

**С.А. Кострюков В.В. Пешков
Г.Е. Шунин В.А. Шунина**

ПРАКТИКУМ ПО ЧИСЛЕННЫМ МЕТОДАМ

Учебное пособие

Воронеж 2017

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО «Воронежский государственный
технический университет»

С.А. Кострюков В.В. Пешков
Г.Е. Шунин В.А. Шунина

ПРАКТИКУМ ПО ЧИСЛЕННЫМ МЕТОДАМ

Утверждено учебно-методическим советом
университета в качестве учебного пособия

Воронеж 2017

УДК 519.6+681.3.06(075.8)

ББК 22.19я7

П 69

Практикум по численным методам: учеб. пособие [Электронный ресурс]. – Электрон. текстовые, граф. данные (4,2 Мб) / С.А. Кострюков, В.В. Пешков, Г.Е. Шунин, В.А. Шунина. – Воронеж: ФГБОУ ВО «Воронежский государственный технический университет», 2017. – 1 электрон. опт. диск (CD-ROM): цв. – Систем. требования: ПК 500 МГц и выше; 512 Мб ОЗУ; Windows XP; SVGA с разрешением 1024x768; Adobe Acrobat; CD-ROM; мышь. – Загл. с экрана.

Пособие содержит краткие теоретические сведения о численных методах решения прикладных математических задач, а также большое число примеров и упражнений, ориентированных на применение систем компьютерной математики. Значительное место отведено методам аппроксимации решений дифференциальных уравнений.

Издание соответствует требованиям Федерального государственного образовательного стандарта высшего образования по направлению 16.03.01 «Техническая физика», направленность «Физическая электроника», по дисциплине «Численные методы» и по направлению 14.03.01 «Ядерная энергетика и теплофизика», направленность «Техника и физика низких температур» по дисциплине «Математические методы моделирования физических процессов».

Табл. 8. Ил. 36. Библиогр.: 23 назв.

Рецензенты: кафедра цифровых технологий

Воронежского государственного университета
(зав. кафедрой д-р физ.-мат. наук,
проф. С.Д. Кургалин);

д-р физ.-мат. наук, проф. В.Н. Нечаев

© Кострюков С.А., Пешков В.В., Шунин Г.Е.,
Шунина В.А., 2017

© ФГБОУ ВО «Воронежский государственный
технический университет», 2017

ВВЕДЕНИЕ

С точки зрения математического моделирования решение физико-технической задачи сводится к решению математической задачи, являющейся ее моделью. Для решения математических задач в основном используются аналитические, графические и численные методы.

Аналитические методы позволяют представить решение в виде формул. С помощью графических методов решение получается в виде графических построений. Численные методы сводят решение математической задачи к выполнению конечного числа арифметических операций над числами, и результаты при этом представляются в виде чисел. Именно спомощью численных методов решается подавляющее число современных сложных задач физико-математического моделирования. Основную роль в этом сыграло появление высокопроизводительных компьютеров, способных выполнять миллиарды, триллионы и более операций в секунду.

В настоящее время имеется значительное число программных продуктов, ориентированных на решение задач вычислительной математики (MathCad, Maple, Mathematica, и т.д.) Конечно, умение пользоваться готовыми программами сокращает время и затраты на решение важных практических задач. С другой стороны, для их использования зачастую достаточно лишь задания входных данных без вникания в сущность методов и алгоритмов, заложенных в программу. Как правило, выбор метода, проведение расчетов, выдачу результатов программа берет на себя. Однако при таком подходе полученное решение носит обычно приближенный характер, поскольку каждая модель и каждый метод имеют существенные ограничения по применимости. Кроме того, незнание метода не дает возможности проанализировать решение (оценить погрешность решения, скорость сходимости, устойчивость и др.) Таким образом, грамотный специалист должен иметь представление об основах математического моделирования, численных методов, о возможностях современных программных

средств, уметь анализировать полученные результаты с точки зрения их точности и достоверности.

В данном учебном пособии рассмотрены основные численные алгоритмы решения математических задач, таких как решение линейных и нелинейных уравнений, интегрирование и дифференцирование, интерполирование, точечная и непрерывная аппроксимации функций, решение дифференциальных уравнений.

Как известно, многие задачи моделирования физических процессов сводятся к решению дифференциальных уравнений. Причем аналитически может быть решен только ограниченный класс таких уравнений. Традиционный подход численного решения связан с переходом к конечным разностям. Однако более эффективными методами решения, в частности, краевых задач, в настоящее время признаются методы, базирующиеся на аппроксимации базисными функциями. Поэтому этим методам в пособии уделено особое внимание. Подробно рассмотрены методы: вариационный (метод Ритца), проекционный (метод Галеркина), вариационно-разностный (метод конечных элементов). Для них даны четкие критерии сходимости аппроксимации.

Особенностью пособия является его практическая направленность. Большинство примеров разбираются в пошаговом режиме с использованием одной из популярных систем компьютерной математики – Maple. Такой подход способствует лучшему усвоению методов, не отвлекаясь на рутинные вычисления.

Учебное пособие может быть использовано преподавателями, аспирантами и студентами физико-технических специальностей вузов.

1. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И СИСТЕМ УРАВНЕНИЙ

1.1. Понятие о линейных и нелинейных уравнениях

Рассмотрим уравнение вида

$$g(x) = b, \quad (1.1)$$

в котором $g(x)$ – некоторая функция, x является неизвестной переменной, а b – некоторое число. Решением (корнем) этого уравнения называется всякое число x из области определения $g(x)$, которое, будучи подставлено в уравнение (1.1), обращает его в тождество. Под термином «решение уравнения» также понимают процесс нахождения (вычисления) его корней. Решить уравнение – значит найти совокупность (множество) всех его корней. Иногда задача ставится о решении уравнения лишь на каком-либо подмножестве числовой оси x (например, на отрезке), тогда ищется совокупность корней уравнения, принадлежащих этому подмножеству.

Пусть x , b – векторы размерности n , тогда приходим к системе уравнений

$$\begin{aligned} g_1(x_1, x_2, \dots, x_n) &= b_1, \\ g_2(x_1, x_2, \dots, x_n) &= b_2, \\ &\dots\dots\dots \\ g_n(x_1, x_2, \dots, x_n) &= b_n, \end{aligned}$$

или, в векторной записи,

$$\mathbf{g}(\mathbf{x}) = \mathbf{b}. \quad (1.2)$$

В этом случае решением является не число, а числовой вектор $\mathbf{x}^* = (x_1, x_2, \dots, x_n)^*$.

Линейные уравнения вида (1.1) или (1.2) характеризуются важным свойством: функция $g(x)$ (или вектор-функция $\mathbf{g}(\mathbf{x})$) удовлетворяет принципу суперпозиции, состоящему в том, что если аргумент x есть линейная комбинация (суперпозиция) двух точек

$$x = c_1 x^{(1)} + c_2 x^{(2)},$$

где c_1 и c_2 – константы, то значение $g(x)$ есть линейная суперпозиция значений $g(x^{(1)})$ и $g(x^{(2)})$, а именно

$$g(x) = g(c_1 x^{(1)} + c_2 x^{(2)}) = c_1 g(x^{(1)}) + c_2 g(x^{(2)}). \quad (1.3)$$

Легко проверить выполнение этого принципа для функций $g(x) = ax$ ($a = \text{const}$) и $\mathbf{g}(\mathbf{x}) = \mathbf{A}\mathbf{x}$, где $\mathbf{A}$ – матрица размером $n \times n$.

Нелинейные функции не удовлетворяют принципу суперпозиции, например, для $g = x^2$ (1.3) не выполняется:

$$(c_1 x_1 + c_2 x_2)^2 \neq c_1 x_1^2 + c_2 x_2^2$$

Этот факт лежит в основе различия линейных или нелинейных уравнений.

Линейные уравнения могут иметь:

- 1) единственное решение;
- 2) бесконечное множество решений;
- 3) не иметь решений вовсе.

Линейные уравнения являются частным случаем нелинейных, поэтому для нелинейных уравнений возможны упомянутые выше три варианта, а также

- 4) имеется n решений, $2 \leq n < \infty$.

Примером четвертого варианта могут служить задачи определения корней полиномов $P_n(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n$.

Нелинейные уравнения можно разделить на два класса – алгебраические и трансцендентные. Алгебраическими называются уравнения, содержащие только алгебраические функции (целые, рациональные, иррациональные). В частности, многочлен является целой алгебраической функцией. Уравнения, содержащие другие функции (тригонометрические, показательные, логарифмические и др.), называются трансцендентными.

Нелинейные уравнения принято записывать в виде

$$f(x) = 0, \quad (1.4)$$

который получается из (1.1) путем замены $f(x) \equiv g(x) - b$.

1.2. О методах решения нелинейных уравнений

Методы решения нелинейных уравнений делятся на прямые и итерационные. Прямые методы позволяют записать корни в виде некоторого конечного соотношения (формулы). Из школьного курса алгебры читателю известны такие методы для решения тригонометрических, логарифмических, показательных, а также простейших алгебраических уравнений.

Однако встречающиеся на практике уравнения часто не удастся решить такими простыми методами. Для их решения используются итерационные методы, т.е. методы последовательных приближений. Алгоритм нахождения корня нелинейного уравнения с помощью итерационного метода состоит из двух этапов: а) отыскания приближенного значения корня (начального приближения); б) уточнения приближенного значения до некоторой заданной степени точности. В некоторых методах отыскивается не начальное приближение, а некоторый отрезок, содержащий корень.

Общая схема итерационных методов состоит в последовательном уточнении начального приближения x_0 . Каждый такой шаг называется итерацией. В результате итераций находится последовательность приближенных значений корня $x_1, x_2, \dots, x_k$. Если эти значения с ростом k стремятся к истинному значению корня

$$\lim_{k \rightarrow \infty} x_k = x^*,$$

то говорят, что итерационный процесс сходится.

Прежде чем приступить к вычислительной процедуре решения уравнения, определяют отрезки числовой оси, содержащие по одному корню данного уравнения, или, как говорят, отделяют корни. Для отделения корней уравнения можно использовать критерий: если на отрезке $[a, b]$ функция $f(x)$ непрерывна и монотонна, а ее значения на концах отрезка имеют разные знаки, то на этом отрезке существует и притом один корень данного уравнения. Достаточным признаком монотонности функции $f(x)$ на отрезке $[a, b]$ является сохранение знака

производной функции. При отделении корней стараются определить отрезок $[a, b]$ как можно меньшей длины.

Отделение корней можно выполнить также графически. Найти корень уравнения (1.4) – это значит найти абсциссу точки пересечения графика функции $y=f(x)$ с прямой $y=0$ (осью абсцисс). Если построить график функции затруднительно, то уравнение (4) следует представить в эквивалентном виде

$$f_1(x) = f_2(x) \quad (1.5)$$

с таким расчетом, чтобы графики функций $f_1(x)$ и $f_2(x)$ строились проще. Корни же уравнения (1.5) определяются как абсциссы точек пересечения графиков функций $y = f_1(x)$ и $y = f_2(x)$. Если уравнения (1.4) и (1.5) эквивалентны, то, отделив корни уравнения (1.5), мы тем самым отделяем и корни уравнения (1.4).

Выбор алгоритма построения последовательности $\{x_k\}$, или метода решения, – весьма важный момент при практическом решении задачи. Здесь немалую роль играют такие свойства алгоритма, как простота, надежность, экономичность и т.д.

Одной из характеристик вычислительного алгоритма является его скорость сходимости. Последовательность $\{x_k\}$, сходящаяся к пределу x^* ($x^* \neq x_1$), имеет порядок сходимости $\alpha \geq 1$, если существуют числа $c \geq 0$, $k_0 > 0$, $\alpha \geq 1$ такие, что для всех $k \geq k_0$

$$|x_{k+1} - x^*| \leq c |x_k - x^*|^\alpha.$$

Сходимость при $\alpha = 1$ называется линейной, или 1-го порядка, при $\alpha = 2$ – квадратичной и т.д. С ростом α вычислительный алгоритм построения последовательности $\{x_k\}$ усложняется, и условия, обеспечивающие сходимость последовательности $\{x_k\}$, становятся более жесткими.

На практике чаще всего используются следующие методы решения нелинейных уравнений: половинного деления, секущих (хорд), простой итерации, парабол, Ньютона.

1.3. Решение нелинейных уравнений

1.3.1. Метод деления отрезка пополам

Простейшим и надежным алгоритмом уточнения корня является метод деления отрезка пополам (или *бисекции*). Пусть на отрезке $[a, b]$ функция $f(x)$ непрерывна и на концах этого отрезка имеет разные знаки, т.е. $f(a)f(b) < 0$. Очевидно, что середина отрезка служит приближением к корню уравнения (1.4) с точностью $\delta \leq (b-a)/2$. В середине отрезка $x_1 = (a+b)/2$ определяется знак функции $f(x)$, затем выбирается та половина отрезка, на концах которой функция $f(x)$ принимает значения разных знаков, и деление повторяется. Если требуется найти корень с точностью ε , то деление отрезка пополам продолжается до тех пор, пока длина отрезка не станет меньше 2ε . Тогда середина последнего отрезка даст значение корня с требуемой точностью.

Алгоритм метода:

1. Найти $x = (a + b)/2$.
2. Вычислить $f(x)$.
3. Если $f(x) \cdot f(a) > 0$, то принять $a = x$, иначе $b = x$.
4. Проверить условие $|b - a| > 2\varepsilon$, если оно выполняется, перейти к пункту 1, если оно не выполняется, закончить вычисления, считая корнем x с точностью ε .

