



ЧИСЛЕННЫЕ МЕТОДЫ И МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

**Учебно-методический комплекс
дисциплины для студентов АмГУ по
направлению подготовки
дипломированных специалистов по
специальности 010701 – «Физика»**

Благовещенск 2007

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное агентство по образованию

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ГОУВПО «АМГУ»)

ЧИСЛЕННЫЕ МЕТОДЫ И
МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

Учебно-методический комплекс дисциплины для студентов
АмГУ по направлению подготовки дипломированных
специалистов по специальности 010701 – «Физика»

<p>Утвержден на заседании кафедры математического анализа и моделирования факультета математики и информатики «__» _____ 2007г., (протокол № от _____) Зав. кафедрой Т.В. Труфанова</p>	<p>Утвержден на заседании УМС специальности 010701 – «Физика» «__» _____ 2007г., (протокол № от _____) Председатель УМС факультета В.Ф. Ульянычева</p>
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------

ББК
Т 80

*Печатается по решению
редакционно-издательского совета
факультета математики и
информатики
Амурского государственного
университета*

Масловская А.Г.

Численные методы и математическое моделирование. Учебно-методический комплекс дисциплины для студентов АмГУ по направлению подготовки дипломированных специалистов по специальности 010701 – «Физика» – Благовещенск: Амурский гос. ун-т, 2007 – 88с.

Учебно-методический комплекс по дисциплине содержит рабочую программу дисциплины, краткий курс лекций, варианты индивидуальных заданий к лабораторному практикуму на ЭВМ, а также контролирующие материалы для осуществления контроля усвоения знаний учащимися.

СОДЕРЖАНИЕ

1	Цели и задачи дисциплины, ее место в учебном процессе	3
1.1	Цели и задачи курса	3
1.2	Требования к уровню содержания дисциплины	3
1.3	Перечень дисциплин с указанием разделов (тем), усвоение которых необходимо при изучении данной дисциплины	4
2	Содержание дисциплины	4
2.1	Федеральный компонент	4
2.2	Наименование тем, их содержание, объем в лекционных часах	5
2.3	Лабораторные занятия, их содержание, и объем в часах	8
2.4	Самостоятельная работа студентов	10
2.5	Вопросы к зачету	11
2.6	Виды контроля	13
2.7	Требования к знаниям студентов, предъявляемые на экзамене	13
3	Краткий курс лекций	13
	Решение нелинейных уравнений	13
	Решение систем линейных алгебраических уравнений	22
	Решение систем нелинейных уравнений	30
	Интерполирование функций	36
	Обработка экспериментальных данных	39
	Численное дифференцирование и интегрирование	44
	Численные методы решения обыкновенных дифференциальных уравнений	51
4	Учебно-методические материалы по дисциплине	59
4.1	Лабораторный практикум	59
	Лабораторная работа №1	59
	Лабораторная работа №2	72
	Лабораторная работа №3	73
	Лабораторная работа №4	76

Лабораторная работа №5	77
Лабораторная работа №6	79
Лабораторная работа №7	81
Лабораторная работа №8	82
4.2 Перечень обязательной литературы	84
4.3 Перечень дополнительной литературы	85
4.4 Перечень методических пособий	86
5 Необходимое техническое и программное обеспечение	86
6 Учебно-методическая карта дисциплины	86

1 ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ

1.1 Цели и задачи курса

Численные методы занимают важное место в системе прикладного математического образования. Целью преподавания дисциплины является изучение численных методов решения задач алгебры, математического анализа и дифференциальных уравнений, а также освоение методов построения, классификации и анализа математических моделей.

Задачи изучения курса составляют следующие вопросы: численные методы построения, решения и исследования различных задач, разработка и выбор оптимального алгоритма решения конкретных задач, обработка и анализ полученных результатов, корректировка способа решения при наличии особенностей задачи, анализ вопроса устойчивости и сходимости метода решения, оценка границ применимости построенной математической модели.

1.2 Требования к уровню освоения содержания дисциплины

В результате освоения дисциплины студенты должны иметь четкое представление о видах математических моделей, основанных на численных методах, о способах их построений, методах моделирования, возможностях программной реализации, разрабатывать алгоритм реализации метода решения, анализировать полученные результаты, оценивать погрешность вычислений. В процессе обучения студенты должны приобрести навыки применения численных методов для решения задач, самостоятельно осуществлять выбор методики решения и построения алгоритма той или иной задачи, давать полный анализ результатов решения и оценивать границы применимости выбранного метода.

1.3 Перечень дисциплин с указанием разделов (тем), усвоение которых студентами необходимо при изучении данной дисциплины

Данный курс базируется на ранее изученных дисциплинах: «Математический анализ», «Аналитическая геометрия и линейная алгебра», связан с дисциплинами: «Дифференциальные уравнения», «Уравнения математической физики» и дает основу для реализации изученных методов в виде программных алгоритмов на занятиях по курсу: «Вычислительная физика».

2 СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

2.1 Федеральный компонент

Дисциплина «Численные методы и математическое моделирование» является дисциплиной, входящей в блок дисциплин информатики федерального компонента для специальности 010701 «Физика». Государственный стандарт – ЕН.Ф.04.03.

Приближенные числа, погрешности. Вычисление значений простейших функций. Интерполяция и приближение функций. Интерполяционные полиномы. Наилучшее приближение. Среднеквадратичное приближение. Равномерное приближение. Ортогональные многочлены. Сплайн интерполяция. Поиск корней нелинейных уравнений. Итерационные методы. Метод Ньютона. Отделение корней. Комплексные корни. Решение систем уравнений. Вычислительные методы линейной алгебры. Прямые и итерационные процессы. Задачи на собственные значения. Численное дифференцирование. Численное интегрирование. Численное интегрирование быстро осциллирующих функций. Многомерные интегралы. Метод Монте-Карло. Задача Коши для обыкновенных дифференциальных уравнений. Интегрирование уравнений второго и высшего порядков. Численные методы решения краевой задачи и задачи на собственные значения для обыкновенных дифференциальных уравнений. Вычислительные методы решения краевых задач математической физики. Разностные схемы. Аппроксимация. Устойчивость. Сходимость. Вариационно-разностные методы, метод конечных элементов. Численные методы решения

интегральных уравнений. Поиск экстремума, одномерная и многомерная оптимизация. Методы математического программирования. Вычисление псевдообратных матриц и псевдообратных решений. Сингулярное разложение. Обработка экспериментальных данных.

2.2 Наименование тем, их содержание, объем в лекционных часах

ТЕМАТИЧЕСКИЙ ПЛАН ЛЕКЦИОННЫХ ЗАНЯТИЙ

Наименование темы	Кол-во часов
1. Введение.	2
2. Точность вычислительного эксперимента.	2
3. Численные методы решения нелинейных алгебраических уравнений.	2
4. Численные методы линейной алгебры.	4
5. Численное решение систем нелинейных уравнений.	2
6. Аппроксимация функций	6
7. Численное дифференцирование	2
8. Численное интегрирование	4
9. Обработка экспериментальных данных.	2
10. Приближенное решение обыкновенных дифференциальных уравнений.	2
11. Краевые задачи для обыкновенных дифференциальных уравнений	2
12. Численное решение уравнений с частными производными.	6
ИТОГО	36

Тема 1. Введение.

Предмет вычислительной математики. Методы вычислительной математики. Численные методы как раздел вычислительной математики. Общие сведения о моделировании. Применение численных методов в математическом моделировании. Классификация математических моделей и основные этапы моделирования.

Тема 2. Точность вычислительного эксперимента.

Правила приближенных вычислений и элементы теории погрешностей. Приближенные числа, абсолютные и относительные погрешности. Арифметические действия над приближенными числами. Устойчивость. Корректность. Сходимость.

Тема 3. Численные методы решения нелинейных алгебраических уравнений.

Метод половинного деления. Метод хорд. Метод Ньютона. Метод простых итераций. Метод релаксаций. Метод Чебышева третьего порядка. Геометрическая интерпретация рассмотренных методов.

Тема 4. Численные методы линейной алгебры.

Численное решение систем линейных алгебраических уравнений. Основные понятия. Прямые и итерационные методы. Метод Гаусса. Схема Гаусса с выбором главного элемента. Схема Халецкого. Метод простой итерации. Метод Зейделя. Вычисление определителей. Вычисление элементов обратной матрицы методом Гаусса. Задачи на собственные значения. Метод Крылова для нахождения собственных чисел и векторов матриц.

Тема 5. Численное решение систем нелинейных уравнений.

Метод Ньютона. Метод простой итерации. Метод скорейшего спуска. Варианты итерационных схем.

Тема 6. Аппроксимация функций.

Понятие о приближении функции. Использование рядов. Многочлены Чебышева и наилучшие равномерные приближения. Интерполирование функций. Линейная и квадратичная интерполяции. Интерполяционные сплайны. Полиномиальная интерполяция. Интерполяционные формулы Ньютона для равноотстоящих узлов. Интерполяционный многочлен Лагранжа. Схема Эйткена. Интерполяционные формулы Гаусса, Стирлинга, Бесселя. Обратное интерполирование. Нахождение корней уравнения методом обратного интерполирования.

Тема 7. Численное дифференцирование.

Аппроксимация производных. Погрешности, возникающие при численном дифференцировании. Выбор оптимального шага. Использование интерполяционных формул. Частные производные.

Тема 8. Численное интегрирование.

Квадратурные формулы. Выбор шага интегрирования. Интегрирование с помощью степенных рядов. Интегралы от разрывных функций. Метод Гаусса. Интегралы с бесконечными пределами. Кратные интегралы. Метод повторного интегрирования. Метод Диткина. Метод Монте-Карло.

Тема 9. Обработка экспериментальных данных.

Подбор эмпирических формул. Эмпирические формулы. Определение параметров эмпирической зависимости. Метод наименьших квадратов. Локальное сглаживание данных. Нахождение приближающей функции в виде линейной функции и квадратичного трехчлена. Аппроксимация функцией произвольного вида.

Тема 10. Приближенное решение обыкновенных дифференциальных уравнений.

Основные понятия и методы решения. Задача Коши. Одношаговые методы. Метод последовательных приближений. Метод Эйлера. Модификации метода Эйлера. Метод Рунге-Кутты. Многошаговые методы. Метод Адамса. Метод Милна. Аппроксимация, устойчивость, сходимость численного решения задач для дифференциального уравнения.

Тема 11. Краевые задачи для обыкновенных дифференциальных уравнений.

Постановка задачи. Метод конечных разностей для линейных и нелинейных дифференциальных уравнений второго порядка. Метод прогонки. Метод Галеркина. Метод коллокации.

Тема 12. Численное решение уравнений с частными производными.

Метод сеток. Итерационный метод решения системы конечно-разностных уравнений. Решение краевых задач для криволинейных областей. Метод сеток для уравнений параболического и гиперболического типа. Интегральные уравнения. Виды интегральных уравнений. Методы решения. Решение уравнений Фредгольма и уравнения Вольтера второго рода методом конечных сумм

2.3 Лабораторные занятия, их содержание и объем в часах.

ТЕМАТИЧЕСКИЙ ПЛАН ЛАБОРАТОРНЫХ ЗАНЯТИЙ

Наименование темы	Кол-во часов
1. Основные приемы работы с пакетом Matlab.	2
2. Численные методы решения нелинейных алгебраических уравнений.	2

3. Численное решение систем линейных уравнений.	2
4. Численное решение систем нелинейных уравнений.	2
5. Интерполирование функций	2
6. Численное дифференцирование и интегрирование	2
7. Обработка экспериментальных данных.	2
8. Приближенное решение обыкновенных дифференциальных уравнений.	2
9. Численное решение уравнений с частными производными.	2
ИТОГО	18

При выполнении лабораторных работ по данному курсу студенты должны продемонстрировать умение решать прикладные задачи, как с использованием возможностей математических прикладных программ, так и создавая собственные алгоритмы разработанных математических моделей. При прохождении практикума на ЭВМ рекомендован ППП Matlab 6.0.

Лабораторная работа выполняется строго в соответствии с выданным преподавателем заданием и вариантом. Завершающим этапом выполнения лабораторной работы является оформление отчета. Отчет содержит: титульный лист, лист задания, раздел, содержащий теоретические основы соответствующего раздела курса, включая расчетные формулы основного метода и расчет погрешности метода, раздел, содержащий описание программной реализации: листинг программного блока (описание интерфейса программы можно вынести в приложение), раздел, содержащий описание результатов, полученных с использованием возможностей ППП, список использованной литературы.

2.4 Самостоятельная работа студентов (28 часов).

В качестве самостоятельной работы по дисциплине «Численные методы и математическое моделирование» студентам предлагается изучить дополнительный раздел и рассмотреть вопросы:

Вычисление значений функции. Вычисление значений многочлена. Схема Горнера. Вычисление значений трансцендентных функций с помощью степенных рядов. Применение метода итераций для приближенного вычисления значений функций.

Для промежуточного контроля приобретенных практических и вычислительных навыков предусмотрены индивидуальные задания для типового расчета (по вариантам) по следующим темам: «Приближенные вычисления и оценка погрешностей», «Нахождение собственных чисел и собственных векторов матриц по методу Крылова», «Приближение функций».

2.5 Вопросы к зачету

1. Приближенные числа, их абсолютные и относительные погрешности. Верные знаки числа.
2. Арифметические действия над приближенными числами.
3. Устойчивость. Корректность. Сходимость.
4. Численное решение систем линейных алгебраических уравнений. Прямые методы. Метод Гаусса. Схема Гаусса с выбором главного элемента.
5. Численное решение систем линейных алгебраических уравнений. Итерационные методы. Метод простой итерации. Метод Зейделя. Схема Халецкого.
6. Численные методы линейной алгебры. Вычисление определителей. Вычисление элементов обратной матрицы методом Гаусса.
7. Численные методы линейной алгебры. Задачи на собственные значения. Метод Крылова для нахождения собственных чисел и векторов матриц.

8. Численные методы решения нелинейных алгебраических уравнений. Метод половинного деления. Метод хорд. Геометрическая интерпретация рассмотренных методов.
9. Численные методы решения нелинейных алгебраических уравнений. Метод Ньютона. Метод простых итераций. Метод релаксаций. Геометрическая интерпретация рассмотренных методов.
10. Численное решение систем нелинейных уравнений. Метод Ньютона. Метод простой итерации.
11. Аппроксимация функций. Понятие о приближении функций. Точечная аппроксимация. Равномерное приближение.
12. Аппроксимация функций. Использование рядов. Многочлены Чебышева. Вычисление многочленов. Рациональные приближения.
13. Аппроксимация функций. Интерполирование функций. Линейная и квадратичная интерполяции. Интерполяционные сплайны.
14. Аппроксимация функций. Интерполирование функций. Полиномиальная интерполяция. Интерполяционные формулы Ньютона для равноотстоящих узлов.
15. Аппроксимация функций. Интерполирование функций. Интерполяционный многочлен Лагранжа. Схема Эйткена.
16. Аппроксимация функций. Интерполирование функций. Интерполяционные формулы Гаусса, Стирлинга, Бесселя.
17. Аппроксимация функций. Интерполирование функций. Обратное интерполирование. Нахождение корней уравнения методом обратного интерполирования.
18. Обработка экспериментальных данных. Метод наименьших квадратов. Локальное сглаживание данных.
19. Численное дифференцирование. Аппроксимация производных. Погрешности. Выбор оптимального шага. Использование интерполяционных формул. Частные производные.

20. Численное интегрирование. Квадратурные формулы. Выбор шага интегрирования. Интегрирование с помощью степенных рядов.
21. Численное интегрирование. Интегралы от разрывных функций. Интегралы с бесконечными пределами.
22. Численное интегрирование. Кратные интегралы. Метод повторного интегрирования. Метод Диткина. Метод Монте-Карло.
23. Приближенное решение обыкновенных дифференциальных уравнений. Задача Коши. Одношаговые методы. Метод последовательных приближений. Метод Эйлера. Модификации метода Эйлера. Метод Рунге-Кутты.
24. Приближенное решение обыкновенных дифференциальных уравнений. Многошаговые методы. Метод Адамса. Метод Милна.
25. Приближенное решение обыкновенных дифференциальных уравнений. Аппроксимация, устойчивость, сходимость численного решения задач для дифференциального уравнения.
26. Краевые задачи для обыкновенных дифференциальных уравнений. Метод конечных разностей для линейных и нелинейных дифференциальных уравнений второго порядка.
27. Краевые задачи для обыкновенных дифференциальных уравнений. Метод прогонки. Метод Галеркина. Метод коллокации.
28. Численное решение уравнений с частными производными. Метод сеток. Итерационный метод решения системы конечно-разностных уравнений.
29. Численное решение уравнений с частными производными. Решение краевых задач для криволинейных областей. Метод сеток для уравнений параболического и гиперболического типа.
30. Интегральные уравнения. Виды интегральных уравнений. Методы решения. Решение уравнений Фредгольма и уравнения Вольтера второго рода методом конечных сумм.

2.6 Виды контроля

Текущий контроль за аудиторной и самостоятельной работой обучаемых осуществляется во время проведения лабораторных занятий посредством устного опроса по контрольным вопросам соответствующего раздела, а также проверки отчетов по лабораторным работам и индивидуальным заданиям. Промежуточный контроль осуществляется два раза в семестр в виде анализа итоговых отчетов на аттестационные вопросы. Итоговый контроль осуществляется после успешного прохождения студентами текущего и промежуточного контроля в виде зачета.

