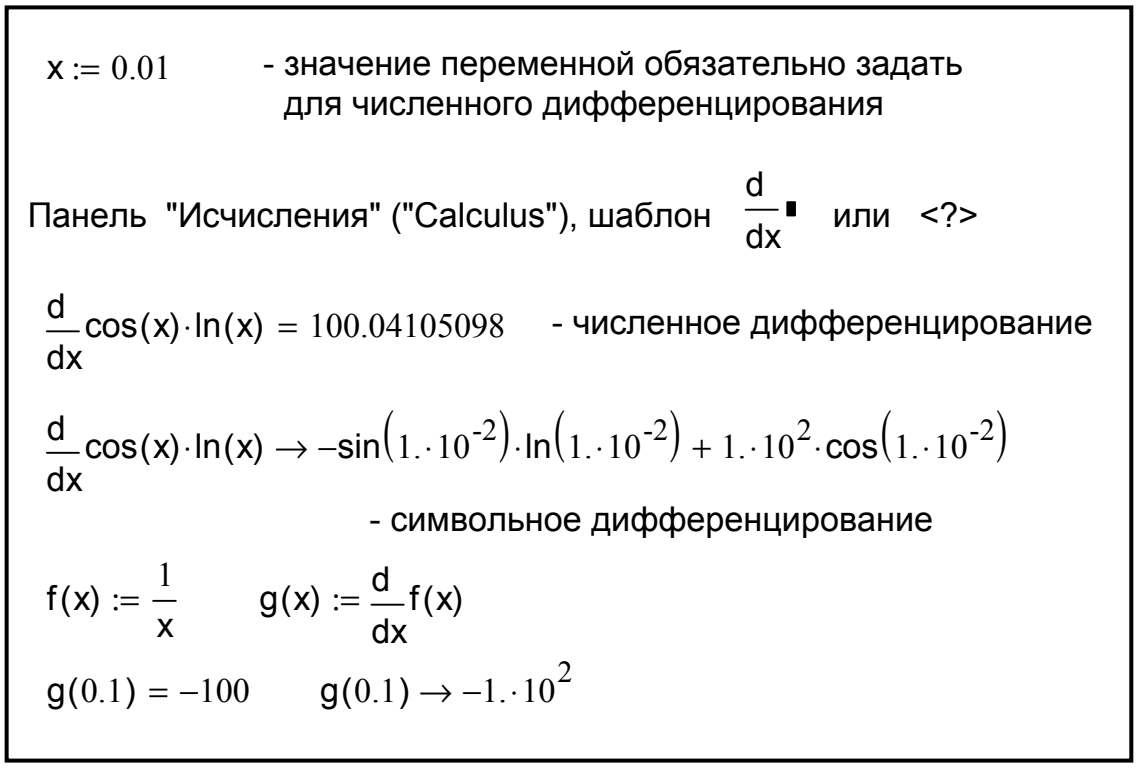


Рис. 8.1. Дифференцирование в MathCAD

Для численного дифференцирования MathCAD применяет довольно сложный алгоритм (метод Риддера), вычисляющий производную с точностью до седьмого-восьмого знака после запятой. Погрешность не зависит от констант *TOL* или *CTOL*, а определяется непоследовательно алгоритмом. При дифференцировании в окрестности сингулярной точки выдается сообщение об ошибке (для $f(x)=\frac{1}{x}$ это точки вблизи $x = 0$).

Для нахождения производных высших порядков применяется тот же метод Риддера. Точность понижается на один разряд при повышении порядка производной на единицу.

Пример 8.2. Вычисление производных высших порядков средствами MathCAD показано на рис. 8.2.