Метод деления отрезка пополам имеет линейную скорость сходимости.

Пример. Решить уравнение $0.5^x + 2 = (x-2)^2$.

Решение в системе компьютерной математики Maple

Документ пакета Maple рекомендуется начинать с команды очистки оперативной памяти:

```
> restart;
```

1) Задаем функцию $f(x) = 0.5^x + 2 - (x-2)^2$:

```
> f:=t->0.5^t+2-(t-2)^2;
```

2) Отделяем корни, для этого строим график функции $f(x)$. Поскольку заранее неизвестно, на каком промежутке (или

промежутках) расположены корни уравнения (нули функции $f(x)$), график функции сначала строим на достаточно большом отрезке, например на отрезке $[-20, 20]$, а затем локализуем его для лучшей визуализации корней.

```
> plot(f(t), t=-20..20) ;
```

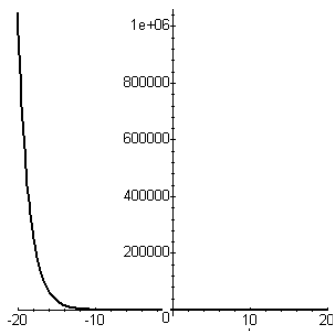


Рис. 1.1

Из рис. 1.1 видно, что функция $f(x)$ при отрицательных значениях аргумента резко возрастает, и, значит, корни уравнения могут находиться примерно при $x > -10$. Строим новый график на более узком интервале (рис. 1.2).

```
> plot(f(t), t=-10..20) ;
```

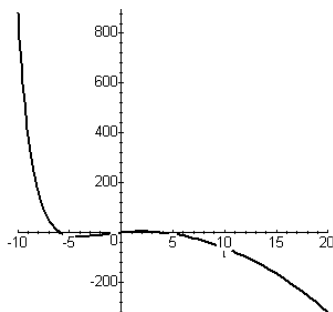


Рис. 1.2

Чтобы еще точнее локализовать корни, строим график на интервале от -6.5 до 5 (рис. 1.3):

```
> plot(f(t), t=-6.5..5) ;
```

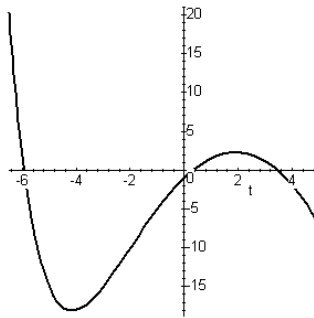


Рис. 1.3

Видим, что график пересекает ось x в трех точках, т.е. имеем три корня. Сначала найдем первый корень, лежащий на отрезке $[-7, -5]$.

3) Присвоим значения переменным:

```
> eps:=0.001: a:=-7.0: b:=-5.0: n:=0:
```

(**eps** – точность, **a** и **b** – границы отрезка, содержащего корень, **n** – переменная для подсчета числа итераций).

4) Ниже реализован основной алгоритм метода в виде цикла.

```
> while (b-a)>2*eps do
  x:=(a+b)/2;
  if evalf(f(x)*f(a))>0 then a:=x else b:=x fi;
  n:=n+1;
od:
```

5) Вывод результата (**x** – корень, **n** – число итераций):

```
> evalf(x); n;
```

–5.926818850

15

Итак, первый корень равен -5.927 , для его нахождения потребовалось 15 итераций. Чтобы найти второй корень, расположенный вблизи нуля, нужно выполнить шаги 3–5, придав переменным **a** и **b** соответственно значения -1 и 1 . В результате получим числа 0.3275756837 и 15 . Аналогично ищется третий корень, лежащий между $a=3$ и $b=4$. В этом случае Maple выдаёт: 3.446228029 и 14 . Сведем в ответе все найденные ре-

шения, записывая для корней по 4 цифры после запятой – на единицу больше, чем используемая точность 10^{-3} .

Ответ: корни уравнения $x_1 = -5.9268$; $x_2 = 0.3276$; $x_3 = 3.4462$.

1.3.2. Метод простой итерации

Метод простой итерации применяется к решению нелинейного уравнения с выделенным линейным членом вида

$$x = \varphi(x) \quad (1.6)$$

и состоит в построении последовательности $\{x_n\}$, начиная с некоторого заданного начального значения x_0 по правилу

$$x_{n+1} = \varphi(x_n). \quad (1.7)$$

Если $\varphi(x)$ имеет на отрезке $S = \{x: |x - x_0| < \delta\}$ непрерывную производную, то достаточным условием сходимости метода простой итерации является выполнение неравенства

$$|\varphi'(x)| < 1, \quad x \in S. \quad (1.8)$$

Если исходное уравнение имеет вид (1.4), то для применения метода предварительно следует преобразовать его к эквивалентному виду (1.6), например, следующим образом. Умножим уравнение $f(x) = 0$ на некоторую постоянную λ и сложим с тождеством $x = x$. Получим эквивалентное уравнение $x = x + \lambda f(x)$, обозначим правую часть $\varphi(x) = x + \lambda f(x)$. Подбором λ добиваемся, чтобы $\varphi(x)$ в окрестности корня удовлетворяла условию (1.8).

Метод простой итерации относится к классу методов с линейной скоростью сходимости.

Программа реализации метода простой итерации:

```
> restart; # 1) Очистка оперативной памяти
> f:=t->0.5^t+2-(t-2)^2; # 2) Задание функции f из левой
  части уравнения f(x)=0
> phi:=x+lambda*f(x); u:=diff(phi,x); # 3) Построение
  функции phi(x) и ее производной
> x0:=0.5; # 4) Задание начального приближения x0
```

```

> r:=subs(x=x0,u): evalf("# 5) Вычисление функции
φ'(x) в точке x0
> solve(abs(r)<1,lambda); # 6) Решение неравенства (1.8)
относительно параметра λ
RealRange(Open(-1.209609323), Open(0))

> v:=-0.5: # 7) значение λ из полученного диапазона
> eps:=0.0001: n:=0: # 8) Присвоение начальных значений
> while abs(f(x0))>eps do # 9) Цикл “пока |f(x)|>ε ...”
    x0:=subs(lambda=v,x=x0,phi); # 10) Подстановка в
φ(x) значений λ и x_n
    n:=n+1; # 11) Счетчик цикла
od: # 12) Окончание цикла
> evalf(x0); n; # 13) Вывод результата и числа итераций
.327586329
9

```

З а м е ч а н и я . 1) Начальное приближение x_0 не должно слишком сильно отличаться от точного значения корня. В противном случае может произойти заикливание, даже несмотря на правильный подбор параметра λ . В рассмотренной программе взято значение $x_0=0.5$, чтобы найти корень, который графически определяется в окрестности 0.3. Чтобы найти другой корень, например, лежащий между -7 и -5 (см. рис. 1.3), можно положить $x_0=-6$ и затем выполнить все шаги программы после строки 4 «Задание начального приближения x_0 ».

2) Диапазон изменения λ выводится после выполнения команды 6. Для успешной работы программы необходимо взять какое-то число из этого диапазона (лучше где-то посередине) и присвоить его λ (переменная $\mathbf{v}$) в строке 7. В приведенном фрагменте указанный диапазон получился $(-1.209609323, 0)$. В качестве λ выбрано -0.5 . При изменении x_0 параметр λ должен быть пересчитан. К примеру, если взять $x_0=-6$, получим интервал для λ $(0; 0.07051833198)$, и, взяв число 0.03 из этого интервала, следует в строке 7 вместо $\mathbf{v:=-0.5}$ записать $\mathbf{v:=0.03}$.

3) В ряде случаев данная программа может не работать из-за того, что Maple не преобразует должным образом вызовы функции $\mathbf{f}(\mathbf{x})$, заданной во второй строке, в числа с плавающей точкой. Для решения проблемы следует в строке 3 вместо $\mathbf{f}(\mathbf{x})$ вставить $\mathbf{evalf}(\mathbf{f}(\mathbf{x}))$, а в строке 9 $\mathbf{abs}(\mathbf{f}(\mathbf{x}_0))$ поменять на $\mathbf{abs}(\mathbf{evalf}(\mathbf{f}(\mathbf{x}_0)))$.

1.3.3. Метод Ньютона

Метод Ньютона применяется к уравнению $f(x)=0$, где $f(x)$ – непрерывно дифференцируемая функция. Для начала вычислений требуется задание одного начального приближения x_0 . Последующие приближения вычисляются по формуле

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad f'(x_n) \neq 0, \quad n = 0, 1, 2, \dots \quad (1.9)$$

Геометрически x_{n+1} является значением абсциссы точки пересечения касательной к кривой $y=f(x)$ в точке $(x_n, f(x_n))$ с осью Ox , поэтому метод Ньютона называют также методом касательных. Условия сходимости метода определяет теорема Канторовича: пусть на отрезке $S = \{x: |x-x_0| < \delta\}$ функция дважды непрерывно дифференцируема и выполняются неравенства

$$f'(x_0) \neq 0, \quad |[f(x_0)]^{-1}| \leq B; \quad \left| \frac{f(x_0)}{f'(x_0)} \right| \leq \eta;$$

$$|f''(x)| \leq K \quad \forall x \in S; \quad h = B\eta K \leq 1/2; \quad \frac{1 - \sqrt{1 - 2h}}{h} \eta \leq \delta,$$

тогда на отрезке S существует решение x^* уравнения $f(x)=0$, к которому сходится последовательность $\{x_n\}$, $n=0, 1, 2, \dots$

На практике можно встретиться как со случаем сходимости метода Ньютона, когда x_0 далеко от искомого корня, так и со случаем расходимости метода для x_0 , близких к корню. Возможен также и случай заикливания метода.

Сходимость метода Ньютона квадратичная (при условии $f'(x^*) \neq 0$). Когда $f'(x^*)=0$, наблюдается замедление скорости сходимости.

Программа решения уравнения $0,5^x + 2 - (x - 2)^2 = 0$ на языке Maple может быть составлена следующим образом:

```
> restart; # 1) Очистка оперативной памяти
> f:=t->0.5^t+2-(t-2)^2; # 2) Задание функции f из урав-
нения f(x)=0
> fd:=t->D(f)(t); # 3) Задание производной функции f
> x0:=0.: # 4) Задание начального приближения x0
> n:=0: eps:=0.0001: # 5) Инициализация переменных (n –
счетчик итераций, eps входит в условие окончания счета |f(xn)| ≤ ε)
> while abs(f(x0))>eps do # 6) Цикл “пока |f(x)| > ε, вы-
полнять ...”
    x:=x0-f(x0)/fd(x0); # 7) Новое приближение по фор-
муле (1.9)
    r:=evalf(x-x0); # 8) Разность двух последних прибли-
жений
    x0:=x; n:=n+1; # 9) Изменение переменных
    if n=50 then break fi; # 10) Защита от заикливания
od: # 11) Конец цикла
> evalf(x); # 12) Вывод приближенного значения корня x_{n+1}
.3276217483
> n; abs(r); # 13) Вывод числа итераций и |x_{n+1} - x_n|
3
.0001810073
```

Замечания. 1) Начальное приближение x_0 (строка 4) задаётся пользователем исходя из априорной информации о поведении функции (например, из графика). Для нахождения каждого корня необходимо вводить свое значение x_0 . От удачного выбора x_0 зависит успех метода.

2) Если после строки 13 выдается число итераций n , равное 50, выданный результат для корня нельзя считать правильным, так как в этом случае, скорее всего, алгоритм не получил своего естественного завершения и произошло заикливание. Для исправления ситуации, убедившись сначала в правильности текста программы, попробуйте изменить начальное приближение x_0 .

3) Параметр ε (**eps**), который фигурирует в условии окончания счета, влияет на точность вычисления корня: чем он меньше, тем точнее результат. Вместо условия окончания счета $|f(x_n)| \leq \varepsilon$ может использоваться условие $|x_{n+1} - x_n| \leq \varepsilon$, которое означает, что столько-то цифр после запятой в значении корня не меняются с увеличением n , и это значение можно принять за ответ. В этом случае строка 6 примет вид

```
> while r>eps do
```

(Переменная **r** до входа в цикл должна быть проинициализирована.)

4) Для некоторых функций $f(x)$ Maple может отказаться выполнять данную программу. Например, пусть решается уравнение $e^x + \sin x = 0$, при этом строка 2 запишется в виде

```
> f:=t->exp(t)+sin(t);
```

После запуска цикла (строки 6–11) появится сообщение об ошибке

```
Error, cannot evaluate boolean
```

и счет будет остановлен. Дело в том, что Maple в первую очередь пытается вычисления проводить в символьном виде. Для него, скажем, $\sin(1)$ гораздо более точное число, чем значение $\sin 1 = 0,8414709848\dots$ в виде десятичной дроби с любым конечным числом значащих цифр. Поэтому во внутренних расчетах Maple по возможности не переводит подобные объекты в числовой формат. Метод же Ньютона – это численный метод, он ориентирован на действия с числами. Отсюда возникает потенциальный источник ошибок. Решить эту проблему можно с помощью функции **evalf**, т.е. явно указывая на операции в форме с плавающей запятой. Изменения должны коснуться всех вызовов функций $f(x)$ и $f'(x)$, а именно в шестой и седьмой строках:

```
> while abs(evalf(f(x0)))>eps do
  x:=x0-evalf(f(x0)/fd(x0));
```

Кстати, если функцию $f(x) = e^x + \sin x$ заявить с использованием чисел в форме десятичных дробей, в частности,

```
> f:=t->exp(1.*t)+sin(1.*t);
```

то Maple это воспримет как указание на все вычисления функции исключительно в числовом формате (с использованием арифметики чисел с плавающей точкой) и надобность в использовании Maple-функции **evalf** отпадёт. По этой же причине данная программа работает для исходной функции в строке 2 – там фигурирует десятичная дробь 0.5.