2.7 Требования к знаниям студентов, предъявляемые на зачете

Зачет сдается в конце семестра. Форма сдачи зачета – устная. Необходимым условием допуска на зачет является сдача всех лабораторных работ и трех расчетных работ. В предлагаемый билет входят два вопроса: основной и дополнительный. Студент должен дать развернутый ответ на основной вопрос, и краткий – на дополнительный. Развернутый ответ предполагает полное знание теории по данной части курса, свободную ориентацию в материале, краткий ответ – основных теоретических моментов: понятий и терминологии. При выполнении указанных требований ставится отметка «зачтено».

3 КРАТКИЙ КУРС ЛЕКЦИЙ РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

Рассмотрим задачу приближенного нахождения корней уравнения вида:

$$f(x) = 0, \tag{1}$$

где $f : R_1 \mapsto R_1$ – алгебраическая или трансцендентная функция.

Аналитические методы решений скалярных уравнений (1) существуют только в отдельных классах уравнений. В общем случае можно говорить лишь о приближенном вычислении корней уравнения (1), т.е. таких значений

аргумента $x = \xi$, при которых равенство $f(\xi) = 0$ истинно. При этом под близостью приближенного значения \bar{x} к корню ξ уравнения (1) понимают выполнение неравенства:

$$|\xi - \bar{x}| < \varepsilon. \quad (2)$$

При малых $\varepsilon > 0$ часто важно контролировать не абсолютную погрешность, а относительную: $\frac{|\xi - \bar{x}|}{|\bar{x}|}$.

Локализация корней

Нелинейная функция $f(x)$ в своей области определения $D(f) \subseteq R_1$ может иметь конечное или бесконечное количество нулей или не иметь их вообще. Большинство же методов нахождения корней требует знания промежутков, где имеется и притом единственный нуль функции. Таким образом, ставится подзадача существования и единственности, нахождения границ и локализации (изоляции, отделения) корней. Для функций общего вида нет универсальных способов решения поставленных подзадач. На практике используют следующие подходы.

1. *Графический способ локализации корней.* Если по геометрической интерпретации функции $f(x)$ можно однозначно представить ситуацию, касающуюся количества и расположения нулей, то локализацию производят, отделяя те промежутки на оси абсцисс, где график функции $y = f(x)$ пересекает ось Ox .

Если построение графика функции $y = f(x)$ вызывает затруднения, но исходное уравнение (1) очевидным образом представляется в виде $f_1(x) = f_2(x)$ и функции $f_1(x)$ и $f_2(x)$ таковы, что построение их графиков возможно, задача определения корней и областей их единственности решается отслеживанием точек пересечения этих графиков и выделением на оси абсцисс тех промежутков, которым принадлежат проекции данных точек.

Пример 1. Найти промежутки локализации корней уравнения $x^2 - \sin x - 1 = 0$.

Представив уравнение в виде $x^2 - 1 = \sin x$, строим графики функций

2. *Аналитический способ отделения корней.* Убедиться в том, что на данном отрезке $[a, b]$ (определенным, например, «грубым» графическим способом) имеется нуль непрерывной функции $f(x)$ можно, используя теорему Больцано-Коши, наложив дополнительное требование монотонности функции на этом отрезке.

Утверждение 1. Непрерывная строго монотонная функция $f(x)$ имеет и притом единственный нуль на отрезке $[a, b]$ тогда и только тогда, когда на его концах она принимает значения разных знаков.

Реально установить монотонность на данном отрезке можно для дифференцируемой функции, потребовав знакопостоянства ее производной на всем отрезке.

Утверждение 2. Пусть $f \in C^1_{[a,b]}$. Тогда если $f'(x)$ не меняет знак на (a, b) , то условие $f(a)f(b) < 0$ является необходимым и достаточным для того, чтобы уравнение (1) имело и притом единственный корень на $[a, b]$.

Пример 2. Локализовать корни уравнения $x^2 e^x = \pi$.

Пусть $f(x) = x^2 e^x - \pi$, тогда $f'(x) = 2xe^x + x^2 e^x = x(x+2)e^x$. $f'(x) = 0$ только при $x = 0$ и $x = -2$. Исследуем знаки значений производной и самой функции:

	$(-\infty, -2)$	-2	$(-2, 0)$	0	$(0, +\infty)$
$f'(x)$	+	0	-	0	+
$f(x)$	-	$\frac{4}{e^2} - \pi$	-	$-\pi$	+

На основании утверждения 2 можно заключить, что данное уравнение имеет единственный корень и этот корень положителен. Установив знаки функции $f(x)$ в дополнительных точках $x = 1$ (минус) и $x = 2$ (плюс), область поиска решения $[0, +\infty]$ сужаем до промежутка конечной длины $[1, 2]$.

В общем случае, если указанные процедуры затруднительны для анализа, всю область определения или какую-нибудь ее часть, вызывающую

по тем или иным соображениям интерес, разбивают на отрезки точками x_i , расположенными на условно небольшом расстоянии h одна от другой. Вычислив значения во всех этих точках, проверяют выполнение условия $f(x_{i-1})f(x_i) \leq 0$. Если заранее известно число корней в исследуемой области, то, измельчая шаг h поиска, таким процессом можно либо их все локализовать, либо утверждать, что возможно наличие пар корней, не различимых с точностью $h = \varepsilon$.

Методы дихотомии. Метод половинного деления

Пусть функция $f(x)$ определена и непрерывна при всех $x \in [a, b]$ и на $[a, b]$ меняет знак. Согласно утверждению 2 уравнение (1) на (a, b) имеет единственный корень.

Выберем произвольную точку $c \in (a, b)$, $[a, b]$ – промежуток существования корня, c – пробная точка. Вычисление значения $f(c)$ приведет к какой-либо одной из следующих взаимоисключающих ситуаций:

$$1) f(a)f(c) < 0, \quad 2) f(c)f(b) < 0, \quad 3) f(c) = 0,$$

которые можно интерпретировать следующим образом:

- 1) корень находится на интервале (a, c) ,
- 2) корень находится на интервале (c, b) ,
- 3) точка c – искомый корень.

Таким образом, одно вычисление значения функции позволяет уменьшить промежуток $[a, b]$ существования корня. Описанная процедура сужения промежутка существования нуля непрерывной функции может быть применена к промежутку (a, c) (случай 1) или к промежутку (c, b) (случай 2) и далее повторена циклически. Такой легко программируемый процесс объединяет семейство *методов дихотомии* (деление на две части).

Наиболее употребительным частным случаем метода дихотомии является *метод половинного деления*, реализующий самый простой способ

выбора пробной точки – деление промежутка существования корня пополам:

$$c = \frac{a+b}{2}.$$

За один шаг метода половинного деления промежуток существования корня сокращается ровно вдвое. Поэтому если за k -е приближение этим методом к корню ξ уравнения (1) примем точку x_k , являющуюся серединой полученного на k -м шаге отрезка $[a_k, b_k]$ в результате последовательного сужения $[a, b]$, то получим:

$$|\xi - x_k| < \frac{b-a}{2^k}, \quad \forall k \in N. \quad (3)$$

Неравенство (3), являясь априорной оценкой абсолютной погрешности приближенного равенства $x_k \approx \xi$, дает возможность подсчитать число итераций метода половинного деления, достаточное для получения корня с заданной точностью. Для этого достаточно найти наименьшее натуральное k , удовлетворяющее неравенству $\frac{b-a}{2^k} \leq \varepsilon$.

Рис. 1 иллюстрирует алгоритм нахождения корня методом половинного деления.

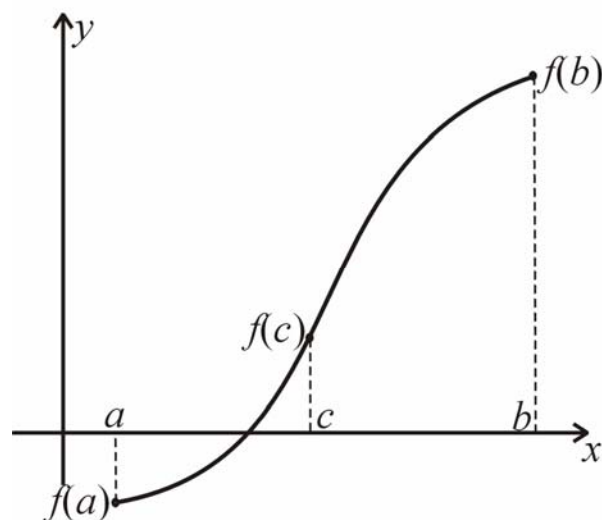


Рис. 1. – Приближение к корню уравнения методом половинного деления.

Метод хорд

В семействе методов дихотомии можно достичь несколько лучших результатов, если отрезок $[a, b]$ делить 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 $B(b, f(b))$ – с хордой AB дуги $A\xi B$. Из уравнения прямой, проходящей через точки A и B , находим:

$$c = a - \frac{f(a)(b - a)}{f(b) - f(a)} \quad (4)$$

Метод, получающийся из метода дихотомии таким фиксированием пробной точки, называют *методом хорд* (методом секущих, методом линейной интерполяции).

Рис. 2 иллюстрирует алгоритм нахождения корня методом хорд.

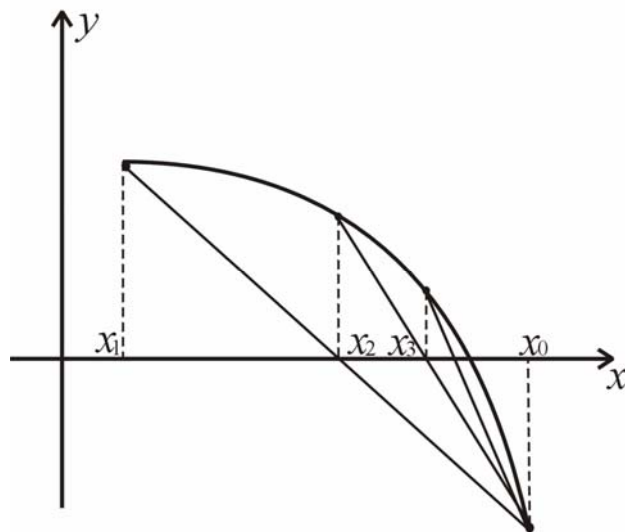


Рис. 2. – Приближение к корню уравнения методом хорд.

Счет обычно ведется до совпадения значений c на двух соседних итерациях с точностью ε или с точностью

$$\frac{m\varepsilon}{M - m}, \text{ если } 0 < m \leq |f'(x)| \leq M, \quad \forall x \in [a, b].$$

Метод Ньютона

Метод Ньютона является одним из популярных итерационных методов решения нелинейных уравнений, что связано с его идейной простотой и быстрой сходимостью.

Пусть функция $f(x)$ дважды дифференцируема на отрезке $[a, b]$, содержащем корень ξ уравнения (1), пусть $x_n \in [a, b]$ – уже известный член последовательности приближений к ξ . Для любого $x \in [a, b]$ можно записать формальное представление $f(x)$ по формуле Тейлора:

$$f(x) = f(x_n) + f'(x_n)(x - x_n) + \frac{1}{2} f''(\theta_n)(x - x_n)^2 + \dots \quad (5)$$

Так как ξ – корень уравнения (1), то разложение (5) справедливо и для $x = \xi$. Считая, что значение x_n достаточно близко к ξ (разность $(x - \xi)^2$ – достаточно мала), ограничимся двумя членами разложения в правой части выражения (5), при этом будет найдено новое приближение x_{n+1} . Предполагая, что, по крайней мере, на элементах последовательности x_n первая производная данной функции в ноль не обращается, итерационный процесс Ньютона определится в явном виде формулой:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 0, 1, 2, \dots \quad (6)$$

В полученном выражении легко усмотреть уравнение касательной к кривой $f(x)$, проведенной в точке $(x_n, f(x_n))$. Отсюда геометрический смысл метода Ньютона: приближения к корню ξ совершаются по абсциссам точек пересечения касательных к графику данной функции, проводимых в точках, соответствующих предыдущим приближениям.

Рис. 3 иллюстрирует алгоритм нахождения корня методом касательных.

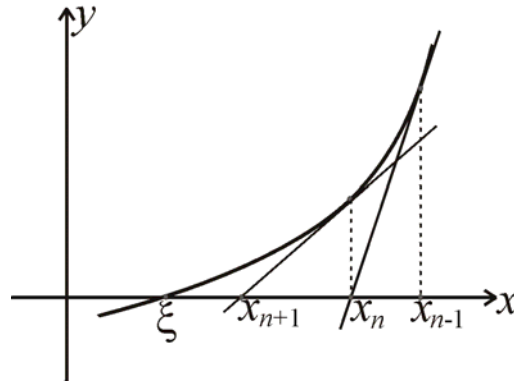


Рис. 3. – Приближение к корню уравнения методом Ньютона.

Для оценки скорости сходимости метода Ньютона можно использовать следующее утверждение.

Утверждение 3. Пусть функция $f(x)$ удовлетворяет условиям

$$\begin{cases} |f'(x)| \geq \alpha > 0, \\ |f''(x)| \leq \beta < \infty, \end{cases} \quad \forall x \in [a, b].$$

Тогда, если члены последовательности (x_n) , определяемые методом Ньютона (6), при любом фиксированном $n \in \mathbb{N}_0$ принадлежат отрезку $[a, b]$ и эта последовательность сходится к корню ξ уравнения (1), то справедливы неравенства:

$$\begin{aligned} |x_{n+1} - \xi| &\leq \frac{\beta}{2 \cdot \alpha} |x_n - \xi|^2, \\ |x_{n+1} - \xi| &\leq \frac{\beta}{2 \cdot \alpha} |x_{n+1} - x_n|^2. \end{aligned}$$

Модификации метода Ньютона

Если заведомо известно число m – показатель кратности корня ξ , то для ускорения сходимости метода Ньютона в формулу (6) рекомендуется ввести корректирующий множитель m :

$$x_{n+1} = x_n - m \frac{f(x_n)}{f'(x_n)}, \quad n = 0, 1, 2, \dots \quad (7)$$

Такую модификацию называют *методом Ньютона-Шредера*.

Цель всех последующих видоизменений формулы (6) – сокращение вычислительных затрат, связанных с вычислением производной на каждом шаге.

Использование на каждом шаге одного и того же шагового множителя $\frac{1}{f'(x_0)}$ дает модифицированный или *упрощенный метод Ньютона*:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_0)}, \quad n = 0, 1, 2, \dots \quad (8)$$

Этот метод имеет очевидную геометрическую интерпретацию: в начальной точке x_0 проводится касательная к графику функции $f(x)$, а во всех последующих точках проводятся прямые, параллельные этой касательной (рис. 4).

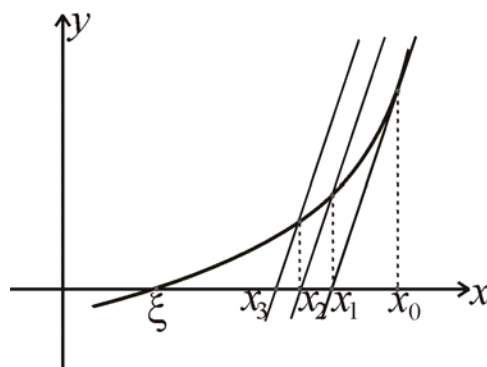


Рис. 4. – Приближение к корню уравнения упрощенным методом Ньютона.

Заменяя производную, вычисленную в точке x_n , ее разностным аналогом получим *разностный метод Ньютона*:

$$x_{n+1} = x_n - \frac{f(x_n)h_n}{f(x_n + h_n) - f(x_n)}, \quad n = 0, 1, 2, \dots, \quad (9)$$

где h_n – конечное приращение аргумента на n -ой итерации.

При выборе $h_n = f(x_n)$ формула (9) преобразуется к виду:

$$x_{n+1} = x_n - \frac{f(x_n)^2}{f(x_n + f(x_n)) - f(x_n)}, \quad n = 0, 1, 2, \dots \quad (10)$$

и называется *методом Стеффенсена*.

Принимая $h_n = x_{n-1} - x_n$ в выражении (9), получим итерационный процесс, который носит название *двухшагового метода Ньютона*:

$$x_{n+1} = x_n - \frac{f(x_n)(x_{n-1} - x_n)}{f(x_{n-1}) - f(x_n)}, \quad n = 1, 2, \dots \quad (11)$$

Метод Чебышева третьего порядка

Пусть производные функции $f(x)$ первого, второго и третьего порядков непрерывны на отрезке $[a, b]$, причем $f'(x) \neq 0$ на $[a, b]$. Пусть начальное приближение $x_0 \in [a, b]$, тогда итерационный процесс *метода Чебышева* осуществляется по формуле:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} - \frac{f''(x_n) \cdot f^2(x_n)}{2 \cdot (f'(x_n))^3}, \quad n = 0, 1, 2, \dots \quad (12)$$

Данный метод обладает скоростью сходимости порядка 3, оценка погрешности имеет вид

$$|\bar{x} - x_n| \leq \left(\alpha \cdot \frac{\beta^3}{3!} \cdot |\bar{x} - x_0| \right)^{\frac{3^n - 1}{2}},$$

где $\alpha = \max_{x \in [a, b]} |F'''(f(x))|$, $\beta = \max_{x \in [a, b]} |f'(x)|$, $F(f(x))$ – функция, обратная $f(x)$

на отрезке $[a, b]$, имеет непрерывные производные первого, второго и третьего порядков.

Численные примеры. Реализация в пакете Matlab

Пример 3. Методом хорд найти корень уравнения $f(x) = x^3 + 8x^2 - 14x - 20 = 0$, расположенного на отрезке $[2, 3]$.