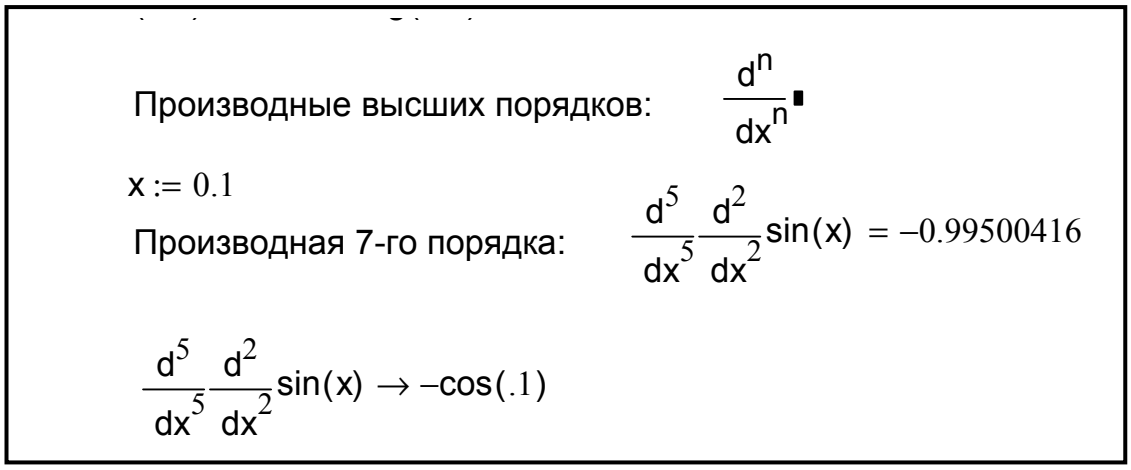


Рис. 8.2. Вычисление производных высших порядков

Контрольные вопросы

1. В каких случаях применяется численное дифференцирование?
2. Как получают формулы численного дифференцирования?
3. Частные случаи формул численного дифференцирования при ограничении степени интерполяционного многочлена.
4. Численное и аналитическое дифференцирование в MathCAD.

ЗАКЛЮЧЕНИЕ

В настоящем пособии рассмотрены основные методы вычислительной математики для решения линейных и нелинейных уравнений и систем; интерполяции и аппроксимации функций; численного интегрирования и дифференцирования; решения обыкновенных дифференциальных уравнений. Рассмотрены основы теории погрешностей и вопросы точности методов. В пособие не вошло большое количество примеров, опубликованных ранее в работе того же автора [2], а также представленных в электронном комплексе, выдаваемом студентам на практических занятиях.

Используемый в пособии учебный материал подобран с учетом того, что дисциплина «Вычислительная математика» изучается на втором курсе.

Полученные знания будут использованы студентами в изучаемых в дальнейшем дисциплинах, таких как «Моделирование», «Методы оптимизации», «Электроника» при обработке экспериментальных данных в исследовательских и выпускных работах.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Вержбицкий, В. М. Основы численных методов: учебник для вузов / В. М. Вержбицкий. – М .: Высш. шк., 2002. – 840 с. – ISBN 5-06-004020-8.

2. Вычислительная математика: метод. указания к практ. занятиям / сост. С.Ю. Кириллова; Владим. гос. ун-т. – Владимир, 2004. – 90 с.

3. Поршнев, С. В. Вычислительная математика: курс лекций / С. В. Поршнев. – СПб.: БХВ-Петербург, 2004. – 320 с. – ISBN 5-94157-400-2.

4. Васильков, Ю. В. Компьютерные технологии вычислений в математическом моделировании: учеб. пособие / Ю. В. Васильков, Н. Н. Василькова. – М.: Финансы и статистика, 2002. – 256 с. – ISBN 5-279-02098-2.

5. Кирьянов, Д. В. Самоучитель MathCAD 2001 / Д. В. Кирьянов. – СПб.: БХВ-Петербург, 2001. – 544 с. – ISBN 5-94157-062-7.

6. Бахвалов, Н. С. Численные методы: учеб. пособие для вузов / Н.С. Бахвалов. [и др.]. – М.; СПб.: Физматлит, 2001. – 632 с.

7. Бахвалов, Н. С. Численные методы в задачах и упражнениях: Учеб. пособие для вузов / Н.С. Бахвалов [и др.]. – М.: Высш. шк., 2000. – 190 с. – ISBN 5-06-003684-7.

8. Костомаров, Д. П. Программирование и численные методы: учеб. пособие для вузов / Д.П. Костомаров [и др.]. – М.: Изд-во МГУ, 2001. – 223 с. – ISBN 5-211-04059-7.

9. Потемкин, В. Г. Система MATLAB 5 для студентов / В. Г. Потемкин. – М.: ДИАЛОГ-МИФИ, 1998. – 314 с. – ISBN 5-86404-114-9.

10. Мэтьюз Г. Джон. Численные методы. Использование MATLAB: пер. с англ. / Джон Г. Мэтьюз, Куртис Д. Финк. – М.: 2001. – 713 с. – ISBN 5-8459-0162-6.

11. Методические указания к лабораторным работам по дисциплине «Информатика», раздел «Численные методы»: В 2 ч. Ч. 1 / сост.: А. П. Алоджанц [и др.]; под ред. Л.М. Вороновой; Владим. гос. ун-т; – Владимир, 1997. – 40 с.

12. Методические указания к лабораторным работам по дисциплине «Информатика», раздел «Численные методы»: В 2 ч. Ч 2 / сост.: Е.В. Евлюхина [и др.]; под ред А.Б. Евлюхина; Владим. гос. ун-т. – Владимир, 1997. – 24 с.

13. Введение в численные методы: метод. указания к курсовому проектированию / сост. Н.А. Новикова; Владим. гос. ун-т. – Владимир, 1998. – 40 с.

ОГЛАВЛЕНИЕ

Учебное издание

КИРИЛЛОВА Светлана Юрьевна

ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА

Учебное пособие

Подписано в печать 15.06.09.
Формат 60x84/16. Усл. печ. л. 6,04. Тираж 100 экз.
Заказ
Издательство
Владимирского государственного университета.
600000, Владимир, ул. Горького, 87.