1.3.4. Метод секущих

В методе Ньютона на каждом шаге нужно вычислять значения функции и производной. На практике часто применяются методы, требующие вычисления лишь значений функции. Заменяв производную $f'(x_n)$ в методе Ньютона разделенной разностью по двум точкам x_n и $x_n + h_n$, получим итерационную формулу

$$x_{n+1} = x_n - \frac{h_n f(x_n)}{f(x_n + h_n) - f(x_n)}, \quad n = 0, 1, 2, \dots \quad (1.10)$$

Формула (10) называется методом секущих (хорд) для решения уравнения вида $f(x) = 0$.

Приближение является абсциссой точки пересечения секущей прямой, проведенной через точки $(x_n, f(x_n))$ и $(x_n + h_n, f(x_n + h_n))$, с осью Ox . Часто берут $h_n = x_{n-1} - x_n$, тогда формула (10) принимает вид

$$x_{n+1} = x_n - \frac{(x_n - x_{n-1})f(x_n)}{f(x_n) - f(x_{n-1})}, \quad n = 0, 1, 2, \dots \quad (1.11)$$

Ниже приведена программа, осуществляющая вычисление корня уравнения $0.5^x + 2 - (x-2)^2 = 0$ в окрестности точки $x_0 = 0.5$ с использованием формулы (1.11).

```
> restart;
> f:=t->0.5^t+2-(t-2)^2;
> x0:=0.5: eps:=0.0001: h:=eps: n:=0:
> x:=x0-h*f(x0)/(f(x0+h)-f(x0));
> while abs(f(x))>eps and abs(x-x0)>eps do
    h:=x-x0; x0:=x;
```

```

      x:=x0-h*f(x0)/(f(x0)-f(x0-h));
      n:=n+1;
> od:
> evalf(x); n; evalf(x-x0);
.3276232087
2
.0005139694

```

Если известны числа a и b , такие, что $f(a)f(b) < 0$, т.е. заранее известно, что на отрезке $[a, b]$ существует корень уравнения, то целесообразно использовать алгоритм

1. Найти $x_0 = (a + b)/2$.

2. Вычислить $x = x_0 - \frac{(b-a)f(x_0)}{f(b)-f(a)}$.

3. Если $f(x) \cdot f(a) > 0$, то $a = x$, иначе $b = x$.

4. Проверить условие $|f(x)| > \varepsilon$, если оно выполняется, перейти к пункту 1, если оно не выполняется, закончить вычисления, считая корнем x с точностью, меньшей, чем $|b-a|$.

Программа, реализующая этот алгоритм, может быть следующей:

```

> restart;
> f:=t->0.5^t+2-(t-2)^2;
> eps:=0.0001: a:=-1.: b:=1.: n:=0: x:=(a+b)/2:
> while abs(f(x))>eps do
  x:=(b-a)*f(x)/(f(b)-f(a));
  if evalf(f(x)*f(a))>0 then a:=x else b:=x fi;
  n:=n+1;
od:
> evalf(x); n;

```

```

.3276503180
3

```

Замечание. Как и ранее, для корректных численных вычислений, в двух последних программах может потребоваться все вызовы функции $f(\dots)$ осуществлять через $\text{evalf}(f(\dots))$.

1.4. Решение систем нелинейных уравнений. Метод Ньютона

Для системы второго порядка

$$\begin{aligned}f(x, y) &= 0, \\g(x, y) &= 0\end{aligned}$$

последовательные приближения по методу Ньютона вычисляются по формулам

$$x_{n+1} = x_n - \frac{A_n}{J_n}, \quad y_{n+1} = y_n - \frac{B_n}{J_n},$$

где

$$\begin{aligned}A_n &= \begin{vmatrix} f(x_n, y_n) & f'_y(x_n, y_n) \\ g(x_n, y_n) & g'_y(x_n, y_n) \end{vmatrix}, \quad B_n = \begin{vmatrix} f'_x(x_n, y_n) & f(x_n, y_n) \\ g'_x(x_n, y_n) & g(x_n, y_n) \end{vmatrix}, \\J_n &= \begin{vmatrix} f'_x(x_n, y_n) & f'_y(x_n, y_n) \\ g'_x(x_n, y_n) & g'_y(x_n, y_n) \end{vmatrix}.\end{aligned}$$

Метод Ньютона сходится, если начальное приближение (x_0, y_0) выбрано удачно и матрица Якоби невырожденная, причём сходимость квадратичная. На практике итерации обычно заканчивают, когда одновременно достаточно малы значения $|f(x_n, y_n)|$ и $|g(x_n, y_n)|$ или разности $|x_{n+1} - x_n|$ и $|y_{n+1} - y_n|$. Для выбора начального приближения применяют графический метод, метод проб и т.д.

Пример. Решить систему уравнений

$$\begin{cases} x^7 - 5x^2y^4 + 1510 = 0, \\ y^5 - 3x^4y - 105 = 0. \end{cases} \quad (1.12)$$

Решение в системе Maple:

1) Очистим оперативную память. Нам в дальнейшем потребуются функции **det** и **implicitplot**, которых нет в ядре системы, поэтому нужно подключить пакеты линейной алгебры **linalg** и графических построений **plots**.

```
> restart;  
> with(linalg): with(plots):
```

2) Зададим функции $f(x, y)$ и $g(x, y)$.

```
> f:=(x,y)->evalf(x^7-5*x^2*y^4+1510);
```

```
> g:=(x,y)->evalf(y^5-3*x^4*y-105);
```

3) Для отделения корней системы построим графики неявных функций $f(x, y) = 0$ и $g(x, y) = 0$.

```
> implicitplot({f(x,y)=0,g(x,y)=0},x=-5..5,y=-5..5);
```

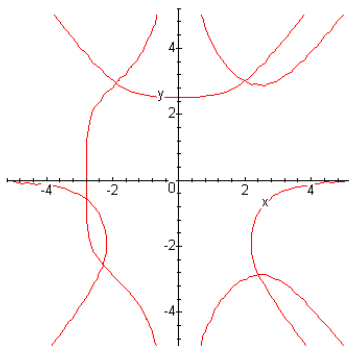


Рис. 1.4

Видим пять пересечений графиков функций, значит, система имеет, как минимум, пять решений.

4) Определим якобиан

```
> J:=[[diff(f(x,y),x),diff(f(x,y),y)],  
[diff(g(x,y),x),diff(g(x,y),y)]];
```

5) Введём функцию $Jn(x, y)$, являющуюся определителем матрицы J .

```
> Jn:=unapply(det(J),x,y);
```

6) Определим величины A_n и B_n в виде соответствующих определителей, при этом для исключения символьных преобразований результат оформляем через **evalf**

```
> An:=det([[f(x,y),diff(f(x,y),y)],  
[g(x,y),diff(g(x,y),y)]]);  
> Bn:=det([[diff(f(x,y),x),f(x,y)],  
[diff(g(x,y),x),g(x,y)]]);
```

7) Присвоим начальные значения переменным. x_0 и y_0 задают точку начального приближения (определяется графически).

```
> x0:=-3.; y0:=-1.; eps:=0.001: n:=0:
```

8) Организуем цикл вычислений по методу Ньютона

```
> while
abs(evalf(f(x0,y0)))+abs(evalf(g(x0,y0)))>eps do
  if abs(evalf(Jn(x0,y0)))<10^(-4) then
    print(`Якобиан = 0`); break fi; # Если якоби-
ан = 0, выйти из программы
    x0:=evalf(x0-subs(x=x0,y=y0,An)/Jn(x0,y0));
    y0:=evalf(y0-subs(x=x0,y=y0,Bn)/Jn(x0,y0));
    if n=50 then break fi; # Защита от заикливания
    n:=n+1;
  od:
```

9) Вывод результата: (x_0, y_0) – решение системы, n – число итераций

```
> evalf({x0,y0});n;
      {-2.844483289, -5348543094}
      4
```

10) Для вычисления остальных четырех корней системы нужно выполнить шаги 7–9, предварительно изменив в п. 7 начальное приближение. В частности, установив

```
> x0:=-2.; y0:=-3.;
```

получим второе решение системы:

```
{-2.573256593, -2.304767678}
      4
```

Третье, четвертое и пятое решение получим с начальными приближениями а) $x_0:=2., y_0:=-3.$, б) $x_0:=-2., y_0:=3.$, в) $x_0:=2., y_0:=3.$ соответственно. Эти значения следует последовательно подставить в п. 7 и выполнить все последующие команды программы.

В завершение сведем все найденные решения в ответе.

Ответ: $(-2,844; -0,535)$, $(-2,573, -2,305)$, $(2,430; -2,873)$, $(-1,922, 2,957)$, $(1,994; 3,012)$.

Из других методов решения систем нелинейных уравнений имеют применение метод простой итерации, метод наискорейшего спуска, градиентные методы.

1.5. Использование стандартных функций системы Maple

Для получения численного решения нелинейного уравнения или системы уравнений удобно использовать Maple-функцию **fsolve**(eqns, vars, options);

Здесь **eqn**, **eqn1**, ... – уравнения, содержащие неизвестные переменные **var**, **var1**, ... Эта функция может быть использована со следующими параметрами:

complex – находит один или все корни (чаще полинома) в комплексной форме;

fulldigits – определяет счет для полного числа цифр, заданного функцией **Digits**;

maxsols=n – задает вычисление только n корней;

interval – задается в виде **a..b** или **x=a..b**, или **{x=a..b, y=c..d, ...}** и обеспечивает поиск корней в указанном интервале.

Следующие примеры демонстрируют возможности использования **fsolve**.

```
> fsolve(exp(x)=sin(x), x=-4..0); # Поиск корня уравнения  $e^x = \sin x$ , принадлежащего отрезку  $[-4, 0]$   
-3.183063012
```

```
> fsolve(exp(x)=sin(x), x=-7..-4); # То же на отрезке  $[-7, -4]$   
-6.281314366
```

Обратите внимание, Maple может выдать только один корень, лежащий на заданном отрезке, даже если на самом деле их там несколько. Поэтому вся ответственность по отделению корней возлагается на пользователя.

```
> fsolve(exp(x)=sin(x), x, complex); # Комплексный корень  
.3627020561 - 1.133745919 I
```

```
> fsolve({sin(x+y)-exp(x)*y=0, x^2-y=2}, {x, y},  
{x=-1..1, y=-2..0}); # Численное решение системы с указанием интервалов, которым принадлежат неизвестные  
{x = -0.6687012050, y = -1.552838698}
```

У п р а ж н е н и я

1. Решить нелинейные уравнения и системы уравнений одним из итерационных методов или с помощью встроенной Maple-функции. Предварительно отделить корни графическим методом.

а) Уравнения с одним неизвестным

- 1) $x^6 - 3x^2 + 1 = 0$; 2) $\sqrt{x+1} - 1/x = 0$;
- 3) $3x - e^{0.5x} = 0$; 4) $x^{21} = a$;
- 5) $x^2 - \cos^2 \pi x = 0$; 6) $5x^3 + 2x^2 + x - 1 = 0$;
- 7) $3x - \cos x - 1 = 0$; 8) $2x - \ln x = 5$;
- 9) $2x - 1,3^x = 0$; 10) $e^x - 6x - 3 + \operatorname{tg} x = 0, x \in [-\pi, \pi]$.

б) Системы уравнений

- 1) $\sin(x - y) - xy = -1, x^2 - y^2 = 3/4$;
- 2) $(x+1)^2 + y^2 = 2, x + y = 1.5$;
- 3) $x + 2 \ln x - y^2 = 0, 2x^2 - xy - 5x + 1 = 0$;
- 4) $x^{10} + y^{10} = 1024, e^x - e^y = 1$;
- 5) $x + x^5 - 2yz = 0,1, y - y^4 + 3xz = -0,2, z + z^2 + 2xz = 0,3$.

2. Найти корни нелинейных уравнений, приведенных в таблице, методами

- а) половинного деления,
- б) простой итерации,
- в) касательных,
- г) секущих.

Количество и положение корней определить графически. Оценить точность полученных значений.