РЕШЕНИЕ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ

Все методы решения систем линейных алгебраических уравнений можно разбить на два класса: прямые и итерационные. *Прямые методы* приводят к решению за конечное число арифметических итераций. Если

Так как реальные машинные вычисления производятся с накоплением ошибки округления, то, анализируя формулы (3), можно заключить, что выполнение алгоритма может прекратиться или привести к неверным результатам, если знаменатели дробей на некотором этапе окажутся равными нулю или очень маленькими числами. Чтобы уменьшить влияние ошибок округления и исключить деление на нуль, на каждом этапе прямого хода уравнения системы обычно представляют так, чтобы деление производилось на наибольший по модулю в данном столбце элемент. Числа, на которые производится деление, называют *ведущими* или *главными элементами*, а сам метод Гаусса – *метод Гаусса с постолбцовым выбором главного элемента*. Устойчивость алгоритма к погрешностям исходных данных и результатов промежуточных вычислений можно еще больше усилить, если выполнять деление на каждом этапе на элемент, наибольший по модулю во всей матрице. Такая модификация метода Гаусса носит название *метода главных элементов*.

Метод прогонки

Часто возникает необходимость в решении линейных алгебраических систем, матрицы которых имеют ленточную структуру – ненулевые элементы располагаются на главной диагонали и на нескольких побочных. Будем искать решение системы вида:

$$b_i x_{i-1} + c_i x_i + d_i x_{i+1} = r_i, \quad i = 1, 2, \dots, n, \quad b_1 = 0, \quad d_n = 0. \quad (5)$$

Система (5) имеет трехдиагональную структуру:

$$\begin{bmatrix} c_1 & d_1 & 0 & 0 & \dots & 0 & 0 & 0 \\ b_2 & c_2 & d_2 & 0 & \dots & 0 & 0 & 0 \\ 0 & b_3 & c_3 & d_3 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & b_{n-1} & c_{n-1} & d_{n-1} \\ 0 & 0 & 0 & 0 & \dots & 0 & b_n & c_n \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \dots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ \dots \\ r_{n-1} \\ r_n \end{bmatrix} \quad (6)$$

Предполагая, что существуют такие наборы чисел δ_i, λ_i , при которых

$$x_i = \delta_i x_{i+1} + \lambda_i, \quad (7)$$

подставляя (7) в (5), получим:

$$x_i = -\frac{d_i}{c_i + b_i \delta_{i-1}} x_{i+1} + \frac{r_i - b_i \lambda_{i-1}}{c_i + b_i \delta_{i-1}}. \quad (8)$$

Последнее имеет место, если при всех $i = 1, 2, \dots, n$ выполняются рекуррентные соотношения:

$$\delta_i = -\frac{d_i}{c_i + b_i \delta_{i-1}}, \quad \lambda_i = \frac{r_i - b_i \lambda_{i-1}}{c_i + b_i \delta_{i-1}}. \quad (9)$$

Процесс вычисления в силу условия $b_1 = 0$ может быть начат со значений: $\delta_1 = -\frac{d_1}{c_1}$, $\lambda_1 = \frac{r_1}{c_1}$ и продолжен далее по формулам (9).

При $i = n$ в силу $d_n = 0$ получим $\delta_n = 0$ и

$$x_n = \lambda_n = \frac{r_n - b_n \lambda_{n-1}}{c_n + b_n \delta_{n-1}}, \quad (10)$$

где $\lambda_{n-1}, \delta_{n-1}$ – уже известные с предыдущего шага числа. Далее по формулам (7) последовательно находятся $x_{n-1}, x_{n-2}, \dots, x_1$, $i = n-1, n-2, \dots, 1$.

Описанное решение уравнений вида (5) носит название *метода прогонки*, который состоит из двух этапов: вычисление *прогоночных коэффициентов* по формулам (9) – *прямая прогонка* и вычисление неизвестных по формулам (10), (7) – *обратная прогонка*.

Для успешного применения метода прогонки требуется, чтобы в процессе вычислений не возникало ситуаций с делением на ноль и накоплением ошибки округления.

Прогонку называют *корректной*, если знаменатель прогоночных коэффициентов (9) не обращается в ноль, и *устойчивой*, если

$$|\delta_i| < 1, \quad i \in \{1, 2, \dots, n\}.$$

или:

$$X = \Phi(X) \tag{2'}$$

Рекуррентное равенство:

$$X^{(k+1)} = \Phi(X^{(k)}), \quad k = 0, 1, 2, \dots, \tag{3}$$

определяет *метод простых итераций* для задачи (1) – (1').

Если начать построение последовательности $X^{(k)}$ с некоторого вектора $X^{(0)}$ и продолжать по формуле (3), то при определенных условиях эта последовательность будет приближаться к вектору X^* – неподвижной точке отображения $\Phi(X): R_n \rightarrow R_n$. Условия сходимости метода определяет следующее утверждение.

Утверждение 1. Пусть функция $\Phi(X)$ и замкнутое множество $M \subseteq D(\Phi) \subseteq R_n$ таковы, что: 1) $\Phi(X) \in M \quad \forall X \in M$; 2) $\exists q < 1: \|\Phi(X) - \Phi(\bar{X})\| \leq q \|X - \bar{X}\| \quad \forall X, \bar{X} \in M$. Тогда $\Phi(X)$ имеет в M единственную неподвижную точку X^* , последовательность $X^{(k)}$, определяемая методом простых итераций (3), при любом $X^{(0)} \in M$ сходится к X^* и справедливы оценки:

$$\|X^* - X^{(k)}\| \leq \frac{q}{1-q} \|X^{(k)} - X^{(k-1)}\| \leq \frac{q^k}{1-q} \|X^{(1)} - X^{(0)}\| \quad \forall k \in N.$$

При решении систем нелинейных уравнений, также как и в случае решения систем линейных алгебраических уравнений, может оказаться полезной модификация – аналог метода Зейделя. Можно реализовать так называемый *метод покомпонентных итераций*:

$$\begin{cases} x_1^{(k+1)} = \varphi_1(x_1^{(k)}, x_2^{(k)}, \dots, x_{n-1}^{(k)}, x_n^{(k)}) \\ x_2^{(k+1)} = \varphi_2(x_1^{(k+1)}, x_2^{(k)}, \dots, x_{n-1}^{(k)}, x_n^{(k)}) \\ \dots \\ x_n^{(k+1)} = \varphi_n(x_1^{(k+1)}, x_2^{(k+1)}, \dots, x_{n-1}^{(k+1)}, x_n^{(k)}) \end{cases} \tag{4}$$

Метод Ньютона и его модификации

Для нахождения приближенных решений системы уравнений (1) можно формально записать итерационный процесс:

$$X^{(k+1)} = X^{(k)} - A_k F(X^{(k)}), \quad k = 0, 1, 2, \dots, \quad (5)$$

определяющий большое семейство методов с матричными параметрами A_k .

Положим $A_k = [F'(X^{(k)})]^{-1}$, где

$$F'(X) = J(X) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \dots & \frac{\partial f_n}{\partial x_n} \end{pmatrix} \quad (6)$$

– матрица Якоби вектор-функции $F(X)$.

Подставляя (6) в (5), получим явную формулу метода Ньютона:

$$X^{(k+1)} = X^{(k)} - J(X^{(k)})F(X^{(k)}), \quad k = 0, 1, 2, \dots \quad (7)$$

Перепишем формулу (7) в явном виде:

$$F'(X^{(k)})(X^{(k+1)} - X^{(k)}) = -F(X^{(k)}), \quad k = 0, 1, 2, \dots \quad (8)$$

Сравнивая (8) с формальным разложением $F(X)$ в ряд Тейлора:

$$F(X) = F(X^{(k)}) + F'(X^{(k)})(X - X^{(k)}) + \frac{1}{2!}F''(X^{(k)})(X - X^{(k)})^2 + \dots, \quad (9)$$

можно заключить, что последовательность $X^{(k)}$ в методе Ньютона получается пошаговой линеаризацией соответствующих нелинейных уравнений системы (1). При достаточной гладкости $F(X)$ и достаточно хорошем начальном приближении $X^{(0)}$ сходимость порождаемой методом Ньютона последовательности $X^{(k)}$ к решению X^* будет квадратичной и в многомерном случае.

Необходимость решения n -линейных задач, а также обращение матрицы производных на каждом шаге приводит к росту вычислительных

затрат. Пути уменьшения таких затрат приводят к различным модификациям метода Ньютона.

Если матрицу Якоби вычислить и обратить всего один раз – в начальной точке $X^{(0)}$, получим *модифицированный метод Ньютона*:

$$X^{(k+1)} = X^{(k)} - J(X^{(k)})F(X^{(k)}), \quad k = 0, 1, 2, \dots \quad (10)$$

Этот метод требует значительно меньше вычислительных затрат на один итерационный шаг, но итераций при этом может потребоваться значительно больше для достижения заданной точности, метод имеет скорость сходимости геометрической прогрессии.

Вычисление и обращение матриц Якоби не на каждом шаге, а через несколько шагов приводят к *рекурсивному двушаговому ступенчатому процессу*:

$$\begin{cases} Z^{(k)} = X^{(k)} - A_k F(X^{(k)}), \\ X^{(k+1)} = Z^{(k)} - A_k F(Z^{(k)}) \end{cases} \quad (11)$$

На базе метода Ньютона можно построить близкий к нему по поведению итерационный процесс, не требующий вычисления производных. Заменяем частные производные в матрице Якоби разностными отношениями, подставив в формулу (5) матрицу $J(X^{(k)}, h^{(k)})^{-1}$, где

$$J(X, h) = \left(\frac{f_i(x_1, \dots, x_j + h_j, \dots, x_n) - f_i(x_1, \dots, x_j, \dots, x_n)}{h_j} \right)_{i,j=1}^n.$$

При удачном задании последовательности малых векторов $h^{(k)} = (h_1^{(k)}, h_2^{(k)}, \dots, h_n^{(k)})$ получим *разностный метод Ньютона*:

$$X^{(k+1)} = X^{(k)} - [J(X^{(k)}, h^{(k)})]^{-1} F(X^{(k)}), \quad (12)$$

имеющий сверхлинейную скорость сходимости.

Метод Брауна

В отличие от пошаговой линеаризации векторной функции $F(X)$, приведшей к методу Ньютона (5), *метод Брауна* предполагает проведение на каждом итерационном шаге поочередной линеаризации компонент вектор-

функции $F(X)$. Приведем расчетные формулы метода Брауна в двумерном случае. Пусть требуется найти решение системы уравнений:

$$\begin{cases} f(x, y) = 0, \\ g(x, y) = 0 \end{cases} \quad (13)$$

и пусть уже получены приближения x_k, y_k . Итерационный процесс метода Брауна определяется соотношениями:

$$\begin{aligned} \hat{x}_k &= x_k - \frac{f(x_k, y_k)}{f'_x(x_k, y_k)}, & q_k &= \frac{g(\hat{x}_k, y_k) \cdot f'_x(x_k, y_k)}{f'_x(x_k, y_k) \cdot g'_y(\hat{x}_k, y_k) - f'_y(x_k, y_k) \cdot g'_x(\hat{x}_k, y_k)} \\ p_k &= \frac{f(x_k, y_k) - q_k \cdot f'_y(x_k, y_k)}{f'_x(x_k, y_k)}, & x_{k+1} &= x_k - p_k, & y_{k+1} &= y_k - q_k \end{aligned} \quad (14)$$

Метод скорейшего спуска

Локальный характер сходимости всех рассмотренных выше методов затрудняет их применение в случаях, когда имеются проблемы с выбором хороших начальных приближений. В этом случае на помощь могут прийти численные методы оптимизации.

В этом методе решение системы (1) сводится к задаче отыскания минимумов функции

$$\Phi(x_1, x_2, \dots, x_n) = \sum_{i=1}^n f_i^2(x_1, x_2, \dots, x_n). \quad (15)$$

Так как функция (15) неотрицательная, то найдется точка $x^* = (x_1^*, x_2^*, \dots, x_n^*)$ такая, что

$$\Phi(x_1, x_2, \dots, x_n) \geq \Phi(x_1^*, x_2^*, \dots, x_n^*) \geq 0, \quad \forall (x_1, x_2, \dots, x_n) \in R^n.$$

Следовательно, если удастся найти точку x^* , минимизирующую функцию (15), и если при этом

$$\min_{(x_1, x_2, \dots, x_n) \in R^n} \Phi(x_1, x_2, \dots, x_n) = \Phi(x_1^*, x_2^*, \dots, x_n^*) = 0, \text{ то точка } x^* \text{ — истинное}$$

решение системы (1). Последовательность точек $x^{(k)}$ — приближений к точке x^* вычисляется по формуле $x^{(k+1)} = x^{(k)} - \lambda_k \cdot \Phi_x(x^{(k)})$,

где $\Phi_x(x^{(k)}) = \text{grad } \Phi(x^{(k)}) = \left(\frac{\partial \Phi(x^{(k)})}{\partial x_1}, \frac{\partial \Phi(x^{(k)})}{\partial x_2}, \dots, \frac{\partial \Phi(x^{(k)})}{\partial x_n} \right)$, а λ_k – находится

из условия минимума функции:

$$\psi_k(\lambda_k) = \Phi(x^{(k)} - \lambda_k \cdot \Phi_x(x^{(k)})). \quad (16)$$

Рис. 1 иллюстрирует алгоритм нахождения решения системы уравнений методом наискорейшего спуска.

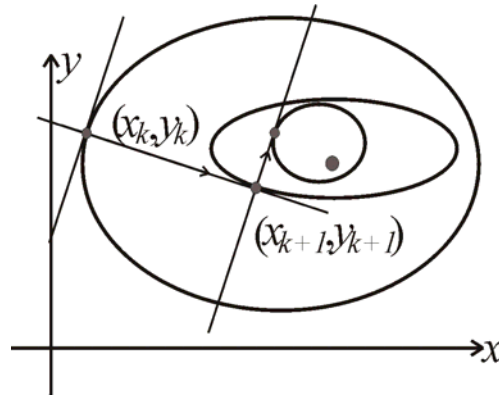


Рис. 1. – Траектория наискорейшего спуска для функции (13).

Главное достоинство градиентных методов решения нелинейных систем – глобальная сходимость. Процесс градиентного спуска приведет к какой-либо точке минимума функции из любой начальной точки. При определенных условиях найденная точка минимума будет искомым решением исходной нелинейной системы уравнений. Однако этот метод обладает и недостатком – медленной сходимостью. Поэтому на практике часто используют построение *гибридных алгоритмов*, которые начинали бы поиск решения искомого решения нелинейной системы глобально сходящимся градиентным методом, а затем производили бы уточнение каким-либо быстросходящимся методом, например, методом Ньютона.

Численные примеры. Реализация в пакете Matlab

Пример 1. Решить систему уравнений методом скорейшего спуска

$$\begin{cases} x^2 \cdot y^2 - 3 \cdot x^2 - 6 \cdot y^3 + 6 = 0, \\ x^4 - 9 \cdot y^2 + 1.5 = 0. \end{cases} \quad (17)$$

Для решения систем нелинейных уравнений средствами пакета Matlab

применяется функция **fsolve()**, возвращающая вектор-столбец, компоненты которого являются корнями системы.

Пример 2. Решить систему уравнений (17) с помощью встроенной функции Matlab.

ИНТЕРПОЛИРОВАНИЕ ФУНКЦИЙ

Пусть функция $f(x)$ задана множеством своих значений для дискретного набора точек:

x_0	x_1	\dots	x_n
f_0	f_1	\dots	f_n

здесь $f_i = f(x_i)$.

Требуется найти интерполяционный многочлен $P(x) = P_n(x)$ степени не выше n , значения которого в узлах интерполяции x_i совпадают со значениями данной функции $P(x_i) = f_i$.

Интерполяционный полином Лагранжа

Для функции $f(x)$, заданной таблицей, построим интерполяционный многочлен $L_n(x)$, степень которого не выше n , в следующем виде:

$$L_n(x) = l_0(x) + l_1(x) + \dots + l_n(x),$$

где $l_i(x)$ – многочлен степени n , причем

$$l_i(x_k) = \begin{cases} f_i, & \text{если } i = k, \\ 0, & \text{если } i \neq k. \end{cases} \quad (1)$$

Многочлены $l_i(x)$ составим следующим образом

$$l_i(x) = c_i (x - x_0)(x - x_1) \dots (x - x_{i-1})(x - x_{i+1}) \dots (x - x_n),$$

где c_i – постоянный коэффициент, значение которого находится из первой части условия (1)

$$c_i = \frac{f_i}{(x_i - x_0)(x_i - x_1) \dots (x_i - x_{i-1})(x_i - x_{i+1}) \dots (x_i - x_n)}.$$

Тогда *интерполяционный многочлен Лагранжа* имеет вид

$$L_n(x) = \sum_{i=1}^n f_i \frac{(x-x_0)(x-x_1)\dots(x-x_{i-1})(x-x_{i+1})\dots(x-x_n)}{(x_i-x_0)(x_i-x_1)\dots(x_i-x_{i-1})(x_i-x_{i+1})\dots(x_i-x_n)}$$

Интерполяционные формулы Ньютона

Формулы Ньютона предназначены для таблиц с равноотстоящими узлами, т.е. $h = x_{i+1} - x_i$, $i = \overline{1, n}$. Определим разности между значениями функции в узлах интерполяции. Конечная разность первого порядка имеет вид

$$\Delta f_i = f_{i+1} - f_i.$$

Конечная разность второго порядка:

$$\Delta^2 f_i = \Delta f_{i+1} - \Delta f_i = f_{i+2} - 2f_{i+1} + f_i.$$

Конечная разность n -го порядка вычисляется по формуле:

$$\Delta^k f_i = f_{i+k} - k \cdot f_{i+k-1} + \frac{k(k-1)}{2!} f_{i+k-2} - \dots + (-1)^k f_i.$$

Первая интерполяционная формула Ньютона используется для интерполирования и экстраполирования в точках x , близких к началу таблицы x_0 , и имеет следующий вид:

$$P_n(x) = f_0 + t \cdot \Delta f_0 + \frac{t(t-1)}{2!} \Delta^2 f_0 + \dots + \frac{t(t-1) \cdot \dots \cdot (t-n+1)}{n!} \Delta^n f_0,$$

где $t = \frac{x-x_0}{h}$, число n выбирают так, чтобы конечные разности n -го порядка были практически постоянными.