№	Уравнение	№	Уравнение
1	$3x^4 + 4x^3 - 12x^2 - 5 = 0$	3	$x^2 - \sin x = 1,45$
2	$x^6 - 3x^2 + x - 1 = 0$	4	$\sqrt{x} - \cos\left(\frac{\pi}{3} + x\right) = 0$

№	Уравнение	№	Уравнение
5	$(x-4)^2 \log_{0,5}(x+1) = -1$	23	$x^3 - 4x^2 + 0,3x + 2,6 = 0$
6	$x^2 \cos 2x = -1, \quad x \in [0, 2\pi]$	24	$x^4 - x^2 - 4x - 6 = 0$
7	$(x-2)^2 2^x = 1$	25	$4x - 2e^x = -3$
8	$((x-2)^2 - 1)2^x = 1$	26	$2 \sin(x-0,6) = 3,5 - x$
9	$(x-2) \cos x = 1, \quad -2\pi \leq x \leq 2\pi$	27	$5x - 8 \ln x = 8$
10	$(x-2)^3 \lg(x+11) = 1$	28	$x^2 = \sqrt{\lg(x+2)}$
11	$5 \sin x = x - 1$	29	$1,8x^2 - \sin 10x = 0$
12	$x^4 \cdot 3^x = 2$	30	$\operatorname{ch}(0,4x) = (x-2)^2$
13	$2 \lg x - \frac{x}{3} + 1 = 0$	31	$\operatorname{ctg} x - \frac{x}{5} = 1, \quad -\pi \leq x \leq \pi$
14	$2 \sin\left(x + \frac{\pi}{3}\right) = 0,5x^2 - 1$	32	$\lg x - \frac{7}{2x+6} = 0$
15	$2x^2 - 0,5^x - 3 = 0$	33	$x^3 + 0,1x^2 + 0,4x - 1,2 = 0$
16	$\cos(x+0,5) = x^2 - 1$	34	$0,5x + \lg(x-1) = 0,5$
17	$2e^x = 5x + 2$	35	$\sin 0,5x + 1 = x^2$
18	$\sin(x-0,5) - \sqrt{x} + 0,65 = 0$	36	$2x + \lg x = -0,5$
19	$\operatorname{tg}^3 x = x - 1, \quad -\frac{\pi}{2} \leq x \leq \frac{\pi}{2}$	37	$(x-2)^2 = \frac{1}{2}e^{\sqrt{x}}$
20	$\operatorname{arctg}(x-1) + x^2/2 = 0$	38	$x = \sqrt{\lg(x+2)} + 6$
21	$2 \cos\left(x + \frac{\pi}{6}\right) + x^2 = 4x - 3$	39	$\sqrt[3]{x^2 + 2} = \frac{1}{x}$
22	$x^2 - 5 + 0,4^{2x} = 0$	40	$x^2 + 4 \sin x = 0$

3. Решить систему двух нелинейных уравнений методом Ньютона. Начальные приближения определить графически.

№	Система уравнений	№	Система уравнений
1.	$\begin{cases} \sin(x+1) - y = 1,2 \\ 2x + \cos y = 2 \end{cases}$	13.	$\begin{cases} \sin(y-1) + x = 1,3 \\ y + \sin(x+1) = 0,8 \end{cases}$
2.	$\begin{cases} \sin x + 2y = 2 \\ \cos(y-1) + x = 0,7 \end{cases}$	14.	$\begin{cases} \sin(y+1) - x = 1,2 \\ 2y + \cos x = 2 \end{cases}$
3.	$\begin{cases} \sin(x-1) = 1,3 - y \\ x - \sin(y+1) = 0,8 \end{cases}$	15.	$\begin{cases} \sin x + 2y = 1,6 \\ x + \cos(y-1) = 1 \end{cases}$
4.	$\begin{cases} \cos(y-1) + x = 0,8 \\ y - \cos x = 2 \end{cases}$	16.	$\begin{cases} \cos(x-1) + y = 1 \\ 2x + \sin y = 1,6 \end{cases}$
5.	$\begin{cases} \cos(x-1) + y = 0,5 \\ x + \cos y = 3 \end{cases}$	17.	$\begin{cases} 2x - \cos(y+1) = 0 \\ y + \sin x = -0,4 \end{cases}$
6.	$\begin{cases} \cos x + y = 1,5 \\ 2x - \sin(y-0,5) = 1 \end{cases}$	18.	$\begin{cases} \cos(x+0,5) + y = 1 \\ \sin y - 2x = 2 \end{cases}$
7.	$\begin{cases} \cos(x-1) + y = 0,8 \\ x - \cos y = 2 \end{cases}$	19.	$\begin{cases} \cos(x+0,5) - y = 2 \\ \sin y - 2x = 1 \end{cases}$
8.	$\begin{cases} \cos(y-1) + x = 0,9 \\ y - \cos x = 2 \end{cases}$	20.	$\begin{cases} \cos(x-1) + \ln y = 0,5 \\ 3x - \cos y = 3 \end{cases}$
9.	$\begin{cases} \cos(y+0,5) + x = 0,8 \\ \sin x - 2y = 1,6 \end{cases}$	21.	$\begin{cases} \cos(y+0,5) - x = 2 \\ \sin x - 2y = 1 \end{cases}$
10.	$\begin{cases} \sin(x+0,5) - y = 1,2 \\ x + \cos(y-2) = 0 \end{cases}$	22.	$\begin{cases} \cos(y-1) + x = 0,5 \\ y + \cos x = 3 \end{cases}$
11.	$\begin{cases} 2y - \cos(x+1) = 0 \\ x + \sin y = -0,4 \end{cases}$	23.	$\begin{cases} \cos x + y = 1,2 \\ 2x - \sin(y-0,5) = 2 \end{cases}$
12.	$\begin{cases} \sin(x+0,6) - y = 0,5 \\ \cos(x-2) + y = 0 \end{cases}$	24.	$\begin{cases} \cos(x+1,8) + 2y = 0,5 \\ \ln x + \cos y = 0,3 \end{cases}$

№	Система уравнений	№	Система уравнений
25.	$\begin{cases} \sin(x + 0,5) - y = 1 \\ x + \cos(y - 2) = 0 \end{cases}$	33.	$\begin{cases} \cos(x + 0,5) + y = 0,8 \\ \sin y - 2x = 1,6 \end{cases}$
26.	$\begin{cases} \sin(x - 1) + y = 1,5 \\ x - \sin(y + 1) = 1 \end{cases}$	34.	$\begin{cases} \sin(y + 1) - x = 1 \\ 2y + \cos x = 2 \end{cases}$
27.	$\begin{cases} \sin(x + 2) - y = 1,5 \\ x + \cos(y - 2) = 0,5 \end{cases}$	35.	$\begin{cases} \cos(x - 1) + y = 0,5 \\ x + \sin y \cdot \cos y = 0,4 \end{cases}$
28.	$\begin{cases} \operatorname{tg}(x - 1) + 3y = 0,5 \\ 4x - \cos y = 0,6 \end{cases}$	36.	$\begin{cases} 2 \cos(x - 1) - y = 0,5 \\ -5x + \cos y = 0,7 \end{cases}$
29.	$\begin{cases} \sin(y + 2) - x = 1,5 \\ y + \cos(x - 2) = 0,5 \end{cases}$	37.	$\begin{cases} \sin(x + 1) - y = 1 \\ 2x + \cos y = 2 \end{cases}$
30.	$\begin{cases} \sin y + 2x = 2 \\ y + \cos(x - 1) = 0,7 \end{cases}$	38.	$\begin{cases} \cos y + x = 1,5 \\ 2y + \sin(x - 0,5) = 1 \end{cases}$
31.	$\begin{cases} \cos(x - 1) + 2y = 2,5 \\ x - \cos y = 3 \end{cases}$	39.	$\begin{cases} \cos(2x - 1) + 4y = 0,5 \\ xy + \cos y = 3 \end{cases}$
32.	$\begin{cases} \cos x + \log_3 y = 1,5 \\ xy + \cos y = 3 \end{cases}$	40.	$\begin{cases} \cos(x + 5) - xy = 2,5 \\ \ln x + y^2 = 3 \end{cases}$

4*. Составить программу решения уравнения $f(x)=0$ с помощью метода:

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

б) $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} - \frac{f''(x_n)f^2(x_n)}{2[f'(x_n)]^3}, \quad n=0, 1, 2, \dots;$

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

5*. Для функции $f(x)$ найти точку минимума x^* и минимальное значение $f(x^*)$ методом Ньютона

а) $f(x) = x^2 - x + e^{-x}$; б) $f(x) = 2x^2 + x + \cos^2 x$;

в) $f(x) = (x - 4)^2 + \ln x$; г) $f(x) = e^x + e^{-2x} + 2x$.

2. РЕШЕНИЕ ЗАДАЧ ЛИНЕЙНОЙ АЛГЕБРЫ

Методам линейной алгебры принадлежит особая роль в математике, особенно в численном анализе. Ведь многие задачи математического анализа, дифференциальных и интегральных уравнений после дискретизации сводятся к решению задач линейной алгебры. Более того, большинство нелинейных моделей в малой окрестности решения могут быть описаны как конечномерные линейные модели, в основе исследования которых также лежит линейная алгебра.

2.1. Матричные и векторные операции

Прежде чем приступить к решению задач в пакете Maple, напомним некоторые определения, относящиеся к матрицам и векторам.

Матрица размера $(m \times n)$ – прямоугольная двумерная таблица, содержащая m строк и n столбцов элементов. В расширенной трактовке каждый элемент матрицы может быть не только числом, но и переменной, функцией, выражением и т.д. Квадратная матрица – матрица, у которой число строк равно числу столбцов.

Вырожденная или *особенная* матрица – квадратная матрица с определителем, равным нулю.

Единичная матрица E – квадратная матрица, у которой диагональные элементы равны 1, а остальные равны нулю.

Нижняя (верхняя) треугольная матрица – матрица, у которой элементы над (под) главной диагональю равны нулю.

Транспонированная матрица A^T – матрица A , у которой столбцы и строки поменялись местами, т.е. элементы матриц A^T и A связаны соотношением $a_{ij}^T = a_{ji}$.

Обратная к A матрица – это матрица A^{-1} , которая, будучи умноженной на квадратную матрицу A , дает единичную матрицу E .

След матрицы – сумма её элементов на главной диагонали.

Ранг матрицы – наибольший из порядков отличных от нуля миноров квадратной матрицы.

Определитель матрицы – это многочлен от элементов квадратной матрицы $\mathbf{A}$ порядка n , определяемый формулой

$$\det \mathbf{A} = \sum_{j=1}^n a_{1j} (-1)^{j+1} M_1^{(j)},$$

где $M_1^{(j)}$ – определитель матрицы порядка $n-1$, полученной из матрицы $\mathbf{A}$ вычеркиванием первой строки и j -го столбца.

Симметричная матрица – матрица $\mathbf{A}$, отвечающая условию $\mathbf{A}^T = \mathbf{A}$.

Положительно определенная матрица – квадратная матрица, отвечающая условию $\mathbf{x} \mathbf{A} \mathbf{x}^T > 0$ для любого вектора $\mathbf{x} \neq 0$.

Собственный вектор ($n \times n$)-матрицы $\mathbf{A}$ – любой вектор $\mathbf{x} \neq 0$ размерности n , удовлетворяющий условию $\mathbf{A} \mathbf{x} = \lambda \mathbf{x}$, где λ – некоторое число, называемое *собственным значением* (числом) матрицы $\mathbf{A}$. Собственные значения матрицы являются корнями её характеристического многочлена $\det(\mathbf{A} - \lambda \mathbf{E})$.

Норма – обобщенное понятие абсолютной величины числа. В качестве нормы вектора $\mathbf{x} = (x_1, x_2, \dots, x_n)$ чаще всего используют следующие

$$\text{а) } \|\mathbf{x}\|_2 = \left(\sum_{i=1}^n x_i^2 \right)^{1/2} \quad (\text{евклидова норма; в трехмерном про-}$$

странстве совпадает с длиной вектора);

$$\text{б) } \|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|;$$

$$\text{в) } \|\mathbf{x}\|_\infty = \max_i |x_i|.$$

Нормой матрицы $\mathbf{A}$, обозначаемой $\|\mathbf{A}\|$, называется наименьшее из чисел c в неравенстве $\|\mathbf{A} \mathbf{x}\| \leq c \|\mathbf{x}\| \quad \forall \mathbf{x} \neq 0$. В зависимости от применяемой нормы вектора будут получаться различные значения нормы матрицы:

$$\|\mathbf{A}\|_1 = \max_j \sum_i |a_{ij}|; \quad \|\mathbf{A}\|_\infty = \max_i \sum_j |a_{ij}|; \quad \|\mathbf{A}\|_2 = \sqrt{\rho(\mathbf{A}^T \mathbf{A})},$$

где $\rho(\mathbf{A}^T \mathbf{A})$ – максимальное собственное число матрицы $\mathbf{A}^T \mathbf{A}$.

Число обусловленности ν матрицы $\mathbf{A}$ определяется как $\nu = \|\mathbf{A}\| \cdot \|\mathbf{A}^{-1}\|$ и характеризует степень зависимости погрешности решения системы $\mathbf{Ax}=\mathbf{b}$ от погрешности правой части. Если матрица близка к особенной, то число ν будет для такой матрицы велико. В этом случае говорят, что матрица $\mathbf{A}$ плохо обусловлена.

> **restart**; # Очистка оперативной памяти

В ядре системы Maple содержатся весьма небольшие и минимально необходимые средства для решения задач линейной алгебры. Полный набор таких средств реализован в подключаемых пакетах. Основным из них является пакет расширения **linalg**, который содержит свыше ста функций. Чтобы его подключить, следует запустить команду:

> **with(linalg)** ;

Ниже приводятся примеры выполнения основных действий над матрицами и векторами.

1. Задание квадратной матрицы 3-го порядка:

> **A:=matrix**([[2,1,3],[5,1,0],[7,8,9]]) ;

$$A := \begin{bmatrix} 2 & 1 & 3 \\ 5 & 1 & 0 \\ 7 & 8 & 9 \end{bmatrix}$$

2. Вычисление определителя матрицы:

> **det(A)** ;

72

3. Формирование матрицы, полученной вычеркиванием i -й строки и j -го столбца матрицы $\mathbf{A}$:

> **minor(A,2,1)** ;

$$\begin{bmatrix} 1 & 3 \\ 8 & 9 \end{bmatrix}$$

Определитель этой матрицы дает минор элемента с индексами i и j матрицы $\mathbf{A}$.