Когда значение аргумент находится ближе к концу отрезка интерполяции x_n , используется *вторая интерполяционная формула Ньютона*:

$$P_n(x) = f_n + q \cdot \Delta f_{n-1} + \frac{q(q+1)}{2!} \Delta^2 f_{n-2} + \dots + \frac{q(q+1) \cdot \dots \cdot (q+n-1)}{n!} \Delta^n f_0,$$

где $q = \frac{x-x_n}{h}$.

Сплайн-интерполяция

При большом количестве узлов интерполяции приходится использовать полиномы высокой степени. Можно этого избежать, разбив отрезок интерполяции на несколько частей и построив на каждой части свой интерполяционный многочлен. Существенный недостаток такого интерполирования состоит в том, что в точках сшивки разных интерполяционных полиномов их первая производная будет разрывной, поэтому для решения задачи кусочно-линейной интерполяции используют особый вид кусочно-полиномиальной интерполяции – сплайн-интерполяцию.

Сплайн – это функция, которая на каждом частичном отрезке интерполирования является алгебраическим многочленом, а на заданном отрезке непрерывна вместе с несколькими своими производными.

Пусть интерполируемая функция задана таблично. Длина частичного отрезка $[x_{i-1}, x_i] - h_i = x_i - x_{i-1}$, $i = \overline{1, n}$.

Найдем кубический сплайн на каждом из отрезков $[x_{i-1}, x_i]$:

$$S(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3, \quad (2)$$

где a_i, b_i, c_i, d_i – неизвестные коэффициенты.

Пусть значения $S(x)$ совпадают в узлах с табличными значениями функции $f(x)$

$$\begin{aligned} S(x_{i-1}) &= f_{i-1} = a_i, \\ S(x_i) &= f_i = a_i + b_i h_i + c_i h_i^2 + d_i h_i^3. \end{aligned} \quad (3)$$

Для получения дополнительных условий потребуем непрерывности первой и второй производных сплайна (2) во всех точках, включая узлы:

$$\begin{aligned} \frac{dS(x)}{dx} &= b_i + 2c_i(x - x_{i-1}) + 3d_i(x - x_{i-1})^2, \\ \frac{d^2S(x)}{dx^2} &= 2c_i + 6d_i(x - x_{i-1}). \end{aligned} \quad (4)$$

Найдем левые и правые производные (4) во внутреннем узле x_i :

$$\frac{dS(x_i - 0)}{dx} = b_i + 2c_i h_i + 3d_i h_i^2, \quad \frac{dS(x_i + 0)}{dx} = b_{i+1},$$

$$\frac{d^2 S(x_i - 0)}{dx^2} = 2c_i + 6d_i h_i, \quad \frac{d^2 S(x_i + 0)}{dx^2} = 2c_{i+1}, \quad i = \overline{1, n-1}. \quad (5)$$

Приравнявая найденные производные, получаем $2(n-1)$ условие. Потребуем нулевой кривизны сплайна на концах отрезка интерполирования

$$c_1 = 0, \quad c_n + 3d_n h_n = 0. \quad (6)$$

Решая систему (3), (5), (6), найдем значения неизвестных коэффициентов, определяющих совокупность всех формул для искомого интерполяционного сплайна

$$S_i(x) = f_{i-1} + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3.$$

Численные примеры. Реализация в пакете Matlab

Пример 1. Решить задачу интерполяции с помощью многочлена Лагранжа для функции $f(x) = \sin(x)$, заданной таблично на интервале $[0, 2\pi]$ при $n = 8$.

Пример 2. Решить задачу интерполяции для функции $f(x) = \sin(x)$, заданной таблично на интервале $[0, 2\pi]$ при $n = 8$ средствами пакета Matlab.

Для решения задачи одномерной интерполяции в пакете Matlab используется функция **interp1()**.

Решение задачи с помощью кубических сплайнов можно построить, используя функцию пакета Matlab **spline()**.

ОБРАБОТКА ЭКСПЕРИМЕНТАЛЬНЫХ ДАННЫХ

Пусть в результате измерений получена таблица некоторой зависимости $f(x)$:

x	x_1	x_2	...	x_n
$F(x)$	y_1	y_2	...	y_n

Требуется найти формулу, выражающую данную зависимость аналитически. Один из подходов состоит в построении интерполяционного многочлена, значения которого в узлах интерполяции x_i совпадают со значениями данной функции $f(x_i) = f_i$.

Если значения функции $f(x)$ известны с некоторой погрешностью, то требование совпадения значений в узлах интерполяции не оправдано, поскольку оно не означает совпадение характеров исходной и интерполирующей функции. Поэтому поставим задачу следующим образом – найти функцию вида

$$y = F(x),$$

которая в точках x_1, x_2, \dots, x_n принимает значения, близкие к табличным значениям y_1, y_2, \dots, y_n .

Предположим, что приближающая функция $F(x)$ в точках x_1, x_2, \dots, x_n имеет значения $\bar{y}_1, \bar{y}_2, \dots, \bar{y}_n$. Тогда нужно найти функцию $F(x)$ определенного вида так, чтобы сумма квадратов

$$(y_1 - \bar{y}_1)^2 + (y_2 - \bar{y}_2)^2 + \dots + (y_n - \bar{y}_n)^2 \quad (1)$$

была наименьшей.

Нахождение приближающей функции в виде линейной

Пусть приближающая функция имеет вид

$$F(x) = ax + b, \quad (2)$$

где a, b – параметры.

Составим сумму вида (1) для этого случая:

$$\Phi(a, b) = \sum_{i=1}^n (y_i - ax_i - b)^2 \rightarrow \min \quad (3)$$

Функция $\Phi(a, b)$ является неотрицательной квадратичной, поэтому в некоторой области она имеет единственную точку минимума (a^*, b^*) , удовлетворяющую условиям:

$$\frac{\partial \Phi}{\partial a} = 0, \quad \frac{\partial \Phi}{\partial b} = 0,$$

т.е. системе уравнений:

$$\begin{cases} \sum_{i=1}^n (y_i - ax_i - b)x_i = 0, \\ \sum_{i=1}^n (y_i - ax_i - b) = 0. \end{cases}$$

Разделив каждое уравнение на n , получаем

$$\begin{cases} \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i^2 \right) \cdot a + \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i \right) \cdot b = \frac{1}{n} \cdot \sum_{i=1}^n x_i y_i, \\ \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i \right) \cdot a + b = \frac{1}{n} \cdot \sum_{i=1}^n y_i. \end{cases} \quad (5)$$

Вычислив значения параметров a , b , получаем конкретный вид функции (2).

В зависимости от характера табличных данных, изучаемого с помощью их изображения в соответствующей системе координат, при обработке экспериментальных данных часто используют иные семейства двухпараметрических функций. Существует возможность с помощью подходящего преобразования переменных получить линейную зависимость и использовать метод (5) для следующих семейств функций:

Функция $y = f(x)$	Линеаризованная форма, $Y = ax + b$	Замена переменных и постоянных
$y = \frac{a}{x} + b$	$y = a \cdot \frac{1}{x} + b$	$X = \frac{1}{x}, Y = y$
$y = \frac{d}{x+c}$	$y = -\frac{1}{c} \cdot xy + \frac{d}{c}$	$X = xy, Y = y,$ $c = -\frac{1}{a}, d = -\frac{b}{a}$
$y = \frac{1}{ax+b}$	$\frac{1}{y} = ax + b$	$X = x, Y = \frac{1}{y}$
$y = \frac{x}{ax+b}$	$\frac{1}{y} = a \cdot \frac{1}{x} + b$	$X = \frac{1}{x}, Y = \frac{1}{y}$
$y = a \cdot \ln(x) + b$	$y = a \cdot \ln(x) + b$	$X = \ln(x), Y = y$

$y = c \cdot e^{ax}$	$\ln(y) = ax + \ln(c)$	$X = x, Y = \ln(y), c = e^b$
$y = c \cdot x^a$	$\ln(y) = a \cdot \ln(x) + \ln(c)$	$X = \ln(x), Y = \ln(y),$ $c = e^b$
$y = \frac{1}{(ax + b)^2}$	$\frac{1}{\sqrt{y}} = ax + b$	$X = x, Y = \frac{1}{\sqrt{y}}$
$y = \frac{c \cdot x}{e^{dx}}$	$\ln\left(\frac{y}{x}\right) = -dx + \ln(c)$	$X = x, Y = \ln\left(\frac{y}{x}\right),$ $c = e^b, d = -a$
$y = \frac{1}{1 + c \cdot e^{ax}}$	$\ln\left(\frac{1}{y} - 1\right) = ax + \ln(c)$	$X = x, Y = \ln\left(\frac{1}{y} - 1\right),$ $c = e^b$

Нахождение приближающей функции в виде квадратного трехчлена

Приближающая функция имеет вид:

$$F(x) = ax^2 + bx + c, \quad (6)$$

где a, b, c – параметры.

Составим сумму вида (1) как функцию $\Phi(a, b)$ для этого случая:

$$\Phi(a, b) = \sum_{i=1}^n (y_i - ax_i^2 - bx_i - c)^2 \rightarrow \min \quad (7)$$

Функция $\Phi(a, b)$ является неотрицательной квадратичной, поэтому в некоторой области она имеет единственную точку минимума (a^*, b^*) , удовлетворяющую условиям:

$$\frac{\partial \Phi}{\partial a} = 0, \quad \frac{\partial \Phi}{\partial b} = 0,$$

т.е. системе уравнений:

$$\begin{cases} \sum_{i=1}^n (y_i - ax_i^2 - bx_i - c)x_i^2 = 0, \\ \sum_{i=1}^n (y_i - ax_i^2 - bx_i - c)x_i = 0, \\ \sum_{i=1}^n (y_i - ax_i^2 - bx_i - c) = 0. \end{cases}$$

Разделив каждое уравнение на n , имеем

$$\begin{cases} \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i^4\right) \cdot a + \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i^3\right) \cdot b + \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i^2\right) \cdot c = \frac{1}{n} \cdot \sum_{i=1}^n x_i^2 y_i, \\ \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i^3\right) \cdot a + \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i^2\right) \cdot b + \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i\right) \cdot c = \frac{1}{n} \cdot \sum_{i=1}^n x_i y_i, \\ \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i^2\right) \cdot a + \left(\frac{1}{n} \cdot \sum_{i=1}^n x_i\right) \cdot b + c = \frac{1}{n} \cdot \sum_{i=1}^n y_i. \end{cases}$$

Решив систему относительно неизвестных a , b , c , находим значения параметров приближающей функции (6).

Численные примеры. Реализация в пакете Matlab

Пример 1. Даны табличные значения квадратичной зависимости. Найти коэффициенты квадратичной аппроксимирующей функции (6).

x	0	1	2	3	4	5	6	7	8	9	10
y	-0.28	0.42	2.11	4.82	7.75	12.43	12.16	15.41	23.07	31.06	36.68

Пример 2. Используя табличные значения примера 1, найти параметры квадратичной аппроксимирующей функции (6) с помощью встроенной функции Matlab.

Для решения задачи обобщенной нелинейной регрессии в пакете Matlab имеется функция **lsqnonlin()**, возвращающая решение задачи нахождения точки минимума функции:

$$\min_x (f(x)) = f_1^2(x) + f_2^2(x) + \dots + f_n^2(x) + L,$$

где $f(x)$ – вектор-функция, x – столбец искомых переменных, L – искомая константа.

ЧИСЛЕННОЕ ДИФФЕРЕНЦИРОВАНИЕ И ИНТЕГРИРОВАНИЕ

Методики дифференцирования и интегрирования имеют важное применение при разработке алгоритмов многих прикладных задач, в т. ч. задач, сводящихся к решениям дифференциальных уравнений и уравнениям в частных производных. К сожалению, аналитические методы не всегда оказываются применимыми. Численные методы дифференцирования приходят на помощь в следующих случаях: аналитический вид функции не задан, возможно сильное усложнение функции при ее аналитическом дифференцировании или требуется получить значения производных с помощью одноплатных вычислительных процессов без привлечения аналитических выкладок. Задачи численного интегрирования возникают каждый раз, когда необходимо провести интегрирование функций, для которых в общем случае первообразная среди элементарных функций может и не существовать.

Численное дифференцирование аналитически заданных функций

По определению производная функции $f(x)$ равна:

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

Переходя от бесконечно малых разностей к конечным, получаем приближенную формулу численного дифференцирования:

$$f'(x) \approx \frac{f(x + \Delta x) - f(x)}{\Delta x}. \quad (1)$$

Используя разложение функции $f(x)$ в ряд Тейлора, можно записать

$$f'(x) \approx f'(x) + \frac{f''(x)}{2!} \Delta x + \dots \quad (2)$$

Согласно формуле (2), основной член погрешности равен $\frac{f''(x)}{2!} \Delta x$, т.е.

формула (1) имеет первый порядок точности по Δx .

Используя симметричную разностную схему, можно записать следующее:

$$f'(x) \approx \frac{f(x + \Delta x) - f(x - \Delta x)}{2 \cdot \Delta x}. \quad (3)$$

Используя разложение в ряд Тейлора, получаем

$$f'(x) \approx f'(x) + \frac{f'''(x)}{3!} (\Delta x)^2 + \dots \quad (4)$$

Формула (4) имеет второй порядок точности по Δx .

Численное дифференцирование функций, заданных таблицей

Пусть функция $f(x)$ задана таблично в конечном числе точек отрезка $[a, b]$. Требуется определить значение производной в некоторой точке отрезка $[a, b]$.

Выбрав $(n + 1)$ узлов, заменим функцию $f(x)$ интерполяционным многочленом $P_n(x)$. Тогда производная от этого многочлена применяется для приближенного представления производной функции $f(x)$:

$$f'(x) \approx P_n'(x).$$

Как правило, формулы численного дифференцирования применяют для нахождения производных в узлах x_i . Дифференцирование интерполяционных многочленов Ньютона в точке x_0 приводит к формуле:

$$f'(x_0) \approx \frac{1}{h} \left(\Delta f_0 - \frac{1}{2} \Delta^2 f_0 + \frac{1}{3} \Delta^3 f_0 - \dots + \frac{(-1)^{n-1}}{n} \Delta^n f_0 \right), \quad (5)$$

$$f'(x_0) \approx \frac{1}{h} \left(\Delta f_{-1} + \frac{1}{2} \Delta^2 f_{-2} + \frac{1}{3} \Delta^3 f_{-3} - \dots + \frac{1}{n} \Delta^n f_{-n} \right). \quad (6)$$

Формула (5) применяется для начальных строк таблицы, формула (6) – для последних строк.

Для середины таблицы применяют формулы центрированной разности второго порядка точности по h .

$$\begin{aligned}f'(x_0) &\approx \frac{f_1 - f_{-1}}{2h}, \\f''(x) &\approx \frac{f_1 - 2f_0 + f_{-1}}{h^2}, \\f'''(x) &\approx \frac{f_2 - f_1 + 2f_{-1} - f_{-2}}{2h^3},\end{aligned}\tag{7}$$

где $f_k = f(x_0 + kh)$, $k = -2, -1, 0, 1, 2$.

Численное интегрирование методом прямоугольников

Геометрический смысл определенного интеграла

$$F = \int_a^b f(x) dx$$

– площадь фигуры, ограниченной графиком функции $f(x)$ и прямыми:

$$x = a, \quad x = b.$$

Разделим отрезок $[a, b]$ на n равных отрезков длиной Δx :

$$\Delta x = \frac{b - a}{n}.$$

Координата правого конца i -го отрезка определяется по формуле

$$x_i = x_0 + i \cdot \Delta x,$$

где $x_0 = a$, $i = \overline{0, n}$.

Простейшая оценка площади кривой может быть получена как сумма площадей прямоугольников, одна из сторон которого равна длине отрезка $[x_i, x_{i+1}]$, а высота равна значению $f(x_i)$ – *метод левых прямоугольников* (рис. 1), или $f(x_{i+1})$ – *метод правых прямоугольников* (рис. 2).

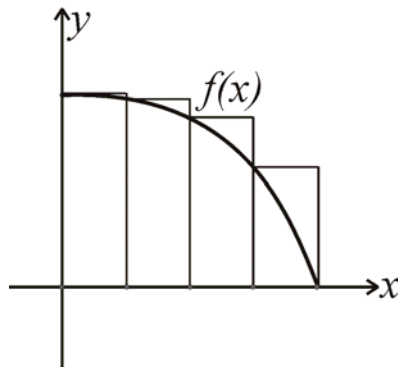


Рис. 1. – Геометрическая интерпретация метода левых прямоугольников.