4. Вычисление ранга матрицы:

> **rank(A)** ;

3

5. Вычисление следа матрицы:

> **trace(A)** ;

12

6. Транспонирование матрицы:

> **AT:=transpose(A)** ;

$$AT := \begin{bmatrix} 2 & 5 & 7 \\ 1 & 1 & 8 \\ 3 & 0 & 9 \end{bmatrix}$$

7. Вычисление обратной матрицы:

> **A1:=inverse(A)** ;

$$A1 := \begin{bmatrix} \frac{1}{8} & \frac{5}{24} & \frac{-1}{24} \\ \frac{-5}{8} & \frac{-1}{24} & \frac{5}{24} \\ \frac{11}{24} & \frac{-1}{8} & \frac{-1}{24} \end{bmatrix}$$

8. Умножение матриц (используется знак операции &*):

> **evalm(A&*A1)** ;

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(**evalm** – это функция оценивания матричных выражений; используется для вывода результата в матричной форме).

То же самое действие умножения можно было выполнить по команде

> **multiply(A,A1)** ;

9. Вычисление собственных значений матрицы:

> **eigenvals(A)** ;

Из-за громоздкости результат выполнения этой команды не приводится; для представления в числовой форме рекомендуется использовать функцию **evalf**, например,

> **eigenvals(A) : evalf(")** ;

12.23573596, -0.117867978 + 2.422913542 I,
-0.117867978 - 2.422913542 I

Другой вариант:

```
> evalf(eigenvals(A));
```

Собственные числа можно получить, непосредственно используя их определение, т.е. через определитель.

```
> Id:=evalm(array(identity,1..3,1..3)); # задание единичной матрицы E
```

$$Id := \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

```
> B:=evalm(A-lambda*Id); # ввод матрицы A-λE
```

$$B := \begin{bmatrix} 2-\lambda & 1 & 3 \\ 5 & 1-\lambda & 0 \\ 7 & 8 & 9-\lambda \end{bmatrix}$$

```
> f:=det(B); # вычисление определителя этой матрицы
```

$$f := 72 - 3\lambda + 12\lambda^2 - \lambda^3$$

Это характеристический многочлен матрицы **A**; его также можно определить с помощью команды

```
> charpoly(A,lambda);
```

Корни этого многочлена определяются с помощью функции **solve**:

```
> res:=evalf(solve(f,lambda),4);
```

$$res := 12.24, -0.118 + 2.422 I, -0.118 - 2.422 I$$

Получены три собственных числа – одно действительное ($\lambda_1 = 12.24$) и два комплексных ($\lambda_{2,3} = -0.118 \pm 2.42i$).

10. Нахождение собственных векторов:

```
> evalf(eigenvects(A),4);
```

$$v := [12.24, 1., \{[2.247, 1., 7.340]\}, \\ [-0.118 + 2.422 I, 1., \{[-0.2237 + 0.4844 I, 1., -0.5665 - 0.5225 I]\}, \\ [-0.118 - 2.422 I, 1., \{[-0.2237 - 0.4844 I, 1., -0.5665 + 0.5225 I]\}]]$$

Формат вывода ясен из нижеследующего комментария:

12.24 – первое собственное число,

1. – кратность этого числа,
 $\{[2.247, 1., 7.340]\}$ – собственный вектор, соответствующий первому собственному числу,
 $-0.118+2.422I$ – второе собственное число (оно комплексное), 1. – его кратность, и т.д.

11. Скалярное и векторное произведения векторов.

```
> u:=vector([2,3,5]) ; v:=vector([-2.3,4,10]) ;  

> dotprod(u,v) ; # скалярное произведение
```

57.4

Отметим, что функция **dotprod** будет работать для векторов произвольной размерности n . В форме **dotprod(u,u)** она вычисляет квадрат длины (евклидовой нормы) вектора **u**.

```
> crossprod(u,v) ; # векторное произведение
```

[10, -31.5, 14.9]

```
> psi:=angle(u,v) ; evalf(psi) ; # угол между векторами в радианах
```

$\psi := \arccos(0.1371563117 \sqrt{38})$

0.5633174125

Найдем смешанное произведение векторов $(\vec{u}, [\vec{v}, \vec{w}])$:

```
> w:=vector([6,-1,9]) ;  

> dotprod(crossprod(v,w),u) ;
```

225.6

12. Норма вектора. Норма матрицы. Число обусловленности матрицы.

Для вычисления нормы предназначена функция **norm**. Второй параметр этой функции определяет способ вычисления нормы (по умолчанию действует параметр **infinity**). Примеры для введённых ранее матрицы **A** и вектора **u**:

```
> norm(A,infinity) ;
```

24

```
> norm(A,1) ;
```

14

```
> norm(u) ;
```

5

С параметром **frobenius** норма вектора соответствует $\|\mathbf{x}\|_2$, норма матрицы вычисляется по формуле $\|\mathbf{A}\| = \sqrt{\sum_i \sum_j a_{ij}^2}$.

```
> norm(A, frobenius) ;
```

$3\sqrt{26}$

```
> norm(u, frobenius) ;
```

$\sqrt{38}$

Число обусловленности матрицы:

```
> cond(A) ;
```

21

```
> norm(A)*norm(A1) ; # A1 – обратная матрица
```

21

```
> [cond(1.*A, 1), cond(1.*A, 2), cond(1.*A, frobenius)] ;
```

[16.91666666, 11.92153451, 13.01561562]

2.2. Решение систем линейных алгебраических уравнений

Согласно оценке ученых-математиков, 75% всех расчетных математических задач приходится на решение систем линейных алгебраических уравнений (СЛАУ). Это не удивительно, так как математические модели тех или иных явлений или процессов либо сразу строятся как линейные алгебраические, либо сводятся к таковым посредством дискретизации и/или линеаризации. Поэтому выбор эффективного метода решения СЛАУ приобретает весьма большое значение. Современная вычислительная математика располагает большим арсеналом методов, а соответствующее программное обеспечение компьютеров – разнообразными пакетами прикладных программ, позволяющими решать многие встречающиеся на практике линейные системы. Чтобы ориентироваться в этом море методов и программ и тем самым уметь делать оптимальный выбор, нужно разбираться в основах построения методов и алгоритмов, знать их сильные и слабые стороны и границы применимости.

Все методы решения СЛАУ можно разбить на два класса: прямые и итерационные. Метод решения называется прямым,

– вещественная $n \times n$ -матрица коэффициентов данной системы. Для существования и единственности решения этой системы следует потребовать $\det \mathbf{A} \neq 0$. Эффективность методов решения системы (2.1) во многом зависит от структуры и свойств матрицы: размерности, обусловленности, симметричности, заполненности (т.е. соотношения между числом ненулевых и нулевых элементов) и др.

Рассмотрим некоторые способы решения систем линейных уравнений в пакете Maple. Следует отметить, что данный пакет располагает достаточно надежными, но не всегда гибкими, встроенными средствами по решению таких систем. Поэтому для лучшего понимания проблемы помимо этих встроенных ресурсов приводятся алгоритмы и программы, непосредственно реализующие тот или иной метод.

2.2.1. Прямые методы решения СЛАУ. Факторизация матриц

2.2.1.1. Правило Крамера

Этот весьма привлекательный в теоретическом плане метод позволяет находить неизвестные в виде дробей

$$x_i = \frac{\det \mathbf{A}_i}{\det \mathbf{A}} \quad (i = 1, 2, \dots, n),$$

знаменателем которых является определитель матрицы системы, а числителем – определители матриц $\mathbf{A}_i$, полученные из $\mathbf{A}$ заменой i -го столбца коэффициентов столбцом свободных членов $\mathbf{b}$.

С практической точки зрения метод Крамера приемлем только для систем небольшой размерности ($n \leq 4$), поскольку с увеличением n происходит резкий рост числа арифметических операций. Так, если определители вычисляются понижением порядка путем разложения по элементам какого-либо столбца или строки, то на вычисление n -го порядка будет затрачиваться $n!$ операций умножения. Факториальный рост – это очень быстрый рост. Попробуйте, например, оценить величины

$10! \approx 3,6 \cdot 10^6$, $20! \approx 2,4 \cdot 10^{18}$, $50! \approx 3,0 \cdot 10^{64}$, $100! \approx 10^{158}$ и понять, что реализовать на существующих персональных компьютерах подобный счет весьма затруднительно даже для таких небольших по современным меркам размерностей (n несколько десятков или сотен). К тому же при столь большом числе операций будут катастрофически нарастать ошибки округления и метод Крамера будет неустойчивым. Попутно заметим, что по тем же соображениям такой простой и наглядный метод решения системы (2.1), как метод обратной матрицы ($\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$), не может рассматриваться как значимый, особенно если матрица $\mathbf{A}^{-1}$ вычисляется через присоединенную.

Пусть требуется решить по правилу Крамера систему $\mathbf{Ax} = \mathbf{b}$, где матрица $\mathbf{A}$ – введенная в п. 1 предыдущего раздела, а вектор правых частей $\mathbf{b}$ равен $\begin{pmatrix} 5 \\ 6 \\ 1 \end{pmatrix}$. Сначала осуществим ввод

вектора $\mathbf{b}$:

> **b:=matrix(3,1,[5,6,1]);** # здесь вектор введен как матрица размером 3×1

Реализация метода в системе Maple может выглядеть так:

> **C:=evalm(A);** # теперь $\mathbf{C}$ то же самое, что и матрица $\mathbf{A}$

$$\mathbf{C} := \begin{bmatrix} 2 & 1 & 3 \\ 5 & 1 & 0 \\ 7 & 8 & 9 \end{bmatrix}$$

> **C1:=copyinto(b,C,1,1);** # замена в матрице $\mathbf{C}$ первого столбца на вектор $\mathbf{b}$

$$\mathbf{C1} := \begin{bmatrix} 5 & 1 & 3 \\ 6 & 1 & 0 \\ 1 & 8 & 9 \end{bmatrix}$$

Аналогичные действия со вторым и третьим столбцом:

> **C:=evalm(A): C2:=copyinto(b,C,1,2);**

$$\mathbf{C2} := \begin{bmatrix} 2 & 5 & 3 \\ 5 & 6 & 0 \\ 7 & 1 & 9 \end{bmatrix}$$

$$a_{ij}^{(1)} = a_{ij} - \frac{a_{i1}}{a_{11}} a_{1j}; \quad b_i^{(1)} = b_i - \frac{a_{i1}}{a_{11}} b_1.$$

2-й шаг. Исключение неизвестной x_2 из всех уравнений системы (2), кроме первого и второго. Для этого проделываем такие же операции, как на первом шаге, с подсистемой системы (2.2), получающейся исключением первого уравнения. Поскольку при пересчете коэффициентов будет производиться деление на элемент $a_{22}^{(1)}$, необходимо поставить условие $a_{22}^{(1)} \neq 0$. Новая преобразованная система этого шага, эквивалентная (2), а значит, и исходной системе (2.1), будет иметь вид

[illegible]

где

$$a_{ij}^{(2)} = a_{ij}^{(1)} - \frac{a_{i2}^{(1)}}{a_{22}^{(1)}} a_{2j}^{(1)}; \quad b_i^{(2)} = b_i^{(1)} - \frac{a_{i2}^{(1)}}{a_{22}^{(1)}} b_2^{(1)} \quad (i, j = 3, 4, \dots, n).$$

Продолжая этот процесс исключения, при условии, что

$$a_{33}^{(2)} \neq 0, a_{44}^{(3)} \neq 0, \dots, a_{n-1, n-1}^{(n-2)} \neq 0,$$

после $(n-1)$ -го шага исключения получим преобразованную исходную систему

[illegible]

На основе предыдущих рассуждений и формул легко убедиться, что коэффициенты этой системы могут быть получены из коэффициентов данной системы (2.1) последовательным пересчетом по формулам

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} a_{kj}^{(k-1)}; \quad b_i^{(k)} = b_i^{(k-1)} - \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} b_k^{(k-1)}, \quad (2.4)$$

где верхний индекс k (номер шага) пробегает значения от 1 до $n-1$, нижние индексы i и j (в любой очередности) – от $k+1$ до n ; по определению полагаем $a_{ij}^{(0)} \equiv a_{ij}$, $b_i^{(0)} \equiv b_i$.

В матричной записи система уравнений (2.3) примет вид

$$\mathbf{U} \mathbf{x} = \mathbf{g}$$

с верхней треугольной матрицей $\mathbf{U}$ и вектором правых частей $\mathbf{g}$

$$\mathbf{U} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ & a_{22}^{(1)} & \dots & a_{2n}^{(1)} \\ & & \ddots & \\ 0 & & & a_{nn}^{(n-1)} \end{pmatrix}; \quad \mathbf{g} = \begin{pmatrix} b_1 \\ b_2^{(1)} \\ \vdots \\ b_n^{(n-1)} \end{pmatrix}. \quad (2.5)$$

Приведение исходной системы линейных уравнений к системе (2.3) с треугольной матрицей $\mathbf{U}$ называют *прямым ходом* метода Гаусса. На этом этапе не вычисляется ни одной компоненты вектора-решения $\mathbf{x}$, но эквивалентными преобразованиями система приводится к такой форме, из которой легко найти эти самые компоненты.

Пусть $a_{nn}^{(n-1)} \neq 0$. Тогда осуществляем *обратный ход* метода Гаусса: вычисляем последовательно одно за другим значения неизвестных в обратном порядке. Из (2.3) находим

$$x_n = \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}},$$

$$x_{n-1} = \frac{1}{a_{n-1,n-1}^{(n-2)}} (b_{n-1}^{(n-2)} - a_{n-1,n}^{(n-2)} x_n),$$

.....

$$x_2 = \frac{1}{a_{22}^{(1)}} (b_2^{(1)} - a_{23}^{(1)} x_3 - \dots - a_{2n}^{(1)} x_n),$$

$$x_1 = \frac{1}{a_{11}} (b_1 - a_{12} x_2 - \dots - a_{1n} x_n).$$

Очевидно, этап обратного хода можно выразить одной формулой

$$x_k = \frac{1}{a_{kk}^{(k-1)}} \left(b_k^{(k-1)} - \sum_{j=k+1}^n a_{kj}^{(k-1)} x_j \right), \quad (2.6)$$

где k полагают равным $n, n-1, \dots, 2, 1$ и сумма по определению считается равной нулю, если нижний предел суммирования у знака Σ имеет значение больше верхнего.

Итак, решение СЛАУ вида (2.1) методом Гаусса сводится к последовательной реализации вычислений по формулам (2.4) и (2.6). Составим на языке Maple программу по этим вычислениям.

```
> restart;
> with(linalg);
> A:=matrix([[4,2,-1,0],[2,4,2,1],[-1,2,8,4],
[0,1,4,10]]);
b:=vector([9,11,7,28]); x:=vector(4):
      A :=  $\begin{bmatrix} 4 & 2 & -1 & 0 \\ 2 & 4 & 2 & 1 \\ -1 & 2 & 8 & 4 \\ 0 & 1 & 4 & 10 \end{bmatrix}$ 
      b := [9, 11, 7, 28]

> A1:=evalm(A); b1:=evalm(b);
> n:=4; eps:=1.0e-7;
# Прямой ход метода Гаусса:
> for k from 1 to n-1 do # 1) для  $k=1, 2, \dots, n-1$ 
    for i from k+1 to n do # 2) для  $i=k+1, \dots, n$ 
        t:=A[i,k]/A[k,k];
        b[i]:=b[i]-t*b[k]; #  $b_i \leftarrow b_i - (a_{ik}/a_{kk}) b_k$ 
        for j from k+1 to n do # для  $j=k+1, \dots, n$ 
```

```

        A[i,j]:=A[i,j]-t*A[k,j]; #  $a_{ij} \leftarrow a_{ij} - (a_{ik}/a_{kk}) a_{kj}$ 
    od;
od;
od;
# Обратный ход метода Гаусса:
x[n]:=b[n]/A[n,n]; #  $x_n \leftarrow b_n / a_{nn}$ 
for k from n-1 by -1 to 1 do # для  $k=n-1, \dots, 2, 1$ 
    s:=0; # переменная s служит для накопления суммы  $\sum_{j=k+1}^n a_{kj} x_j$ 
    for j from k+1 to n do
        s:=s+A[k,j]*x[j]; od;
    x[k]:=(b[k]-s)/A[k,k]; #  $x_k \leftarrow \left( b_k - \sum_{j=k+1}^n a_{kj} x_j \right) / a_{kk}$ 
od;
> evalf[5] (evalm(x)); # вывод вектора-решения
[1., 2., -1., 3.]
> evalm(A1&*x-b1); # проверка решения прямой подстановкой
[0, 0, 0, 0]

```

Так как реальные машинные вычисления производятся не с точными, а с усеченными числами, т.е. неизбежны ошибки округления, то, анализируя приведенные формулы, можно сделать вывод о том, что выполнение алгоритма может прекратиться или привести к неверным результатам, если знаменатели дробей на каком-то шаге окажутся равными нулю или очень маленькими числами. Поэтому чтобы уменьшить влияние ошибок округлений и исключить деление на ноль, на каждом шаге прямого хода уравнения обрабатываемой подсистемы обычно переставляют так, чтобы деление производилось на наибольший по модулю в данном столбце элемент. Элементы матрицы, на которые производится деление в методе Гаусса, называются главными или ведущими. Рассматриваемая модификация метода Гаусса с перестановкой уравнений для исключения деления на ноль и уменьшения вычислительной погрешности называется метод Гаусса с выбором главного элемента по столбцам.

Алгоритм постолбцового выбора главного элемента на k -м шаге прямого хода выглядит так:

- 1) найти $m \geq k$ такое, что $|a_{mk}| = \max_{i \geq k} \{|a_{ik}|\}$;
- 2) если $|a_{mk}| < \varepsilon$, остановить работу программы (система не разрешима),
- 3) иначе (при $m \neq k$) поменять местами b_k и b_m , a_{kj} и a_{mj} для всех $j = k, \dots, n+1$.

Соответствующий фрагмент, приведенный ниже, должен быть вставлен между строками 1 и 2 программы метода Гаусса.

```
main:=abs(A[k,k]); m:=k;
for i from k+1 to n do
  if abs(A[i,k])>main then
    main:=abs(A[i,k]); m:=i; fi; od;
if (main>eps) then if (m<>i) then
  t:=b[m]; b[m]:=b[k]; b[k]:=t; # обмен  $b_k \rightleftharpoons b_m$ 
  for j from k to n do # обмен  $a_{kj} \rightleftharpoons a_{mj}$ 
    t:=A[m,j]; A[m,j]:=A[k,j]; A[k,j]:=t; od;
  fi;
else print(`однозначного решения нет`); `quit`(1);
fi;
```

Устойчивость метода Гаусса можно еще усилить, если на каждом шаге осуществлять выбор главного элемента по всей матрице. Но такая модификация довольно сильно усложняет алгоритм, и поэтому здесь не приводится.

Теперь обратимся к встроенным средствам системы Maple. Напомним, что для их активации потребуется подключение пакета **linalg**. Если не указано специально, далее все примеры вычислений приведены для тех же матрицы **A** и вектора **b**, что были использованы при решении методом Крамера. Отличие только в том, что **b** должен быть определен не как матрица 3×1 , а как вектор. Итак, имеем

```
> with(linalg);
> A:=matrix([[2,1,3],[5,1,0],[7,8,9]]);
> b:=vector([5,6,1]);
```

Следующая функция осуществляет гауссово исключение матрицы A:

> **U:=gausselim(A) ;**

$$H := \begin{bmatrix} 2 & 1 & 3 \\ 0 & \frac{-3}{2} & \frac{-15}{2} \\ 0 & 0 & -24 \end{bmatrix}$$

Если применить гауссово исключение к расширенной матрице системы, получим прямой ход метода Гаусса:

> **Ar:=augment(A,b) ; #** *получение расширенной матрицы*

$$Ar := \begin{bmatrix} 2 & 1 & 3 & 5 \\ 5 & 1 & 0 & 6 \\ 7 & 8 & 9 & 1 \end{bmatrix}$$

> **H:=gausselim(Ar) ; #** *расширенная матрица преобразованной системы (2.3)*

$$H := \begin{bmatrix} 2 & 1 & 3 & 5 \\ 0 & \frac{-3}{2} & \frac{-15}{2} & \frac{-13}{2} \\ 0 & 0 & -24 & -36 \end{bmatrix}$$

Обратный ход метода Гаусса:

> **backsub(H) ;**

$$\left[\frac{11}{6}, \frac{-19}{6}, \frac{3}{2} \right]$$

> **evalf(%) ;**

[1.833333333, -3.166666667, 1.500000000]

Метод Гаусса–Жордана производит элементарные преобразования над расширенной матрицей так, что ее главный минор становится диагональной матрицей с единицами на главной диагонали, а последний столбец – решением системы.

> **gaussjord(Ar) ;**

$$\begin{bmatrix} 1 & 0 & 0 & \frac{11}{6} \\ 0 & 1 & 0 & \frac{-19}{6} \\ 0 & 0 & 1 & \frac{3}{2} \end{bmatrix}$$

Как известно, такие преобразования не меняют ранга матрицы, поэтому отчетливо видно, что ранг исходной матрицы равен 3. Здесь уместно отметить, что Maple умеет решать системы, у которых решение не единственное. Изучите внимательно следующий пример с такой системой:

```
> G:=matrix([ [1,2,3] , [2,1,3] , [1,-1,0] ]) :
v:=vector([1,2,1]) : H1:=augment(G,v) :
F1:=gaussjordan(H1) ;
```

$$F1 := \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

```
> backsub(F1) ;
```

$$[1 - t_1, -t_1, -t_1]$$

Обратите внимание, здесь ранг равен 2, и одно неизвестное выбирается произвольно (обозначено $_t1$), а остальные выражаются через него.

С методом Гаусса тесно связана факторизация (разложение на сомножители) матрицы:

$$\mathbf{A} = \mathbf{L}\mathbf{U},$$

где $\mathbf{L}$ – нижняя треугольная матрица с единицами на главной диагонали:

$$\mathbf{L} = \begin{pmatrix} 1 & & & & \\ \frac{a_{21}}{a_{11}} & 1 & & & 0 \\ \frac{a_{31}}{a_{11}} & \frac{a_{32}^{(1)}}{a_{22}^{(1)}} & 1 & & \\ \dots & \dots & \dots & \dots & \\ \frac{a_{n1}}{a_{11}} & \frac{a_{n2}^{(1)}}{a_{22}^{(1)}} & \dots & \frac{a_{n,n-1}^{(n-2)}}{a_{n-1,n-1}^{(n-2)}} & 1 \end{pmatrix},$$

$\mathbf{U}$ – верхняя треугольная матрица, которая уже была введена ранее (см. (2.5)). Приведём две теоремы, связанные с такой факторизацией: 1) для существования LU -разложения матрицы

A необходимо и достаточно, чтобы у матрицы **A** все главные миноры были отличны от нуля; 2) произвольная невырожденная матрица перестановкой строк (столбцов) может быть приведена к матрице с главными минорами, отличными от нуля.

Следующая функция осуществляет данную факторизацию

```
> u:=LUdecomp(A,L='l'); # здесь символ l буква, а не цифра
```

Указанные треугольные матрицы хранятся в переменных **u** и **l**; вывести их можно с помощью стандартных команд

```
> eval(l);
```

```
> eval(u);
```

На основе полученного разложения можно получить решение системы $\mathbf{Ax}=\mathbf{b}$. Для этого последовательно решаются треугольные системы $\mathbf{Lg}=\mathbf{b}$ и $\mathbf{Ux}=\mathbf{g}$. Для решения этих систем в Maple предусмотрены функции **forwardsub** и **backsub**:

```
> g:=forwardsub(l,b); # прямой ход
```

$$g := \left[5, \frac{-13}{2}, -36 \right]$$

```
> x:=backsub(u,g); # обратный ход
```

$$x := \left[\frac{11}{6}, \frac{-19}{6}, \frac{3}{2} \right]$$

```
> evalm(l&*u&*x-b); # проверка решения: вычисляется  $\mathbf{LUx-b}$   
[0, 0, 0]
```

Maple-функция **linsolve** позволяет непосредственно решать системы линейных уравнений вида $\mathbf{Ax}=\mathbf{b}$, а также матричных уравнений $\mathbf{AX}=\mathbf{B}$. Результат действия функции – вектор **x** или матрица **X**.

```
> linsolve(A,b); # решение линейной системы
```

Вектор правых частей здесь можно задавать как матрицу размером 3×1 (см. правило Крамера), так и непосредственно как вектор (**vector**):

```
> b:=vector([5,6,1]): linsolve(A,b);
```

$$\left[\frac{11}{6}, \frac{-19}{6}, \frac{3}{2} \right]$$

Если эту функцию запустить с третьим аргументом –
`> linsolve(A,b,'r');`
 то через переменную **r** можно будет вывести ранг матрицы **A**:
`> r;`

3

2.2.1.3. Метод Холесского

Симметричные положительно определенные матрицы факторизуются по методу Холесского, который, по сути, является симметричным вариантом исключения метода Гаусса. При этом разложение имеет вид

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T \quad (2.7)$$

(**L** – нижняя треугольная матрица). Для положительно определенных матриц выполняется критерий Сильвестра: все главные миноры строго положительны, в силу чего разложение (2.7) существует. Формулы для вычисления элементов матрицы Холесского **L** можно вывести, приравнявая последовательно по столбцам соответствующие матричные элементы слева и справа равенства (2.7). Выпишем их:

$$l_{jj} = \left(a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2 \right)^{1/2}, \quad j = 1, \dots, n,$$

$$l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} \right), \quad i = j+1, \dots, n; \quad (2.8)$$

Эти соотношения программируются как вложенные циклы: цикл по *j* – внешний цикл, а по *i* – внутренний. После того как получено разложение (2.7), решение СЛАУ получается решением двух систем с треугольными матрицами: **Lg=b** и **L^Tx=g**. Важно отметить, что метод Холесского применительно к симметричным положительно определенным матрицам всегда численно устойчив и не требует выбора главного элемента.