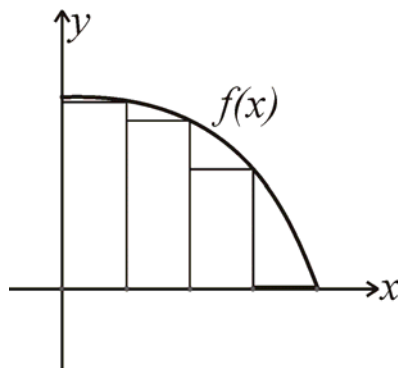


Рис. 2. – Геометрическая интерпретация метода правых прямоугольников.

Тогда для левых прямоугольников определенный интеграл вычисляется по формуле

$$F_L = \sum_{i=0}^{n-1} f(x_i) \cdot \Delta x, \quad (8)$$

а для правых прямоугольников – по формуле

$$F_R = \sum_{i=1}^n f(x_i) \cdot \Delta x. \quad (9)$$

Погрешность метода левых и правых прямоугольников пропорциональна n^{-1} .

Численное интегрирование методом трапеций

Заменим функцию на каждом интервале $[x_i, x_{i+1}]$, $i = \overline{0, n-1}$ отрезком прямой, проходящей через точки $(x_i, f(x_i))$ и $(x_{i+1}, f(x_{i+1}))$. Тогда фигура, ограниченная графиком функции и прямыми $x = x_i$, $x = x_{i+1}$, является трапецией.

Тогда определенный интеграл определяется как сумма площадей всех трапеций по формуле:

$$F_n = \frac{1}{2} \sum_{i=0}^{n-1} (f(x_{i+1}) + f(x_i)) \cdot \Delta x = \left[\frac{1}{2} f(a) + \sum_{i=1}^{n-1} f(x_i) + \frac{1}{2} f(b) \right] \cdot \Delta x. \quad (10)$$

Полная погрешность формулы трапеций на отрезке $[a, b]$ по порядку величины равна $O(n^{-2})$.

Численное интегрирование методом Симпсона

Формула получается при использовании параболической интерполяции по трем соседним точкам

$$y = ax^2 + bx + c. \quad (11)$$

Для нахождения параметров a , b , c полинома, проходящего через три точки (x_0, y_0) , (x_1, y_1) , (x_2, y_2) , решаем систему линейных уравнений

$$\begin{cases} y_0 = ax_0^2 + bx_0 + c, \\ y_1 = ax_1^2 + bx_1 + c, \\ y_2 = ax_2^2 + bx_2 + c. \end{cases} \quad (12)$$

Решив систему (12), подставим найденные значения в (11) и получим

$$y = y_0 \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} + y_1 \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} + y_2 \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}. \quad (13)$$

Интегрируя (13) на отрезке $[x_0, x_2]$, находим

$$F_0 = \frac{1}{3} (y_0 + 4y_1 + y_2) \cdot \Delta x,$$

где $\Delta x = x_1 - x_0 = x_2 - x_1$.

Искомый определенный интеграл находится как площадь всех параболических сегментов:

$$F_n = \frac{1}{3} [f(a) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \dots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(b)] \cdot \Delta x.$$

В формуле Симпсона число n должно быть четным.

Полная погрешность формулы Симпсона на отрезке $[a, b]$ по порядку величины составляет $O(n^{-4})$.

Численное интегрирование методом трех восьмых

$$F_n = \frac{3 \cdot \Delta x}{8} [f(a) + f(x_{3m}) + 2(f(x_3) + f(x_6) + \dots + f(x_{3m-3})) + 3(f(x_1) + f(x_2) + f(x_4) + f(x_5) + \dots + f(x_{3m-2}) + f(x_{3m-1}))], \quad (14)$$

где $\Delta x = \frac{b-a}{n} = \frac{b-a}{3m}$.

В формуле (14) число n должно быть равно $3m$.

Значение полной погрешности правила трех восьмых по порядку величины совпадает со значением полной погрешности формулы Симпсона.

Заметим, что оценка погрешности вычисления интеграла от функции, зависящей от d переменных, определяется следующим образом: если для одномерного случая погрешность составляет $O(n^{-\alpha})$, то в d -мерном случае

она равна $O\left(n^{-\frac{\alpha}{d}}\right)$.

Численное интегрирование методом Монте-Карло

Пусть существует прямоугольник высотой H и длиной $(b-a)$, такой, что функция $f(x)$ целиком лежит внутри прямоугольника. Сгенерируем n пар случайных чисел, равномерно распределенных в данном прямоугольнике: $a \leq x_i \leq b$, $0 \leq y_i \leq H$.

Доля точек (x_i, y_i) , удовлетворяющих условию $y_i \leq f(x_i)$, является оценкой отношения интеграла от функции $f(x)$ к площади рассматриваемого прямоугольника. Оценка интеграла может быть получена по формуле

$$F_n = A \frac{n_s}{n}, \quad (15)$$

где n_s – число точек, удовлетворяющих условию $y_i \leq f(x_i)$, A – площадь прямоугольника.

Можно вычислить определенный интеграл как среднее значение функции $f(x)$ на отрезке $[a, b]$:

$$F_n = \frac{(b-a)}{n} \sum_{i=1}^n f(x_i), \quad (16)$$

где x_i – последовательность случайных чисел с равномерным законом распределения на отрезке $[a, b]$.

В отличие от рассмотренных выше методов, погрешность метода Монте-Карло не зависит от размерности подынтегральной функции d и составляет $O(n^{-0.5})$. Поэтому при достаточно больших d интегрирование по методу Монте-Карло приводит к меньшим погрешностям при тех же значениях n .

Численные примеры. Реализация в Matlab

Пример 1. Вычислить значение производной функции на отрезке $[0, 6\pi]$.

$$f(x) = \sin(0.1 \cdot x^2) \quad (17)$$

Для аппроксимации производных конечными разностями в пакете Matlab существует функция **diff()**:

diff(x) – возвращает конечные разности, вычисленные по смежным элементам вектора x ;

diff(x, n) – возвращает конечные разности n -го порядка, вычисленные по смежным элементам вектора x .

Пример 2. Вычислить значение производной функции (17) на отрезке $[0, 6\pi]$ с использованием соответствующей функции Matlab.

Пример 3. Вычислить значение определенного интеграла

$$\int_0^{\sqrt{\frac{\pi}{2}}} x \cdot \sin(x^2) dx \quad (18)$$

методом правых прямоугольников.

Пример 4. Вычислить значение определенного интеграла (18) методом Симпсона.

Пример 5. Вычислить значение определенного интеграла (18) методом Монте-Карло.

Для вычисления значений определенных интегралов в пакете Matlab применяются функции **quad ()**, **quad1 ()**, **trapez ()**.

Функция **quad (fun, a, b)** – возвращает значение интеграла от функции *fun* на отрезке $[a, b]$, при вычислении используется метод Симпсона.

Функция **quad1 (fun, a, b)** – возвращает значение интеграла от функции *fun* на отрезке $[a, b]$, используя для вычисления метод Лоббато.

Функция **trapez (y)** – возвращает значение определенного интеграла в предположении, что $x = 1 : \text{length}(y)$.

Функция **trapez (x, y)** – возвращает значение определенного интеграла на отрезке $[x(1), x(n)]$.

Пример 6. Вычислить значение определенного интеграла (18) с помощью встроенной функции Matlab **trapez ()**.

ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

Рассмотрим обыкновенное дифференциальное уравнение первого порядка:

$$y' = f(x, y), \quad x \in [x_0, b] \tag{1}$$

с начальным условием:

$$y(x_0) = y_0, \tag{2}$$

где $f(x, y)$ – некоторая заданная, в общем случае, нелинейная функция двух переменных.

Будем считать, что для задачи (1) – (2), называемой *задачей с начальными условиями* или *задачей Коши*, выполняются требования,

обеспечивающие существование и единственность ее решения $y = y(x)$ на отрезке $[x_0, b]$, также будем считать, что искомое решение обладает той или иной степенью гладкости.

Аналитическое решение задачи (1) удается найти только для специальных типов уравнений. Используемые на практике приближенные методы решения задач Коши для обыкновенных дифференциальных уравнений можно условно разделить на три группы:

1. *Приближенно-аналитические методы.* К методам этой группы относят такие, которые позволяют находить приближение решения $y = y(x)$ сразу в виде некоторой функции $\varphi(x)$. Таким методом является, напр., метод степенных рядов, реализация которого основана на представлении искомой функции отрезком ряда Тейлора с учетом заданной точности, где коэффициенты определяются последовательным дифференцированием самого уравнения. К этой группе методов относят также методы неопределенных коэффициентов и последовательных приближений.

2. *Графические методы* основаны на приближенном представлении искомого решения $y = y(x)$ на промежутке $[x_0, b]$ в виде графика, который можно строить по тем или иным правилам, связанными с графическим толкованием задачи.

3. *Численные методы* предполагают получение набора приближенных значений y_i искомого решения $y(x)$ на некоторой сетке аргументов $x_i \in [x_0, b]$.

Метод Пикара

Проинтегрируем обе части уравнения (1) в границах от x_0 до x :

$$\int_{x_0}^x y'(t) dt = \int_{x_0}^x f(t, y(t)) dt .$$

Отсюда, с учетом условия (2), получим:

$$y(x) = y_0 + \int_{x_0}^x f(t, y(t)) dt . \quad (3)$$

Применяя к последнему интегральному уравнению метод простых итераций:

$$y_{n+1} = \varphi(y_n), \quad n = 0, 1, 2, \dots,$$

и выбирая в качестве начальной функции $y(x_0) = y_0$, по формуле (3) при $n=0$ находим первое приближение:

$$y_1(x) = y_0 + \int_{x_0}^x f(t, y_0) dt,$$

подстановка которого в (3) при $n = 1$ дает второе приближение, и т.д. Таким образом, получим *метод последовательных приближений* или *метод Пикара*:

$$y_{n+1}(x) = y_0 + \int_{x_0}^x f(t, y_n(t)) dt, \quad n = 0, 1, 2, \dots \quad y_0(x) \equiv y_0. \quad (4)$$

Оценка погрешности k -го приближения имеет вид:

$$|y(x) - y_k(x)| \leq M^k N \frac{d^{k+1}}{(k+1)!}, \quad (5)$$

где $M = \max |f'_y(x)|$ – константа Липшица, $|f(x, y)| \leq N$ – верхняя грань модуля функции $f(x, y)$, d – величина, определяющая окрестность $|x - x_0| \leq d$, $d = \min\left(a, \frac{b}{N}\right)$.

Метод Эйлера

Пусть требуется построить таблицу приближенных значений y_i решения $y = y(x)$ задачи Коши (1) – (2) в расчетных точках

$x_i = x_0 + ih$, $i = \overline{1, n}$ с расчетным шагом $h = \frac{b - x_0}{n}$:

x	x_0	x_1	...	$x_n = b$
y	y_0	y_1	...	$y_n \approx y(b)$

Для построения метода Эйлера можно использовать различные подходы. Рассмотрим идею графического построения решения дифференциальных уравнений (рис. 1).

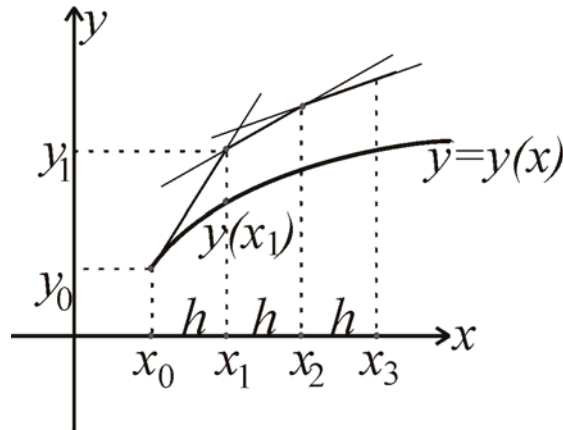


Рис. 1. – Геометрическая интерпретация метода Эйлера.

Запишем уравнение касательной к кривой $y = y(x)$ в точке (x_0, y_0) :

$$y = y_0 + f(x_0, y_0)(x - x_0). \quad (6)$$

При достаточно малом шаге h ордината $y_1 = y_0 + hf(x_0, y_0)$ этой касательной (при подстановке значения $x_1 = x_0 + h$) по непрерывности должна мало отличаться от ординаты $y(x_1)$ решения $y(x)$ задачи (1) – (2). Следовательно, точка (x_1, y_1) пересечения касательной (6) с прямой $x = x_1$ может быть приближенно принята за начальную точку. Через эту точку снова проведем прямую $y = y_1 + f(x_1, y_1)(x - x_1)$, которая уже приближенно отражает поведение касательной к $y = y(x)$ в точке $(x_1, y(x_1))$. Пересекая эту касательную с прямой $x = x_2$, получив приближенное значение $y(x_2)$ и т.д., строим итерационный процесс *метода Эйлера*:

$$y_{i+1} = y_i + hf(x_i, y_i), \quad i = 0, 1, 2, \dots, n, \quad (7)$$

который приближенно график решения $y = y(x)$ представляет в виде ломанной, составленной из отрезков приближенных касательных.

Соотношение (1) можно получить и аналитически. Линеаризуя решение в окрестности начальной точки по формуле Тейлора, получим:

$$y(x) = y(x_0) + y'(x_0)(x - x_0) + \frac{y''(\xi)}{2!}(x - x_0)^2,$$

откуда легко получить и саму формулу метода Эйлера (7) для n последовательных шагов и ее остаточный член: $r_k(h) = \frac{y''(\xi_k)}{2}h^2$, где ξ_k – некоторая точка интервала (x_0, x_k) . Остаточный член характеризует локальную ошибку метода Эйлера – $O(h^2)$, совершаемую на каждом шаге. После $n \sim \frac{1}{h}$ шагов глобальная ошибка будет $O(h)$, т.е. метод Эйлера относится к методам первого порядка.

Другим подходом к получению формулы (7) является аппроксимация производной в выражении (1) правыми разностями.

Известны различные модификации метода Эйлера, которые направлены на уточнение направления перехода из точки (x_i, y_i) в точку (x_{i+1}, y_{i+1}) . Например, в *методе Эйлера-Коши* используется следующий порядок вычисления:

$$y_{i+1}^* = y_i + hf(x_i, y_i), \quad y_{i+1} = y_i + h \frac{f(x_i, y_i) + f(x_{i+1}, y_{i+1}^*)}{2}. \quad (8)$$

Метод второго порядка точности можно получить, используя разложение в ряд Тейлора функции $y(x)$ в окрестности некоторой точки x_i :

$$y(x_{i+1}) = y(x_i) + hy'(x_i) + \frac{1}{2!}h^2 y''(x_i) + O(h^3). \quad (9)$$

Дифференцируя (1) по формуле полной производной

$$y''(x) = f'_x(x, y) + f'_y(x, y)y'$$

найдем приближенное значение второй производной:

$$\begin{aligned}
y''(x_i) &= f'_x(x_i, y(x_i)) + f'_y(x_i, y(x_i))f(x_i, y(x_i)) = \\
&= f'_x(x_i, y_i) + f'_y(x_i, y_i)f(x_i, y_i).
\end{aligned}$$

Подставляя последнее выражение в (9), получим *исправленный метод Эйлера*:

$$y(x_{i+1}) = y_i + h \left[f(x_i, y_i) + \frac{h}{2} (f'_x(x_i, y_i) + f'_y(x_i, y_i) \cdot f(x_i, y_i)) \right]. \quad (10)$$

7.3. Метод Рунге-Кутты

Идея построения явных *методов Рунге-Кутты* p -го порядка заключается в получении приближений к значениям $f(x_{i+1})$ по формуле вида:

$$y_{i+1} = y_i + h\varphi(x_i, y_i, h), \quad (11)$$

где $\varphi(x, y, h)$ – некоторая функция, приближающая отрезок ряда Тейлора до

p -го порядка и не содержащая частных производных функции $f(x, y)$.

Полагая в (11), что $\varphi(x, y, h) \equiv f(x, y)$, приходим к методу Эйлера, который можно считать частным случаем метода Рунге-Кутты при $p = 1$.

Рассмотрим построение методов Рунге-Кутты для $p = 2$. Пусть функция $\varphi(x, y, h)$ имеет следующую структуру:

$$\varphi(x, y, h) = c_1 f(x, y) + c_2 f(x + ah, y + bhf(x, y)).$$

Параметры c_1, c_2, a, b будем подбирать так, чтобы формула (11) определяла метод второго порядка, т.е. чтобы максимальная локальная ошибка составляла величину $O(h^3)$. Разложим функцию двух переменных $f(x + ah, y + bhf(x, y))$ по формуле Тейлора, ограничиваясь линейными членами:

$$f(x + ah, y + bhf(x, y)) = f(x, y) + f'_x(x, y)ah + f'_y(x, y)bhf(x, y) + O(h^2).$$

Подстановка этого выражения в (11) дает:

$$y_{i+1} = y_i + h[(c_1 + c_2)f(x_i, y_i) + h(c_2af'_x(x_i, y_i) + c_2bf'_y(x_i, y_i)f(x_i, y_i))] + O(h^3)$$

Сравнение последнего выражения с тейлоровским квадратичным представлением решения $y(x)$ (10) с точностью до $O(h^3)$ требует выполнения совокупности условий:

$$\begin{cases} c_1 + c_2 = 1, \\ c_2a = 0.5, \\ c_2b = 0.5, \end{cases}$$

считая c_2 свободным параметром α , получим *однопараметрическое семейство методов Рунге-Кутты второго порядка*:

$$y_{i+1} = y_i + h \left[(1 - \alpha)f(x_i, y_i) + \alpha f \left(x_i + \frac{h}{2\alpha}, y_i + \frac{h}{2\alpha} f(x_i, y_i) \right) \right] + O(h^3) \quad (12)$$

)

Данный метод является *двухэтапным* по количеству вычислений на каждом шаге.