Ниже приведен пример на разложение Холесского матрицы **A^TA**, которая, как известно, при условии $\det \mathbf{A} \neq 0$ обладает необходимыми свойствами.

```
> W:=multiply(AT,A) ; # произведение матрицы на свою транспонированную (эти матрицы должны быть в памяти компьютера)
> definite(W,positive_def) ; # проверка на положительную определенность
```

```
> L:=cholesky(W) ; # вычисление матрицы Холецкого
```

$$L := \begin{bmatrix} \sqrt{78} & 0 & 0 \\ \frac{21\sqrt{78}}{26} & \frac{\sqrt{10218}}{26} & 0 \\ \frac{23\sqrt{78}}{26} & \frac{167\sqrt{10218}}{3406} & \frac{24\sqrt{131}}{131} \end{bmatrix}$$

```
> evalm(L&*transpose(L)) ; # вычисление произведения LLT для контроля
```

Этот пример подсказывает возможный способ применения метода Холецкого к решению системы с произвольной невырожденной матрицей. Умножив слева обе части системы $Ax=b$ на A^T , получим эквивалентную систему $Wx=v$, где $W=A^T A$, $v=A^T b$, к которой в силу симметричности и положительной определенности матрицы W может применяться факторизация Холецкого. В рамках реализации этого способа последний пример следует дополнить строками

```
> g:=forwardsub(L,AT*b) ;
> x:=backsub(transpose(L),g) ;
```

$$x := \left[\frac{11}{6}, \frac{-19}{6}, \frac{3}{2} \right]$$

3

2.2.2. Итерационные методы решения СЛАУ

С помощью итерационных методов строится последовательность векторов $\{x^{(k)}\}$ такая, что $\lim_{k \rightarrow \infty} x^{(k)} = x$, где x – точное решение системы. Эффективность таких методов определяется скоростью сходимости последовательных приближений $x^{(k)}$ к x , причем немаловажно, чтобы вычисления на каждой итерации были не слишком затратными.

Итерационные способы построения решения систем линейных алгебраических уравнений являются серьёзной альтернативой прямым методам, по крайней мере, в случаях, когда размерность этих систем велика. Особенно широкого применения методы итераций получили для систем уравнений, к которым приводят численные методы решения дифференциальных уравнений в частных производных. Матрицы таких систем имеют большое число нулевых элементов, и в отличие от прямых методов итерационные не увеличивают число ненулевых элементов матрицы в процессе вычислений.

Отметим еще одну важную особенность итерационных методов: как правило, они менее чувствительны к погрешностям округления, возникающим в реальных компьютерных расчетах из-за конечности разрядной сетки. Эти методы даже используют для уточнения решения, полученного прямыми методами. С другой стороны, если матрица системы плохо обусловлена, их скорость сходимости может быть очень медленной.

2.2.2.1. Методы Якоби и Гаусса–Зейделя

Довольно широкий класс итерационных методов в своей основе использует следующую идею: пусть ищется решение невырожденной системы

$$\mathbf{A}\mathbf{x}=\mathbf{b}, \quad (2.9)$$

которая может быть преобразована к эквивалентной системе

$$\mathbf{x}=\mathbf{B}\mathbf{x}+\mathbf{c} \quad (2.10)$$

с тем же самым вектором неизвестных $\mathbf{x}$ и некоторыми новыми матрицей $\mathbf{B}$ и вектором $\mathbf{c}$. Далее строится итерационный процесс по правилу

$$\mathbf{x}^{(k+1)}=\mathbf{B}\mathbf{x}^{(k)}+\mathbf{c}, \quad k=0, 1, 2, \dots, \quad (2.11)$$

где $\mathbf{x}^{(0)}=(x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})^T$ – заданный вектор неизвестных – начальное приближение. В качестве условия завершения счёта обычно используют малость величин $\|\mathbf{x}^{(k+1)}-\mathbf{x}^{(k)}\|$ и/или

Конкретные методы отличаются способом приведения исходной системы к виду (2.10), а также условиями и параметрами сходимости итерационного процесса (2.11). К примеру, представим матрицу системы в виде

где $\mathbf{A}$ – диагональная, $\mathbf{L}$ и $\mathbf{R}$ – соответственно левая и правая строго треугольные (т.е. с нулевой диагональю) матрицы. Тогда система (2.9) может быть записана в виде

или, если на главной диагонали исходной матрицы нет нулей,

т.е. в (2.10) и (2.11) следует положить

Построенный таким образом итерационный процесс

называется методом Якоби. Его легко записать в развернутом виде, если учесть, что $\mathbf{D}^{-1}$, как и $\mathbf{D}$, – матрица диагональная, но с элементами $d_{ij} = 1/a_{ij}$:

Под методом Гаусса–Зейделя (или просто Зейделя) понимают такую модификацию простых итераций (2.11), когда найденные при вычислении $(k+1)$ -го приближения компоненты вектора $\mathbf{x}^{(k+1)}$ сразу же используются в вычислениях. В координатной форме итерационный процесс Зейделя имеет вид

[illegible]

Рассматривая данную модификацию применительно к итерациям метода Якоби (2.12), получим

$$\left\{ \begin{array}{l} x_1^{(k+1)} = (b_1 - a_{12}x_2^{(k)} - a_{13}x_3^{(k)} - \dots - a_{1n}x_n^{(k)}) / a_{11}, \\ x_2^{(k+1)} = (b_2 - a_{21}x_1^{(k+1)} - a_{23}x_3^{(k)} - \dots - a_{2n}x_n^{(k)}) / a_{22}, \\ \vdots \\ x_n^{(k+1)} = (b_n - a_{n1}x_1^{(k+1)} - a_{n2}x_2^{(k+1)} - \dots - a_{nn-1}x_{n-1}^{(k+1)}) / a_{nn}, \end{array} \right. \quad (2.13)$$

где $k = 0, 1, 2, \dots$, а $\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})^T$ задается.

Заметим, что итерационный процесс (2.13) укладывается в общее представление (2.11), при этом

$$\mathbf{B} = -(\mathbf{L} + \mathbf{D})^{-1} \mathbf{R}; \quad \mathbf{c} = (\mathbf{L} + \mathbf{D})^{-1} \mathbf{b}.$$

Точные формулировки условий сходимости методов Якоби и Зейделя можно найти в литературе по численным методам (см., например, [2, 5]). Отметим здесь лишь, что для определенного класса матриц эти методы непременно сходятся, а именно для матриц $\mathbf{A}$ с диагональным преобладанием, т.е. таких, для которых справедливо

$$|a_{ii}| < \sum_{\substack{j=1, \\ i \neq j}}^n |a_{ij}| \quad \text{для всех } i = 1, 2, \dots, n.$$

Кроме того, метод Зейделя сходится для систем с симметричными положительно определенными матрицами, а это значит, что им можно решать системы общего вида (2.12), но применяя к видоизмененной форме $\mathbf{A}^T \mathbf{A} \mathbf{x} = \mathbf{A}^T \mathbf{b}$ (сходимость при этом может сильно замедляться).

Ниже приводится текст программы по реализации метода Зейделя типа (2.13). Для счёта используется система

$$\begin{cases} 4x_1 + 2x_2 - x_3 = 9, \\ 2x_1 + 4x_2 - 2x_3 + x_4 = 11, \\ -x_1 + 2x_2 + 8x_3 + 4x_4 = 7, \\ x_2 + 4x_3 + 10x_4 = 28. \end{cases}$$

```
> restart;
> with(linalg);
> A:=matrix([[4.,2,-1,0],[2,4,2,1],[-1,2,8,4],
[0,1,4,10]]);
> b:=vector([9.,11,7,28]); # 1)
> x:=vector(4); x0:=vector(4); x:=[0.,0.,0.,0.];# 2)
> eps:=1.0e-5: n:=4: k:=0:
> r:=evalm(1.*A&x-b);
> while sqrt(dotprod(r,r))>eps do
  for i from 1 to n do
    s:=0.; x0[i]:=x[i];
    for j from 1 to n do
      if i<>j then s:=(s+A[i,j]*x[j]);fi; od;
    x[i]:=(b[i]-s)/A[i,i];
    r[i]:=x[i]-x0[i];
  od;
  k:=k+1;
  if k>1000 then
    print(`метод не сходится`); `stop`(1); fi;
od:
> evalm(x); k;
[1.000004396, 1.999996252, -0.9999981812, 2.999999647]
```

Указание. Если в процессе счета наблюдается заикливание (т.е. метод не сходится), рекомендуется провести аналогичный расчет для эквивалентной системы $\mathbf{W}\mathbf{x}=\mathbf{g}$, где $\mathbf{W}=\mathbf{A}^T\mathbf{A}$, $\mathbf{g}=\mathbf{A}^T\mathbf{b}$. Для этого между строками 1 и 2 в данной программе нужно вставить

```
> W:=evalm(transpose(A)&*A);
> g:=evalm(transpose(A)&*b);
```

а в остальном тексте программы вместо матрицы $\mathbf{A}$ и вектора $\mathbf{b}$ вставить соответственно $\mathbf{W}$ и $\mathbf{g}$.

2.2.2.2. Метод сопряженных градиентов

Еще один класс методов итерационного решения СЛАУ – это так называемые методы вариационного типа. К ним относятся методы минимальных невязок, минимальных ошибок, наискорейшего спуска, сопряженных градиентов и др. Их главная особенность в том, что непосредственное решение системы подменяется решением эквивалентной экстремальной задачи.

Например, один из наиболее популярных и хорошо разработанных методов подобного типа – метод сопряженных градиентов (МСГ) – минимизирует квадратичный функционал

$$F(\mathbf{x}) = \frac{1}{2}(\mathbf{Ax}, \mathbf{x}) - (\mathbf{b}, \mathbf{x}).$$

Можно показать, что если матрица $\mathbf{A}$ симметричная и положительно определенная, то вектор $\mathbf{x}$, для которого $F(\mathbf{x})$ достигает минимума, одновременно является решением системы $\mathbf{Ax} = \mathbf{b}$.

Приведем без вывода алгоритм МСГ.

1. Задать $\mathbf{x}^{(0)}$ (начальный вектор) и число ε (уровень допустимых погрешностей).

2. Вычислить вектор $\mathbf{r}^{(0)} = \mathbf{Ax}^{(0)} - \mathbf{b}$ (невязка начального приближения – разность между левой и правой частью системы).

3. Положить $\mathbf{v}^{(0)} = \mathbf{r}^{(0)}$, $k = 0$ (номер итерации).

4. Вычислить число $\alpha_k = -\frac{(\mathbf{r}^{(k)}, \mathbf{v}^{(k)})}{(\mathbf{r}^{(k)}, \mathbf{Av}^{(k)})}$.

5. Вычислить вектор $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{v}^{(k)}$ (новое приближение).

6. Вычислить вектор $\mathbf{r}^{(k+1)} = \mathbf{Ax}^{(k+1)} - \mathbf{b}$ (невязка $(k+1)$ -го приближения).

7. Вычислить число $\beta_k = \frac{(\mathbf{r}^{(k+1)}, \mathbf{r}^{(k+1)})}{(\mathbf{r}^{(k)}, \mathbf{r}^{(k)})}$.

8. Вычислить вектор $\mathbf{v}^{(k+1)} = \mathbf{r}^{(k+1)} + \beta_k \mathbf{v}^{(k)}$.

9. Положить $k \leftarrow k + 1$.

10. Проверить условие $\|\mathbf{r}^{(k+1)}\| < \varepsilon$; если “да”, остановить счёт и вывести результаты, если “нет”, перейти к шагу 4.

Интересно отметить, что МСГ занимает особое положение среди методов решения СЛАУ, ибо доказано, что векторы невязок $\{\mathbf{r}^{(k)}, k = 0, 1, 2, \dots\}$ образуют ортогональную систему векторов, следовательно, нулевая невязка $\mathbf{r}^{(k)} = 0$ (что равносильно точному решению) должна появиться не позднее n -й итерации, где n – порядок системы уравнений. Таким образом, МСГ должен быть отнесен к прямым методам. Тем не менее на практике он применяется именно как итерационный метод, поскольку реальный вычислительный процесс не идеален из-за неизбежных ошибок округления, и на n -м шаге может быть не достигнута нужная точность. Кроме того, для систем большой размерности для получения решения с приемлемой точностью может понадобиться значительно меньшее, чем n , число итераций. Поэтому в силу указанных обстоятельств к МСГ иногда употребляют термин «полуитерационный метод».

Запрограммируем алгоритм МСГ в системе Maple (на примере той же системы, что была решена методом Зейделя).

```
> restart;
> with(linalg);
> A:=matrix([[4.,2,-1,0],[2,4,2,1],[-1,2,8,4],
[0,1,4,10]]);
> b:=vector([9.,11,7,28]); r:=vector(4):
> x:=eval(b): eps:=0.0001: k:=0:
> r:=evalm(A&*x-b); v:=eval(r); t1:=dotprod(r,r);
> while sqrt(t1)>eps do
    alpha:=-evalf(dotprod(r,v)/dotprod(A&*v,v));
    x:=evalm(x+alpha*v): print(eval(x));
    t2:=t1; r:=evalm(A&*x-b); t1:=dotprod(r,r);
    beta:=t1/t2;
    v:=evalm(r+beta*v);
    k:=k+1;
    if k>1000 then print(`метод не сходится`);
                                `stop`(1); fi;
od:
> eval(x); k;
```

```
[1.0000000001, 2.000000000, -1.0000000006, 2.9999999991]
```

Замечание. Для успешной работы программы необходимо отслеживать, чтобы все вычисления проводились в числовом формате (а не в символьном). Прямым указанием на такой режим счёта, как обычно, является явная запись чисел с точкой. Обратите внимание, что здесь так записаны по одному элементу матрицы **A** и вектора **b**. В качестве дополнительной меры можно рекомендовать видоизменить команды вычисления **r**:

```
r:=evalm(1.0*A&*x-b) ;
```

а также все вычисления производить посредством **evalf**.