Анализ метода Рунге-Кутты второго порядка позволяет представить, в какой форме следует конструировать *метод Рунге-Кутты произвольного порядка*. Используется запись, состоящая из следующей последовательности формул, параметры c_k, a_k, b_{kj} в которых подбираются так, чтобы получаемое значение y_{i+1} совпадало со значением разложения $y(x_{i+1})$ по формуле Тейлора с погрешностью $O(h^{p+1})$:

$$\begin{cases} \theta_1^i = f(x_i, y_i), \\ \theta_k^i = f(x_i + a_k h, y_i + h \sum_{j=1}^{k-1} b_{kj} \theta_j^i), \quad k = 2, 3, \dots, p, \\ y_{i+1} = y_i + h \sum_{k=1}^p c_k \theta_k^i. \end{cases} \quad (13)$$

Наиболее употребительным частным случаем семейства методов (13) является метод Рунге-Кутты четвертого порядка:

$$\begin{cases} \theta_1^i = f(x_i, y_i), \\ \theta_2^i = f(x_i + \frac{h}{2}, y_i + \frac{h}{2} \theta_1^i), \\ \theta_3^i = f(x_i + \frac{h}{2}, y_i + \frac{h}{2} \theta_2^i), \\ \theta_4^i = f(x_i + h, y_i + h \theta_3^i), \\ y_{i+1} = y_i + \frac{h}{6} (\theta_1^i + 2\theta_2^i + 2\theta_3^i + \theta_4^i). \end{cases} \quad (14)$$

Численные примеры. Реализация в пакете Matlab

Пример 1. Найти решение задачи Коши дифференциального уравнения

$$\frac{dy}{dx} = x^2, \quad y(0) = 1.3 \quad (15)$$

методом Рунге-Кутты четвертого порядка.

Решение задачи Коши для дифференциальных уравнений методом Рунге-Кутты 4 порядка может быть реализовано в пакете Matlab в виде функции **ode 45()**. Описание других функций, реализующих метод Рунге-Кутты и другие методы, а также решатели систем дифференциальных уравнений можно найти, используя справочную систему Matlab.

Пример 2. Найти решение задачи (15), используя встроенные функции пакета Matlab.

Входными данными для функции `ode45` являются: имя файла, содержащего определение функции, стоящей в правой части уравнения (1), вектор, определяющий интервал интегрирования, начальное условие.

4. УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ

4.1. Лабораторный практикум

Лабораторная работа №1 «Основные приемы работы с пакетом Matlab»

Matlab – интерактивный матрично-ориентированный пакет, предназначенный для выполнения научных и инженерных расчетов. Пакет содержит обширную библиотеку программ по численным методам, использует двух- и трехмерную графику, а также форматы языков высокого уровня. Техника записи циклов, использование логических отношений, вызов подпрограмм и редактирование находят непосредственное применение в среде пакета Matlab.

Найти дополнительную информацию о командах, опциях и примерах можно, воспользовавшись справкой в диалоговом режиме, литературой и прилагаемой к пакету документацией.

Работа в командном окне

Вход в систему Matlab. Доступ к справочной информации

При работе в операционных системах Windows 9x, Windows 2000, Windows XP вход в пакет Matlab осуществляется простым кликом левой кнопки мыши по соответствующей иконке. В версиях пакета 5.0 и выше имеется собственный встроенный редактор текста. Командное окно пакета находится в одном окне, а редактор текста – в другом.

Существуют следующие способы получить информацию о функциях пакета Matlab в процессе работы.

1. Команда `help`

Команда `help <имя_группы>` выведет на экран список функций, размещенных в этой группе с краткими пояснениями. Например, команда

```
>> help elmat
```

выведет список функций, предназначенных для создания и работы с матрицами специального вида. Ввод команды с именем определенной функции выдаст на экран описание этой функции.

2. Команда **lookfor**

Эта команда позволяет выполнить поиск m-функции по ключевому слову, при этом анализируется первая строка комментария и она же выводится на экран, если в ней встретилось ключевое слово. Например, в системе Matlab нет m-функции с именем *inverse* и поэтому на команду

```
>> help inverse
```

ответом будет

inverse.m not found. Однако команда `lookfor inverse` покажет все совпадения, найденные в справочных файлах.

3. Меню **Help**.

4. Обращение к Web-серверу фирмы The Mathworks.

Арифметические операции. Встроенные скалярные функции. Операторы присваивания. Форматы вывода данных

Matlab можно использовать как обыкновенный калькулятор, который выполняет основные арифметические операции: + (сложение), – (вычитание), * (умножение), / (деление), ^ (возведение в степень) и содержит набор констант (*pi*, *e*, *i*).

Пример.

```
>> (2+3*pi)/2  
ans =  
5.7124
```

Для более сложных вычислений в пакете предусмотрены дополнительные операции. Кратко перечислим основные функции, имеющиеся в пакете. Описание других можно найти, используя справку в диалоговом режиме.

Функция	Описание	Функция	Описание
---------	----------	---------	----------

abs()	абсолютное значение числа	cos()	cos(x)
sin()	sin(x)	tan()	tg(x)
exp()	e^x	log()	ln(x) – натуральный логарифм
acos()	arccos(x)	sqrt()	\sqrt{x} – извлечение кв. корня
atan()	arctg(x)	round()	Округление числа до ближайшего целого
ceil()	Наибольшее целое для данного действительного числа	floor()	Наименьшее целое число для данного действительного числа
rem()	Остаток от деления двух чисел	sign()	Знак числа (1, если знак + и -1, если знак -)

Matlab является интерпретирующим языком непосредственных вычислений. Операторы пакета обычно имеют форму:

>> имя_переменной=выражение

или просто

>> выражение

Выражение формируется из операторов, функций и имен переменных. Если имя переменной и знак равенства в левой части отсутствуют, автоматически генерируется переменная **ans**, которой присваивается результат вычислений. Matlab различает строчные и прописные буквы в именах команд, функций и переменных!

Например, для присвоения выражениям различных имен используется знак равенства.

```
>> s=3-floor(exp(3.3))
s =
-24
```

Точка с запятой позволяет задать несколько выражений и вывод значений результата после нажатия клавиши <Enter> не производится:

```
>> d=sin(5.6);           Задание d
>> a=cos(6.5);         Задание a
>> a+d                 Вывод на экран a+d
ans =
0.3453
```


Поскольку все вычисления в Matlab выполняются с двоичной точностью, формат вывода может изменяться с помощью следующих команд:

Название формата	Описание формата представления числа
format short	С фиксированной точкой и 4 знаками после точки (по умолчанию)
format long	С фиксированной точкой и 14 знаками после точки
format short e	Научная нотация с 4 десятичными знаками
format long e	Научная нотация с 15 десятичными знаками
format short g	С фиксированной точкой и 3 знаками после точки
format long g	Научная нотация с 13 десятичными знаками

Например,

```
>> format long
>> 3*cos(1.2)
ans =
    1.08707326343002
```

Создание матриц. Операции с матрицами и массивами.

Векторные и матричные функции

Matlab работает с одним видом объектов – числовыми прямоугольными матрицами, элементами которых могут быть в общем случае комплексные числа. Все переменные представляют собой матрицы, матрицы 1×1 интерпретируются как скаляры, матрицы с одной строкой или одним столбцом – как вектора. В системе Matlab матрицы могут быть созданы разными способами:

1. введены явно с помощью списка элементов;
2. сгенерированы встроенными операторами или функциями;
3. созданы в m-файлах;
4. загружены из внешнего файла данных.

Например, в результате выполнения оператора

```
>> A=[1 2 3;3 2 1;4 5 6]
```

или

```
>> A=[1 2 3
      3 2 1
      4 5 6]
```

Matlab создает матрицу 3×3 и присваивает ее значение переменной A.

```
A =  
 1  2  3  
 3  2  1  
 4  5  6
```

В пакет Matlab встроен ряд функций, позволяющих создавать матрицы специального вида. Приведем некоторые из них:

1. функция **rand(n)** создает матрицу размера $n \times n$ (функция **rand(m,n)** – матрицу размера $m \times n$), каждый элемент которой – случайное число с равномерным законом распределения в диапазоне $[0,1]$;

2. функция **magic(n)** создает матрицу размера $n \times n$, которая является магическим квадратом;

3. функция **zeros(m,n)** создает нулевую матрицу размера $m \times n$;

4. функция **ones(m,n)** создает матрицу размера $m \times n$, каждый элемент которой равен единице;

5. функция **diag(x)** создает матрицу, у которой на главной диагонали стоят элементы вектора x;

6. функция **diag(A)** создает матрицу, у которой на главной диагонали стоят диагональные элементы матрицы A.

Ссылки на отдельные элементы матриц и векторов осуществляются с помощью индексов в круглых скобках.

Приведем несколько примеров задания матриц, выделения элементов и подматриц.

```
>> Z=zeros(4,5);           Создание нулевой матрицы размера 4x5  
>> O=ones(3,2);          Создание единичной матрицы размера  
3x2  
>> X=0:0.5:2             Вывод на экран вектора X размера 1x5  
X =  
 0  0.5000  1.0000  1.5000  2.0000  
>> Y=sin(X);            Создание вектора Y размера 1x5,  
каждый  
элемент которого является sin от X  
>> A(2,3)  
ans =  
 1  
Выделение элемента (2,3) из матрицы A
```

```
>> A(8)
```

```
ans =
```

```
6
```

```
1
```

```
>> A(1:2,1:3)
```

```
ans =
```

```
1 2 3
```

```
3 2 1
```

```
>> A([1 2],[1 3])
```

```
ans =
```

```
1 3
```

```
3 1
```

```
>> A(2,2)=round(tan(8))
```

```
ans =
```

```
1 2 3
```

```
3 -7 1
```

```
4 5 6
```

```
>> A(:,2)
```

```
ans =
```

```
2
```

```
-7
```

```
5
```

```
>> A(2,:)
```

```
ans =
```

```
3 -7 1
```

*Альтернативное выделение элемента(2,3)
как 8-го по счету, если записать матрицу*

один вектор-столбец

Выделение подматриц A

Замена элемента матрицы A

Выделение 2-го столбца

Выделение 2-й строки

В пакете Matlab доступны следующие матричные операции: + (сложение), - (вычитание), * (умножение), ^ (возведение в степень, применима только для квадратных матриц!), ' (транспонирование), / (правое деление), \ (левое деление). Если размерность матриц не соответствует выполняемой операции, то система генерирует сообщение об ошибке. Приведенные операции могут стать поэлементными, если перед ними поставить точку. Приведем примеры использования матричных операций.

```
>> D=[1 2;4 8];
```

```
>> C=D'
```

```
C =
```

```
1 4
```

```
2 8
```

```
>> 5*(D*C)^2
```

```
ans =
```

```
2125 8500
```

```
8500 3400
```

```
>> C^2
```

```
ans =
```

```
9 36
```

*Произведение матриц C*C*

```

18 72
>> C.^2
ans =
    1    16
    4    64
>> sin(D./2)
ans =
    0.4794    0.8415
    0.9093    0.7568

```

Квадрат каждого элемента матрицы C

Часто возникает необходимость использования векторных или матричных функций.

Аргументами векторных функций являются векторы (строки или столбцы). Если в качестве аргумента указана матрица размера $m \times n$, то данная функция действует постолбцово, т.е. результатом действия является вектор-столбец, каждый элемент которого – результат действия этой функции на соответствующий столбец. Например, максимальный элемент прямоугольной матрицы находится помощью команды **max(max(A))**. Наиболее употребительными матричными функциями являются:

- inv(x)** – обратная матрица,
- det(x)** – определитель матрицы,
- size(x)** – размерность матрицы,
- norm(x)** – норма вектора или матрицы,
- rank(x)** – ранг матрицы,
- cond(x)** – число обусловленности, ...

Циклы. Условные операторы и операторы отношения

Операторы управления Matlab и операторы отношения при использовании работают также, как и в большинстве языков программирования.

Оператор цикла **for** в общем виде записывается как:

```

for <переменная_цикла>=<выражение_цикла>
<Выполняемые_операторы>
end

```

Оператор цикла **while** записывается следующим образом:

```
while <условие>
<Выполняемые_операторы>
end
```

Условный оператор `if` имеет следующий синтаксис:

```
if <условие>
<Выполняемые_операторы1>
else
<Выполняемые_операторы2>
end
```

В пакете Matlab возможно также множественное ветвление.

В пакете Matlab используются следующие операторы отношений и логические операторы:

<code>==</code>	Равно	<code>~</code>	Не
<code>~=</code>	Не равно	<code>&</code>	И
<code><</code>	Меньше	<code> </code>	ИЛИ
<code>></code>	Больше		
<code><=</code>	Меньше или равно		
<code>>=</code>	Больше или равно		

Приведем несколько примеров использования циклов и условий.

Пример. Найдем значение $5!$.

```
>> a=1;
>> for i=1:5
a=a*i;
end
>> a
a =
    120
```

Пример. Для различных значений x , полученных делением на 3 чисел от 0 до 10, выведем значения квадратов и кубов этих чисел.

```
>> n=10;
>> k=0;
>> while k<=n
x=k/3;
disp([x x^2 x^3])
k=k+1;
end
```

Пример. В зависимости от s присвоим переменной a разные значения.

```
>> s=10;
```

```
>> if (s<0)|(s>10)
a=s^2+5;
else
a=0;
end
>> a
a =
0
```

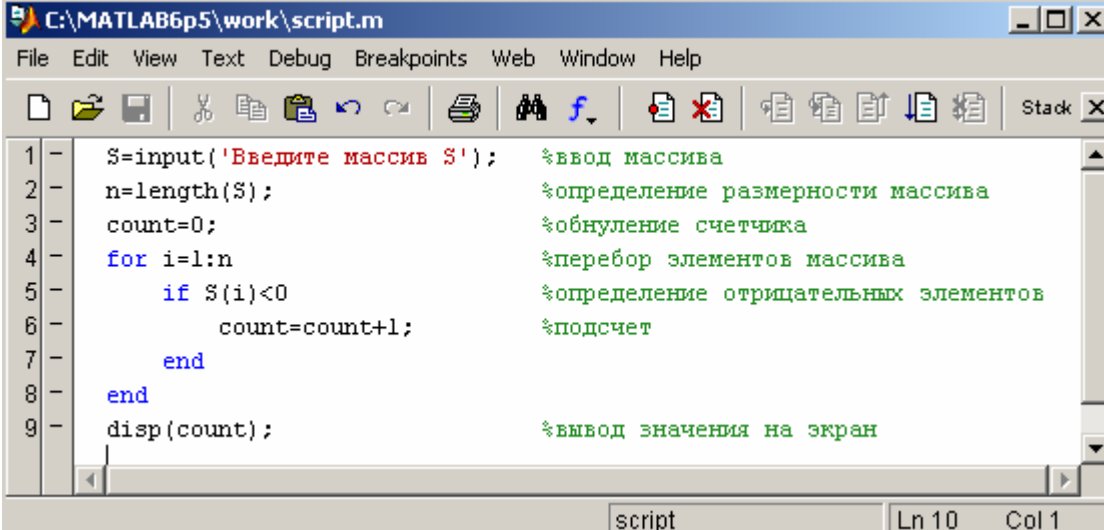
М – файлы

Matlab может последовательно выполнять последовательность операторов, записанных в файл на диске. Имена таких файлов имеют вид <имя>.m и такие файлы называются m-файлами. Большая часть работы в Matlab заключается в создании, редактировании и выполнении таких m-файлов. Существуют два типа m-файлов: файлы-программы (сценарии) и файлы-функции.

Файлы-программы

Файлы-программы состоят из последовательности обычных операторов пакета Matlab. Если m-файл с таким сценарием имеет имя, например, start.m, то команда start, введенная в командной строке, вызовет выполнение последовательности операторов этого файла. Переменные в программе являются глобальными и могут изменить значения переменных с теми же именами в командном окне. В таких файлах легко исправить ошибки, внутри файла можно ссылаться на другие m-файлы и рекурсивно на этот же файл.

Приведем пример программы, которая подсчитывает число отрицательных элементов массива, сохраненной под именем script.m.



```
C:\MATLAB6p5\work\script.m
File Edit View Text Debug Breakpoints Web Window Help
[Icons] Stack
1 - S=input('Введите массив S'); %ввод массива
2 - n=length(S); %определение размерности массива
3 - count=0; %обнуление счетчика
4 - for i=1:n %перебор элементов массива
5 -     if S(i)<0 %определение отрицательных элементов
6 -         count=count+1; %подсчет
7 -     end
8 - end
9 - disp(count); %вывод значения на экран
script Ln 10 Col 1
```

Вызов в командной строке

```
>> script
```

приведет к выполнению операторов программы:

```
>> script
```

```
Введите массив S [2 -7 4 -8 2 -56]
```

```
3
```

Файлы-функции

Файлы-функции дают возможность расширять Matlab, поскольку определенные пользователем функции имеют тот же статус, что и другие функции Matlab. Переменные по умолчанию являются локальными и не влияют на имена и значения переменных в текущей рабочей области Matlab.

Общий вид задания:

```
function [x1,x2,...]=name(y1,y2,...)
```

```
<тело функции>
```

Первая строка функции содержит объявление выходных аргументов, имени функции и входные аргументов, без этой строки файл является программой, но не функцией.

Совпадение имени функции и имени файла является обязательным условием в Matlab.

Символ % указывает на то, что вся строка после него является комментарием и игнорируется при выполнении программы.

Приведем пример функции, которая производит подсчет количества элементов массива, кратных заданному числу и вычисляет сумму таких элементов.

Вызов в командной строке (после задания массива P):

```
>> [m,S]=fun(P,3)
```

```
m =
```

```
2
```

```
S =
```

```
12
```

присвоит переменным m и S значения выходных параметров.

```
1 function [m1,S]=fun(P,q) %Задание функции
2 n=length(P); %Определение размерности массива P
3 m1=0; %Обнуление счетчиков
4 S=0;
5 for i=1:n
6     if rem(P(i),q)==0 %Поиск кратных числу q элементов
7         m1=m1+1; %Подсчет количества таких элементов
8         S=S+P(i); %Подсчет их суммы
9     end
10 end
```

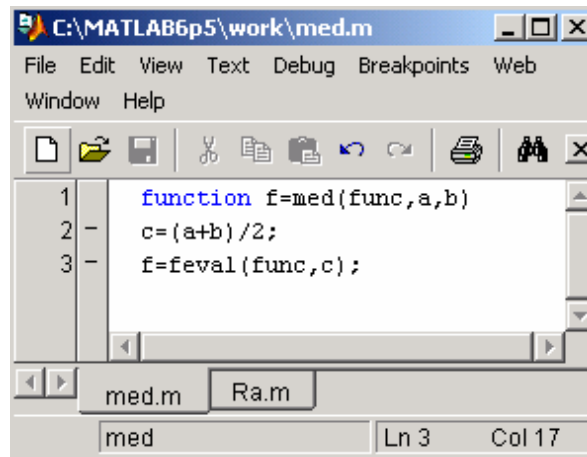
Для вычисления значения функции, имя которой передается в функцию через строковую переменную, часто используется функция **feval**, аргументами которой являются имя функции и координата точки, в которой вычисляется значение данной функции.

Приведем пример использования функции **feval**.

Зададим свою функцию Ra изменений аргумента.

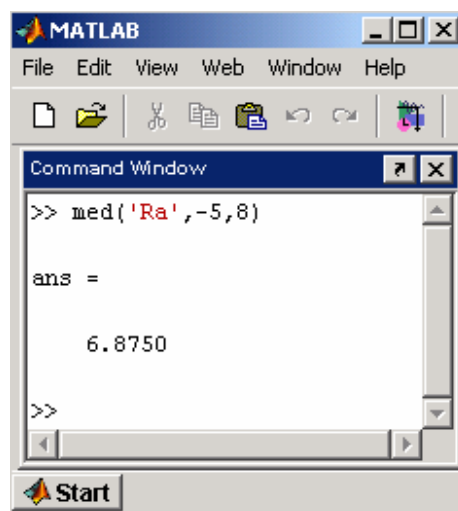
```
1 function z=Ra(x)
2 z=x^3-2*x+x+5;
```

И пусть, например, требуется обратиться к этой функции **med** в другом файле-функции, вычисляющем значение функции в середине указанного отрезка:



```
C:\MATLAB6p5\work\med.m
File Edit View Text Debug Breakpoints Web
Window Help
function f=med(func,a,b)
c=(a+b)/2;
f=feval(func,c);
med.m Ra.m
med Ln 3 Col 17
```

Результат обращения к функции med будет иметь вид:



```
MATLAB
File Edit View Web Window Help
Command Window
>> med('Ra',-5,8)
ans =
6.8750
>>
```

Текстовые строки. Сообщения об ошибках

Текстовые строки вводятся в Matlab в виде текста в одинарных кавычках: `m='Mistake'`. Вывод текстовой строки осуществляется при помощи оператора **disp**. Оператор `disp(m)` выведет на экран сообщение:

Mistake

Сообщение об ошибках лучше выводить при помощи функции `error`, так как после обращения к данной функции выполнение m-файла будет прекращено. Для выхода из цикла (и из программы) можно воспользоваться командой **break**.

В m-файле запрос на ввод данных можно организовать при помощи оператора **input**. При вводе оператора `m=input('введите размерность массива')` на экран выводится запрос и выполнение работы файла

приостанавливается до ввода требуемых данных с клавиатуры. После нажатия клавиши <Enter> данные присваиваются переменной *m*.

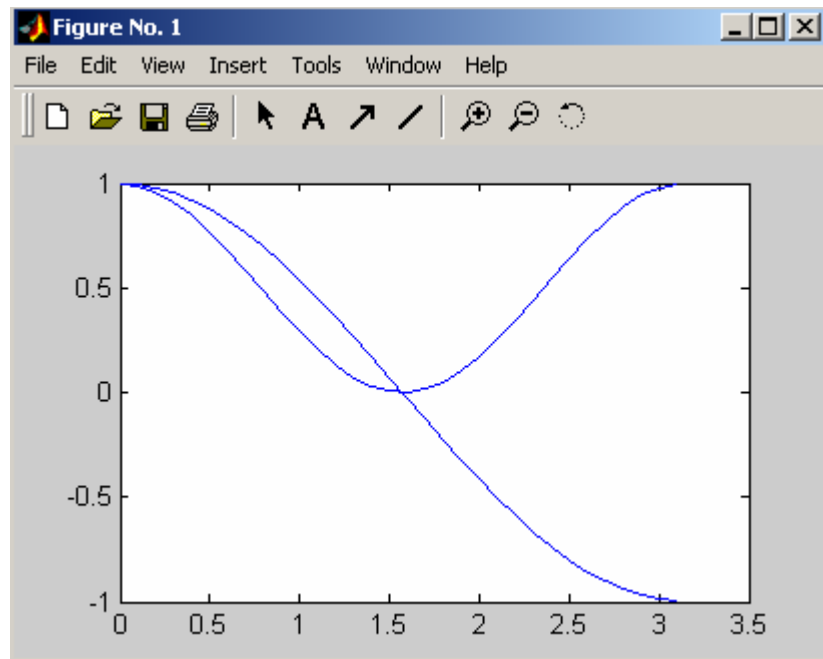
Графики

Пакет Matlab позволяет строить двух- и трехмерные графики кривых и поверхностей. Дополнительные возможности и описание графиков можно найти, используя справку в диалоговом режиме.

Команду `plot` используют для построения графика двумерной функции. Следующий пример иллюстрирует, как построить графики функций $y = \cos(x)$ и $y = \cos^2(x)$ на интервале $[0, \pi]$.

```
>> x=0:0.1:pi;  
>> y=cos(x);  
>> z=cos(x).^2;  
>> plot(x,y,x,z,'b')
```

В первой строке программы задается область с шагом 0.1. В следующих двух строках задаются две функции. Первые три строки заканчиваются точкой с запятой – это предотвращает вывод матриц *x*, *y*, *z* на экран. Команда `plot` строит графики сплошной линией синего цвета.



Лабораторная работа №2 «Решение нелинейных уравнений»

Контрольные вопросы

1. Поясните геометрический смысл методов: половинного деления, секущих, касательных.
2. Назовите основную сущность итерационных методов.
3. Почему в методе касательных начальное приближение $x_0 \in [a, b]$ целесообразно выбирать из условия $f(x_0) \cdot f''(x_0) > 0$?
4. Чему равны порядки сходимости рассмотренных методов?
5. Поясните, почему метод секущих можно считать частным случаем метода Ньютона, а метод Ньютона – частным случаем метода Чебышева?

Индивидуальные задания

1. Используя один из методов, найти область существования решения предложенного уравнения.
2. Найти корень уравнения $f(x) = 0$ в найденной области с точностью $\varepsilon = 10^{-4}$ указанными методами в соответствии с номером варианта. Реализовать алгоритмы указанных методов.
3. Определить число итераций каждого метода и оценить погрешности решения.
4. Выполнить проверку, используя возможности пакета Matlab. Сравнить решения, полученные всеми способами. Сделать вывод.
5. Оформить отчет по лабораторной работе.

Номер варианта	Уравнение $f(x) = 0$	Численные методы
1	$x^3 + 0.3 \cdot x^2 + 1.8 \cdot x - 15 = 0$	Метод Ньютона Метод Чебышева
2	$x^3 + 0.2 \cdot x^2 + 5 \cdot x + 4 = 0$	Метод половинного деления Двушаговый метод Ньютона
3	$x^3 + 0.7 \cdot x^2 + 4.7 \cdot x - 1.5 = 0$	Метод секущих Метод Чебышева
4	$x^3 + 11 \cdot x^2 - 8 \cdot x - 25 = 0$	Упрощенный метод Ньютона

		Метод секущих
5	$x^3 + 3 \cdot x^2 + 3 \cdot x + 4 = 0$	Метод половинного деления Метод Стеффенсена
6	$x^3 - 2 \cdot x - 0.4 = 0$	Метод половинного деления Метод Чебышева
7	$x^3 + x^2 + 3 \cdot x - 13 = 0$	Разностный метод Ньютона Метод секущих
8	$x^3 + x^2 + 3.8 \cdot x - 2 = 0$	Метод половинного деления Двухшаговый метод Ньютона
9	$x^3 + 3.5x^2 + 12 \cdot x + 19 = 0$	Упрощенный метод Ньютона Метод Чебышева
10	$x^3 + 6x^2 - 17 \cdot x - 21 = 0$	Метод Ньютона-Шредера Метод хорд

Лабораторная работа №3

«Решение систем линейных алгебраических уравнений»

Контрольные вопросы

1. В чем заключается основное преимущество метода Гаусса с выбором главного элемента?
2. В чем состоит особенность реализации метода прогонки?
3. Какие существуют способы приведения исходной матрицы к виду, удобному для решения методом простой итерации?
4. Каким образом может быть выбран вектор начальных приближений?
5. Назовите достаточное условие сходимости метода простой итерации.
6. В чем заключается преимущество метода Зейделя при программировании?

Индивидуальные задания

1. Реализовать алгоритм метода Гаусса с выбором главного элемента (по столбцу или по строке) в пакете Matlab и найти решение предложенной системы уравнений $AX = B$. Значения матрицы A и вектора B приведены в таблице.
2. Привести систему уравнений к виду, пригодному для итерационного процесса Зейделя, реализовать метод в пакете Matlab и решить систему

уравнений с точностью $\varepsilon = 10^{-3}$ и исходными данными, приведенными в табл.

Номер варианта	Значения матрицы A и вектора B
1	$A = \begin{pmatrix} 2.1 & -4.5 & -2 \\ 3 & 2.5 & 4.3 \\ -6 & 3.5 & 2.5 \end{pmatrix}, B = \begin{pmatrix} 19.07 \\ 3.21 \\ -18.25 \end{pmatrix}$
2	$A = \begin{pmatrix} -6.7 & 5.9 & 4.1 & -9.1 \\ 5.4 & -8.7 & 11.2 & -0.5 \\ 2.7 & -4.9 & 6.8 & 3.3 \\ 1.6 & 7.2 & 8.3 & -7.9 \end{pmatrix}, B = \begin{pmatrix} 7.12 \\ 58.07 \\ 47.1 \\ 36.8 \end{pmatrix}$
3	$A = \begin{pmatrix} 3.81 & 0.25 & 1.28 & 0.75 \\ 2.25 & 1.32 & 4.58 & 0.49 \\ 5.31 & 6.28 & 0.98 & 1.04 \\ 9.39 & 2.45 & 3.35 & 2.28 \end{pmatrix}, B = \begin{pmatrix} 4.21 \\ 6.47 \\ 2.38 \\ 10.48 \end{pmatrix}$
4	$A = \begin{pmatrix} 2.34 & -4.21 & -11.61 \\ 8.04 & 5.22 & 0.27 \\ 3.92 & -7.99 & 8.37 \end{pmatrix}, B = \begin{pmatrix} 14.41 \\ -6.44 \\ 55.56 \end{pmatrix}$
5	$A = \begin{pmatrix} 0.18 & 0.25 & -0.44 \\ 0.42 & -0.35 & 1.12 \\ 1.14 & 0.12 & -0.83 \end{pmatrix}, B = \begin{pmatrix} 1.15 \\ 0.86 \\ 0.68 \end{pmatrix}$
6	$A = \begin{pmatrix} 21.54 & -95.5 & -96.12 \\ 10.22 & -91.06 & -7.34 \\ 51.21 & 12.29 & 86.45 \end{pmatrix}, B = \begin{pmatrix} -49.93 \\ -12.46 \\ 60.81 \end{pmatrix}$
7	$A = \begin{pmatrix} 2.51 & -3.12 & 4.64 \\ -3.25 & 2.62 & 1.85 \\ -6.53 & -3.5 & 7.31 \end{pmatrix}, b = \begin{pmatrix} -1.05 \\ -14.46 \\ -17.73 \end{pmatrix}$
8	

	$A = \begin{pmatrix} 8.2 & 1.4 & -2.3 & 0.2 \\ -1.6 & 5.4 & -7.7 & 3.1 \\ 0.7 & 1.9 & -8.5 & 4.8 \\ 5.3 & -5.9 & 2.7 & -7.9 \end{pmatrix}, b = \begin{pmatrix} 32.76 \\ 54.39 \\ 59.18 \\ -71.95 \end{pmatrix}$
9	$A = \begin{pmatrix} 4.21 & 22.42 & 3.85 \\ 2.31 & 31.49 & 1.52 \\ 3.49 & 4.85 & 28.72 \end{pmatrix}, B = \begin{pmatrix} 30.24 \\ 4.095 \\ 42.81 \end{pmatrix}$
10	$A = \begin{pmatrix} 0.63 & 0.07 & 0.2 \\ 0.05 & 0.34 & 0.3 \\ 0.15 & 0.1 & 0.71 \end{pmatrix}, B = \begin{pmatrix} 0.34 \\ 0.32 \\ 0.42 \end{pmatrix}$

3. Найти решение этой же системы, используя возможности пакета Matlab. Сравните найденное решение с полученным при реализации метода Гаусса и метода Зейделя. За сколько шагов достигается заданная точность в каждом из методов?

4. Методом прогонки найти вектор $(u_1, u_2, \dots, u_{10})$, являющийся решением системы уравнений:

$$a_k u_{k-1} + b_k u_k + c_k u_{k+1} = f_k, \quad k = 1, 2, \dots, 10; \quad a_1 = c_{10} = 0$$

Коэффициенты $a_k, k = 2, 3, \dots, 10, b_k, k = 1, 2, \dots, 10, c_k, k = 1, 2, \dots, 9$ и $f_k, k = 1, 2, \dots, 10$ задаются следующей таблицей, где N – номер соответствующего варианта.

a_k	b_k	c_k	f_k
$\frac{4}{k \cdot N}$	$\frac{30}{k \cdot N}$	$\frac{6}{k \cdot N}$	$\frac{10}{k \cdot N}$

Что можно сказать о сходимости метода прогонки в вашем случае?

5. Оформить отчет по лабораторной работе.

Лабораторная работа №4 «Решение систем нелинейных уравнений»

Контрольные вопросы

1. Поясните геометрический смысл методов спуска.
2. Каким образом выбираются начальные приближения в рассмотренных методах?
3. Как выбор начального приближения влияет на сходимость метода Ньютона?
4. Почему предложенные методы являются итерационными?
5. Что произойдет, если в окрестности решения нелинейной системы функция будет иметь несколько минимумов?

Индивидуальные задания

1. Найти решение системы уравнений с точностью $\varepsilon = 10^{-3}$ указанными методами. Для определения начального приближения использовать графический способ локализации корней.
2. Найти решение системы нелинейных уравнений, используя возможности пакета Matlab.
3. Определить число итераций, оценить погрешности решения, сравнить используемые методы.
4. Оформить отчет по лабораторной работе.

Номер варианта	Система уравнений	Метод
1	$\begin{cases} 2x^3 - y^2 - 1 = 0, \\ x \cdot y^3 - y - 4 = 0. \end{cases}$	Метод градиентного спуска Метод простых итераций
2	$\begin{cases} e^{x \cdot y} - x^2 + y - 1.06 = 0, \\ (x + 0.5)^2 + y^2 - 0.6 = 0, \end{cases} \quad x > 0, \quad y > 0.$	Метод покоординатных итераций Метод градиентного спуска
3	$\begin{cases} y - 4 + \sqrt{1 - (x - 2.7)^2} = 0, \\ x + 2 \sin(y - 5) - 1 = 0. \end{cases}$	Метод градиентного спуска

		Метод Ньютона
4	$\begin{cases} (x-y)^3 - 8 \cdot (x+y) = 0, \\ 2 \cdot (x+y) + 15 \cdot \ln(x+y) - 5 = 0. \end{cases}$	Метод градиентного спуска Модифицированный метод Ньютона
5	$\begin{cases} x^2 + y^2 + z^2 - 1 = 0, \\ 2x^2 + y^2 - 4z = 0, \\ 3x^2 - 4y + z^2 - 1 = 0. \end{cases}$	Метод градиентного спуска Рекурсивный метод Ньютона
6	$\begin{cases} \operatorname{tg}(x \cdot y + 0.1) - x^2 = 0, \\ 0.6x^2 + 2y^2 - 1 = 0, \end{cases} \quad x > 0, y > 0.$	Метод градиентного спуска Разностный метод Ньютона
7	$\begin{cases} \cos(y + 1.5) - x - 0.4 = 0, \\ \sin(1.1 \cdot x) - 3y - 1 = 0. \end{cases}$	Метод простых итераций Метод Брауна
8	$\begin{cases} x^2 \cdot y^2 - 3x^3 - 6y^3 + 8 = 0, \\ x^4 - 9y + 2 = 0. \end{cases}$	Метод покоординатных итераций Метод Брауна
9	$\begin{cases} 0.8x^2 + 2x \cdot y + 1.3y^2 + 20x - 15y = 0, \\ e^{0.6y - 0.8x} - 1.14x - 1.52y = 0. \end{cases}$	Метод Ньютона Метод Брауна
10	$\begin{cases} 2y - \cos(x + 0.9) - 0.2 = 0, \\ x + \sin(y - 0.5) + 0.6 = 0. \end{cases}$	Разностный метод Ньютона Метод покоординатных итераций

Лабораторная работа №5 «Интерполирование функций»

Контрольные вопросы

1. Какие точки называются узлами интерполяции?
2. При решении каких задач используются интерполяционные формулы Ньютона?
3. Какие узлы называются равноотстоящими?
4. В каких случаях применяется сплайн-интерполяция?

5. В чем заключается основное отличие одномерной интерполяции от кубической?

Индивидуальные задания

1. Пользуясь первой и второй интерполяционными формулами Ньютона, вычислить значение функции для указанных значений аргумента.

2. Решить задачу интерполяции средствами пакета Matlab с помощью линейных и кубических сплайнов. Привести графики исходных данных, результатов сплайн-интерполяции и погрешности аппроксимации.

3. Оформить отчет по лабораторной работе.

Номер варианта	x	$f(x)$	Значения аргумента	Номер варианта	x	$f(x)$	Значения аргумента
1	1.50	0.51183	$x = 1.50911$ $x = 1.59513$	6	0.50	1.6487	$x = 0.50721$ $x = 0.56894$
	1.51	0.50624			0.51	1.6653	
	1.52	0.50064			0.52	1.6820	
	1.53	0.49503			0.53	1.6989	
	1.54	0.48940			0.54	1.7160	
	1.55	0.48376			0.55	1.7333	
	1.56	0.47811			0.56	1.7507	
	1.57	0.47245			0.57	1.7683	
	1.58	0.46678			0.58	1.7860	
	1.59	0.46110			0.59	1.8040	
1.60	0.45540	0.60	1.8221				
2	0.00	0.28081	$x = 0.01928$ $x = 0.47113$	7	1.1	0.89121	$x = 1.1511$ $x = 2.0316$
	0.05	0.31270			1.2	0.93204	
	0.10	0.34549			1.3	0.96356	
	0.15	0.37904			1.4	0.98545	
	0.20	0.41318			1.5	0.99749	
	0.25	0.44774			1.6	0.99957	
	0.30	0.48255			1.7	0.99166	
	0.35	0.51745			1.8	0.97385	
	0.40	0.55226			1.9	0.94630	
	0.45	0.58682			2.0	0.90930	
0.50	0.62096	2.1	0.86321				
3	1.0	0.5652	$x = 1.0113$ $x = 1.9592$	8	0.10	3.63004	$x = 0.1006$ $x = 2.5304$
	1.1	0.6375			0.35	3.75680	
	1.2	0.7147			0.60	3.88933	
	1.3	0.7973			0.85	4.03258	

	1.4	0.8861			1.10	4.19310		
	1.5	0.9817			1.35	4.38042		
	1.6	1.0848			1.60	4.60963		
	1.7	1.1964			1.85	4.90697		
	1.8	1.3172			2.10	5.32331		
	1.9	1.4482			2.35	5.97322		
	2.0	1.5906			2.60	7.18210		
4	0.50	1.6487	$x =$ 0.52301 $x =$ 0.58967	9	1.50	0.51183	$x =$ 1.50253 $x =$ 1.59614	
	0.51	1.6653				1.51		0.50624
	0.52	1.6820				1.52		0.50064
	0.53	1.6989				1.53		0.49503
	0.54	1.7160				1.54		0.48940
	0.55	1.7333				1.55		0.48376
	0.56	1.7507				1.56		0.47811
	0.57	1.7683				1.57		0.47245
	0.58	1.7860				1.58		0.46678
	0.59	1.8040				1.59		0.46110
	0.60	1.8221				1.60		0.45540
5	0.10	3.63004	$x =$ 0.10056 $x =$ 2.57321	10	0.00	0.28081	$x =$ 0.02475 $x =$ 0.48675	
	0.35	3.75680				0.05		0.31270
	0.60	3.88933				0.10		0.34549
	0.85	4.03258				0.15		0.37904
	1.10	4.19310				0.20		0.41318
	1.35	4.38042				0.25		0.44774
	1.60	4.60963				0.30		0.48255
	1.85	4.90697				0.35		0.51745
	2.10	5.32331				0.40		0.55226
	2.35	5.97322				0.45		0.58682
	2.60	7.18210				0.50		0.62096

Лабораторная работа №6 «Обработка экспериментальных данных»

Контрольные вопросы

1. Как можно добиться повышения качества приближения?
2. Какая ошибка является среднеквадратической?
3. За счет чего возникает полиномиальное раскачивание?

Индивидуальные задания

1. Осуществив замену переменных, получите линейризованную форму для каждой из приведенных функций. С помощью метода наименьших квадратов найти значения параметров приближающей линейной функции. Построить графики исходных данных и полученной аппроксимирующей функции.

2. Найти аппроксимирующую функцию средствами пакета Matlab. Сравнить с результатом, полученным методом наименьших квадратов. Рассчитать значение среднеквадратического отклонения.

3. Оформить отчет по лабораторной работе.

Номер варианта	Функция $y = f(x)$	Исходные данные		Номер варианта	Функция $y = f(x)$	Исходные данные	
		x	y			x	y
1	$y = c \cdot e^{ax}$	0	1.5	6	$y = a \cdot \ln(x) + b$	1.5	-0.72
		1	2.5			2.5	0.33
		2	3.5			3.5	1.09
		3	5			4	1.28
		4	7.5			5	1.85
2	$y = c \cdot x^a$	1	0.6	7	$y = \frac{x}{ax + b}$	1	0.03
		2	1.9			2	0.54
		3	4.3			3	0.28
		4	7.6			4	0.55
		5	12.6			5	0.78
3	$y = \frac{1}{ax + b}$	-1	6.62	8	$y = \frac{c \cdot x}{e^{dx}}$	1.6	3.36
		0	3.94			1.8	3.35
		1	2.17			2.1	3.19
		2	1.35			2.3	2.79
		3	0.89			2.9	2.67
4	$y = \frac{1}{(ax + b)^2}$	-1	13.45	9	$y = \frac{d}{x + c}$	2.4	1.30
		0	3.01			2.9	1.14
		1	0.67			3.1	1.24
		2	0.15			3.6	1.16
5	$y = \frac{a}{x} + b$	1	5.51	10	$y = \frac{1}{1 + c \cdot e^{ax}}$	0.1	0.90
		1.5	4.72			0.4	0.71
		1.9	4.54			0.7	0.95
		2.2	4.33			1.3	0.92
		2.5	4.02			1.9	1.14

Лабораторная работа №7

«Численное дифференцирование и интегрирование»

Контрольные вопросы

1. Каким образом можно повысить точность численного дифференцирования?
2. В чем основные преимущества формулы трапеций по отношению к методу прямоугольников?
3. Как выбирается шаг интегрирования?
4. Чему равен порядок погрешности формулы Симпсона для двумерной подынтегральной функции?
5. Какие условия обязательно должны выполняться в методе трех восьмых и почему?

Индивидуальные задания

1. Дана таблица значений функции $y = f(x)$.

x	y	x	y	x	y
1.0	1.2661	1.6	1.7500	2.2	2.6291
1.1	1.3262	1.7	1.8640	2.3	2.8296
1.2	1.3937	1.8	1.9896	2.4	3.0493
1.3	1.4693	1.9	2.1277	2.5	3.2898
1.4	1.5534	2.0	2.2796	2.6	3.5533
1.5	1.6467	2.1	2.4463	2.7	3.8417
				2.8	4.1573

Найдите: а) значение первой производной функции в точке $x = 0.8 + i \cdot 0.2$, где i – номер вашего варианта; б) значение второй производной функции в точке $x = 2.6 - i \cdot 0.1$, где i – номер вашего варианта.

2. С помощью интерполяционных формул Ньютона найдите значение первой и второй производных функции, заданной таблично в пункте 4 вашего варианта, в указанных точках.

Номер варианта	x	Номер варианта	x
1	$x = 1.50, x = 1.51,$ $x = 1.59, x = 1.60.$	6	$x = 0.50, x = 0.51,$ $x = 0.59, x = 0.60.$

2	$x = 0.00, x = 0.05,$ $x = 0.45, x = 0.50.$	7	$x = 1.1, x = 1.2,$ $x = 2.0, x = 2.1.$
3	$x = 1.0, x = 1.1,$ $x = 1.9, x = 2.0.$	8	$x = 0.10, x = 0.35,$ $x = 2.35, x = 2.60.$
4	$x = 0.50, x = 0.51,$ $x = 0.59, x = 0.60.$	9	$x = 1.50, x = 1.51,$ $x = 1.59, x = 1.60.$
5	$x = 0.10, x = 0.35,$ $x = 2.35, x = 2.60.$	10	$x = 0.00, x = 0.05,$ $x = 0.45, x = 0.50.$

3. Вычислить приближенное значение интеграла по формулам трапеций и трех восьмых, разбивая отрезок интегрирования на 9 одинаковых частей. Вычислить определенный интеграл с помощью функции quad(). Сравнить полученные значения интеграла с точным.

4. Оформить отчет по лабораторной работе.

Номер варианта	Интеграл	Номер варианта	Интеграл
1	$\int_1^e \frac{\ln^2(x)}{x} dx$	6	$\int_1^2 \frac{dx}{x^2 + x}$
2	$\int_0^2 \sqrt{4 - x^2} dx$	7	$\int_1^{e^{\pi/2}} \cos(\ln(x)) dx$
3	$\int_0^1 \frac{x}{1 + x^4} dx$	8	$\int_1^3 x^3 \sqrt{x^2 - 1} dx$
4	$\int_0^{\frac{\pi}{6}} \frac{\sin^2(x)}{\cos(x)} dx$	9	$\int_{\ln(2)}^{2\ln(2)} \frac{dx}{e^x - 1}$
5	$\int_{-1}^1 x \cdot \arctan(x) dx$	10	$\int_0^1 \frac{2 \cdot \arcsin(x)}{\sqrt{1 - x^2}} dx$

Лабораторная работа №8 «Численные методы решения обыкновенных дифференциальных уравнений»

Контрольные вопросы

1. В чем заключается основная особенность метода Пикара?

2. Поясните геометрический смысл метода Эйлера для решения обыкновенных дифференциальных уравнений.

3. Как определяется порядок точности рассмотренных методов Эйлера и Рунге-Кутты?

Индивидуальные задания

1. Применяя указанные методы, реализовать алгоритм численного решения дифференциального уравнения с начальным условием на отрезке $[a, b]$ с произвольно заданным шагом.

2. Решить предложенную задачу Коши, используя встроенные функции пакета Matlab.

3. Оформить отчет по лабораторной работе.

Номер варианта	Задача Коши	Отрезок $[a, b]$	Методы
1	$y' = 2x - 3y, \quad y(0) = 1$	$[0, 10]$	Метод Рунге-Кутты 2-го порядка Исправленный метод Эйлера
2	$y' = 0.2y^2 + \frac{0.8}{x^2}, \quad y(1) = 1$	$[1, 2]$	Метод Эйлера-Коши Метод Пикара
3	$y' = \frac{0.2}{x^2} - \frac{y}{x} - 0.8y^2, \quad y(1) = 0.5$	$[1, 2]$	Метод Пикара Метод Рунге-Кутты 4-го порядка
4	$y' = 1.6y^2 + \frac{0.4}{x^2}, \quad y(1) = 1$	$[1, 2]$	Метод Эйлера Метод Рунге-Кутты 4-го порядка
5	$y' = \frac{0.9}{x^2} - \frac{y}{x} - 1.6y^2, \quad y(1) = 0.75$	$[1, 2]$	Метод Эйлера Метод Рунге-Кутты 2-го порядка

6	$y' = 0.9y^2 + \frac{1.6}{x^2}, y(1) = 1$	[1,2]	Исправленный метод Эйлера Метод Пикара
7	$y' = xy^2 + 1, y(0) = 0$	[0,1]	Метод Эйлера Метод Рунге-Кутты 4-го порядка
8	$y' = y^2 + \frac{1}{x^2}, y(1) = 1$	[1,2]	Метод Пикара Метод Эйлера
9	$y' = y^2 + x^2, y(0) = 0$	[0,1]	Исправленный метод Эйлера Метод Рунге-Кутты 4-го порядка
10	$y' = 0.25y^2 + \frac{2}{x^2}, y(1) = 1$	[1,2]	Метод Рунге-Кутты 2-го порядка Метод Пикара

4.2. Перечень обязательной (основной) литературы

1. Вержбицкий В. М. Численные методы. (Линейная алгебра и нелинейные уравнения) – М.: Высшая школа, 2001. – 266 с.
2. Вержбицкий В. М. Численные методы. (Математический анализ и обыкновенные дифференциальные уравнения) – М.: Высшая школа, 2001. – 266 с.
3. Бахвалов И.В., Жидков Н. П., Кобельков Г.М. Численные методы. – М.: Лаборатория базовых знаний, 2000.– 624с.
4. Березин И.С., Жидков Н.П. Методы вычислений. – М.: Гл. ред. Физ. – мат. лит., 1991.– 521 с.
5. Демидович Б.П., Марон И.А. Основы вычислительной математики. – М.: Наука, 1992. – 402 с.
6. Копченова Н.В., Марон И.А. Вычислительная математика в примерах и задачах. – М.: Наука, 1992.– 363 с.
7. Боглаев Ю.А. Вычислительная математика и программирование. – М.: Высшая школа, 1994. – 366 с.

8. Турчак Л.И. Основы численных методов. – М.: Наука, 1992. – 310 с.
9. Волков Е.А. Численные методы. – М.: Наука, ФИЗМАТЛИТ, 1997. – 248 с.
10. Самарский А.А. Введение в численные методы: Учеб. Пособие. – М.: Наука, 1998. – 345 с.

4.3. Перечень дополнительной литературы

1. Поршнева С.В. Вычислительная математика. – СПб.: БХВ-Петербург, 2004.– 320 с.
2. Мэтьюз Д.Г., Куртис Ф.Д. Численные методы. Использование Matlab. Пер. с англ. – М.: Изд. дом «Вильямс», 2001. – 720 с.
3. Иглин С.П. Математические расчеты на базе Matlab. – СПб.: БХВ-Петербург, 2005.– 640 с.
4. Рашиков В.И., Рошаль А.С. Численные методы решения физических задач: Учеб пособие. – СПб.: Изд-во «Лань», 2005.– 208 с.
5. Косарев В.И. 12 лекций по вычислительной математике. – М.: Физматкнига, Изд-во МФТИ, 2000. – 221 с.
6. Демидович Б.П., Марон И.А., Шувалова Э.З. Численные методы анализа. – М.: Наука, 1997. – 368 с.
7. Карманов В.Г. Математическое программирование. – М.: Наука, ФИЗМАТЛИТ, 2001. – 259 с.
8. Плохотников К.Э. Математическое моделирование и вычислительный эксперимент. – М.: Едиториал УРСС, 2003. – 280 с.
9. Matlab Дьяконов В. Mathcad 8/2000. – СПб: Питер, 2001. – 592 с.
10. Потемкин В.Г. Система инженерных и научных расчетов. Matlab 5x – в 2-х т. – М.: Диалог МИФИ. 1999. – 670 с.
11. Потемкин В.Г. Инструментальные средства Matlab 5x – М.: Диалог МИФИ, 2001. – 324 с.
12. Кетков Ю., Кетков А., Шульц М. Matlab 6.x: программирование численных методов – СПб: БХВ – Петербург, 2004. – 660 с.

1	2	3	4	5	6	7	8	9
1	1	1-2		1	лекция, доп. литература		2	отчет по лабораторной работе №1
2	2	1-4		1	лекция	индивид. задание №1	4	отчет по индивид. заданию №1
3	3	1-3		2	лекция		2	
4	4	1-3		2	лекция			отчет по лабораторной работе №2
5	4	4-5		3	лекция	индивид. задание №2	3	отчет по индивид. заданию №2
6	5	1-3		3	лекция		2	отчет по лабораторной работе №3
7	6	1-3		4	лекция		2	
8	6	4-6		4	лекция			отчет по лабораторной работе №4
9	6	7-9		5	лекция	индивид. задание №3	3	отчет по индивид. заданию №3
10	7	1-3		5	лекция		2	отчет по лабораторной работе №5
11	8	1-3		6	лекция			
12	8	4-6		6	лекция		2	отчет по лабораторной работе №6
13	9	1-3		7	лекция			
14	10	1-2		7	лекция		2	отчет по лабораторной работе №7
15	11	1-2		8	лекция		2	
16	12	1-2		8	лекция			отчет по лабораторной работе №8
17	12	3-4		9	лекция			
18	12	5-6		9	лекция		2	отчет по лабораторной работе №9,

								устный и письменный опрос, зачет
--	--	--	--	--	--	--	--	----------------------------------------

Примечание

Индивидуальное задание №1 – «Приближенные вычисления и оценка погрешностей».

Индивидуальное задание №2 – «Нахождение собственных чисел и собственных векторов матриц по методу Крылова».

Индивидуальное задание №3 – «Приближение функций».