Если матрица системы не симметричная и не положительно определенная, сходимость МСГ не гарантируется. В этом случае предлагается применить алгоритм метода сопряженных градиентов к эквивалентной системе $A^T A x = A^T b$, у которой матрица $W = A^T A$ обладает необходимыми свойствами. Скорость сходимости при этом может быть низкой. Для реализации этого алгоритма в приведенной программе сразу после задания матрицы **A** и вектора **b** следует вставить команды

```
> W:=evalm(inverse(A) &*A) ;
```

```
> g:=evalm(inverse(A) &*b) ;
```

а далее по тексту везде поменять символ **A** на символ **W**, а символ **b** – на **g**.

Упражнения

1. В качестве **A**, **b** взять матрицу и вектор правых частей системы уравнений из таблицы; **B** задать произвольно.

2. а) Вычислить определитель матрицы **A**, её ранг, обратную матрицу, транспонированную матрицу, а также миноры элементов какой-либо строки (столбца) данной матрицы.

б) Используя встроенную функцию **norm**, определить норму вектора **b** и матрицы **A**, а также число обусловленности **A** в различных смыслах. Провести вычисление указанных величин непосредственным применением формул.

3. а) Вычислить след матриц $\mathbf{AB}$ и $\mathbf{BA}$.

б) Ввести прямоугольную матрицу $\mathbf{C}$ подходящего размера и найти $\mathbf{A}^T\mathbf{BA}$, $\mathbf{AC}$, $\mathbf{BAC}$, $(\mathbf{AB})\mathbf{C}$, $\mathbf{b}^T\mathbf{Ab}$.

4. а) Решить задачу на собственные векторы и собственные значения матрицы $\mathbf{A}$. Сделать проверку.

б) Решить обобщенную задачу на собственные значения $\mathbf{Ax}=\lambda\mathbf{Bx}$.

5. Решить систему $\mathbf{Ax}=\mathbf{b}$ уравнений: по правилу Крамера и путем вычисления обратной матрицы. Выполнить проверку.

6. а) Решить систему $\mathbf{Ax}=\mathbf{b}$ с помощью программы, реализующей метод Гаусса с выбором главного элемента. Попутно найти определитель матрицы $\det\mathbf{A}$ и выписать явно матрицы $\mathbf{L}$ и $\mathbf{U}$. Проверить равенство $\mathbf{A}=\mathbf{LU}$.

б) Используя LU -разложение, найти решение матричного уравнения $\mathbf{AX}=\mathbf{B}$. Проверить результат подстановкой.

в) На основе LU -разложения вычислить матрицу, обратную к $\mathbf{A}$. Проверить равенство $\mathbf{A}^{-1}\mathbf{A}=\mathbf{E}$.

7. а) Получить разложение Холецкого симметричной положительно определенной матрицы $\mathbf{W}=\mathbf{A}^T\mathbf{A}$. На основе этого разложения решить систему $\mathbf{Wx}=\mathbf{A}^T\mathbf{b}$ (и/или уравнение $\mathbf{WX}=\mathbf{A}^T\mathbf{B}$).

б) Задать матрицу $\mathbf{S}$, элементы которой s_{ij} представляют собой скалярные произведения векторов: $s_{ij}=(\mathbf{x}_i, \mathbf{x}_j)$, где $\mathbf{x}_1, \mathbf{x}_2, \dots$ – произвольные линейно независимые ненулевые векторы одинаковой размерности. Убедившись в положительной определенности $\mathbf{S}$, решить систему $\mathbf{Sx}=\mathbf{b}$ методом Холецкого.

8. а) Ввести три произвольных некомпланарных вектора $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ трехмерного евклидова пространства. Найти смешанное $([\mathbf{x}_1, \mathbf{x}_2], \mathbf{x}_3)$ и двойное векторное произведение $[\mathbf{x}_1, [\mathbf{x}_2, \mathbf{x}_3]]$ этих векторов. Вычислить углы, которые векторы образуют друг с другом, а также угол между $\mathbf{x}_1$ и плоскостью $\mathbf{x}_2\mathbf{x}_3$.

б) К уже введенным в № 8а векторам добавить еще один вектор $\mathbf{x}_4$, который разложить по базису $\mathbf{B}=(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3)$.

9. Применить один из итерационных методов (простой итерации, Якоби, Гаусса–Зейделя) к решению системы из таблицы. В случае отсутствия сходимости добиться диагонального преобладания путем комбинирования уравнений, либо применить метод к эквивалентной системе $\mathbf{A}^T \mathbf{A} \mathbf{x} = \mathbf{A}^T \mathbf{b}$.

10. Решить СЛАУ методом сопряженных градиентов. В качестве системы взять $\mathbf{S} \mathbf{x} = \mathbf{b}$ с матрицей $\mathbf{S}$, введенной в №76, либо $\mathbf{W} \mathbf{x} = \mathbf{g}$, где $\mathbf{W} = \mathbf{A}^T \mathbf{A}$ и $\mathbf{g} = \mathbf{A}^T \mathbf{b}$.

№	СЛАУ	№	СЛАУ
1	$\begin{cases} 0,21x_1 - 0,45x_2 - 0,20x_3 = 1,97 \\ 0,30x_1 + 0,25x_2 + 0,43x_3 = 0,32 \\ 0,60x_1 - 0,35x_2 - 0,25x_3 = 1,83 \end{cases}$	2	$\begin{cases} 14,38x_1 - 2,41x_2 + 1,39x_3 = 5,86 \\ 1,84x_1 + 25,36x_2 - 3,31x_3 = -2,28 \\ 2,46x_1 - 3,49x_2 + 16,37x_3 = 4,47 \end{cases}$
3	$\begin{cases} 1,53x_1 - 1,65x_2 - 0,76x_3 = 2,18 \\ 0,86x_1 + 1,17x_2 + 1,84x_3 = 1,95 \\ 0,32x_1 - 0,65x_2 + 1,11x_3 = -0,47 \end{cases}$	4	$\begin{cases} 2,34x_1 - 4,21x_2 - 1,61x_3 = 14,41 \\ 8,04x_1 + 5,22x_2 + 0,27x_3 = -6,44 \\ 3,92x_1 - 7,99x_2 + 8,37x_3 = 55,56 \end{cases}$
5	$\begin{cases} 0,45x_1 - 0,94x_2 - 0,15x_3 = -0,15 \\ -0,01x_1 + 0,34x_2 + 0,06x_3 = 0,31 \\ -0,35x_1 + 0,05x_2 + 0,65x_3 = 0,37 \end{cases}$	6	$\begin{cases} 1,02x_1 - 0,73x_2 - 9,11x_3 = -1,25 \\ 6,25x_1 + 2,32x_2 + 7,62x_3 = 2,33 \\ 1,13x_1 - 8,88x_2 + 4,64x_3 = -3,75 \end{cases}$
7	$\begin{cases} 0,63x_1 + 0,05x_2 + 0,15x_3 = 0,34 \\ 0,15x_1 + 0,10x_2 + 0,71x_3 = 0,42 \\ 0,03x_1 + 0,34x_2 + 0,10x_3 = 0,32 \end{cases}$	8	$\begin{cases} 0,62x_1 + 0,92x_2 + 0,03x_3 = -0,82 \\ 0,99x_1 + 0,01x_2 + 0,07x_3 = 0,66 \\ 1,01x_1 - 0,02x_2 + 0,99x_3 = -0,98 \end{cases}$
9	$\begin{cases} -0,20x_1 + 1,60x_2 - 0,10x_3 = 0,30 \\ -0,30x_1 + 0,10x_2 - 1,50x_3 = 0,40 \\ 1,20x_1 - 0,20x_2 + 0,30x_3 = -0,60 \end{cases}$	10	$\begin{cases} 0,10x_1 - 0,07x_2 - 0,96x_3 = -2,04 \\ 0,04x_1 - 0,99x_2 - 0,85x_3 = -3,73 \\ 0,91x_1 + 1,04x_2 + 0,19x_3 = -1,67 \end{cases}$
11	$\begin{cases} 0,30x_1 + 1,20x_2 - 0,20x_3 = -0,60 \\ -0,10x_1 - 0,20x_2 + 1,60x_3 = 0,30 \\ 0,50x_1 + 0,34x_2 + 0,10x_3 = 0,32 \end{cases}$	12	$\begin{cases} 0,62x_1 + 0,84x_2 + 0,77x_3 = -8,18 \\ 0,03x_1 - 1,11x_2 - 1,08x_3 = 0,08 \\ 0,97x_1 + 0,02x_2 - 1,08x_3 = 0,06 \end{cases}$

№	СЛАУ	№	СЛАУ
13	$\begin{cases} 0,20x_1 + 0,44x_2 + 0,81x_3 = 0,74 \\ 0,58x_1 + 0,29x_2 + 0,05x_3 = 0,02 \\ 0,05x_1 + 0,34x_2 + 0,10x_3 = 0,32 \end{cases}$	14	$\begin{cases} 0,63x_1 - 0,37x_2 + 1,76x_3 = -9,29 \\ 0,90x_1 + 0,99x_2 + 0,05x_3 = 0,12 \\ 0,13x_1 - 0,95x_2 + 0,69x_3 = 0,69 \end{cases}$
15	$\begin{cases} 6,34x_1 + 11,75x_2 + 10x_3 = -41,40 \\ 7,42x_1 - 19,03x_2 + 11,75x_3 = -49,49 \\ 5,57x_1 + 7,48x_2 + 6,36x_3 = -27,67 \end{cases}$	16	$\begin{cases} 0,98x_1 + 0,88x_2 - 0,24x_3 = 1,36 \\ 0,16x_1 - 0,44x_2 - 0,88x_3 = -1,27 \\ 9,74x_1 - 10x_2 + 1,74x_3 = -5,31 \end{cases}$
17	$\begin{cases} 0,13x_1 - 0,14x_2 - 2,00x_3 = 0,15 \\ 0,75x_1 + 0,18x_2 + 0,77x_3 = 0,11 \\ 0,28x_1 - 0,17x_2 + 0,39x_3 = 0,12 \end{cases}$	18	$\begin{cases} 0,21x_1 - 0,94x_2 - 0,94x_3 = -0,25 \\ 0,98x_1 - 0,19x_2 + 0,93x_3 = 0,23 \\ 0,87x_1 + 0,56x_2 - 0,14x_3 = 0,33 \end{cases}$
19	$\begin{cases} 56,43x_1 - 8,54x_2 + 6,36x_3 = 9,76 \\ 4,34x_1 + 49,87x_2 + 9,18x_3 = 43,48 \\ 6,75x_1 - 8,93x_2 + 48,88x_3 = 56,92 \end{cases}$	20	$\begin{cases} 3,43x_1 + 4,07x_2 - 1,06x_3 = 46,08 \\ 74,4x_1 + 1,84x_2 - 1,85x_3 = -26,5 \\ 3,34x_1 + 94,3x_2 + 1,02x_3 = 92,3 \end{cases}$
21	$\begin{cases} 0,66x_1 + 0,44x_2 + 0,22x_3 = -0,58 \\ 1,54x_1 + 0,74x_2 + 1,54x_3 = -0,32 \\ 1,42x_1 + 1,42x_2 + 0,86x_3 = 0,83 \end{cases}$	22	$\begin{cases} 0,72x_1 + 3,54x_2 + 7,28x_3 = 0,33 \\ -0,28x_1 - 0,72x_2 + 3,04x_3 = 0,22 \\ 1,00x_1 + 0,35x_2 - 0,78x_3 = 1,12 \end{cases}$
23	$\begin{cases} 0,78x_1 - 0,02x_2 - 0,12x_3 = 0,56 \\ 0,02x_1 - 0,86x_2 + 0,04x_3 = 0,77 \\ 0,12x_1 + 0,44x_2 - 0,72x_3 = 1,01 \end{cases}$	24	$\begin{cases} 0,34x_1 + 0,71x_2 + 0,63x_3 = 2,08 \\ 0,71x_1 - 0,65x_2 - 0,17x_3 = 0,18 \\ 1,18x_1 - 2,35x_2 + 0,75x_3 = 1,28 \end{cases}$
25	$\begin{cases} -3x_1 + 0,5x_2 + 0,5x_3 = -56,5 \\ 0,5x_1 - 6x_2 + 0,5x_3 = -100 \\ 6,5x_1 + 0,6x_2 - 3x_3 = -210 \end{cases}$	26	$\begin{cases} 0,21x_1 - 0,18x_2 + 0,75x_3 = 0,11 \\ 0,13x_1 + 0,75x_2 - 0,11x_3 = 2,01 \\ 3,01x_1 - 0,33x_2 + 0,11x_3 = 0,13 \end{cases}$
27	$\begin{cases} 0,92x_1 - 0,83x_2 + 0,62x_3 = 2,15 \\ 0,24x_1 - 0,54x_2 + 0,43x_3 = 0,62 \\ 0,73x_1 - 0,81x_2 - 0,67x_3 = 0,88 \end{cases}$	28	$\begin{cases} 3,75x_1 - 0,28x_2 + 0,17x_3 = 0,75 \\ 2,11x_1 - 0,11x_2 - 0,12x_3 = 1,11 \\ 0,22x_1 - 3,17x_2 + 1,81x_3 = 0,05 \end{cases}$

№	СЛАУ	№	СЛАУ
29	$\begin{cases} 1,02x_1 + 0,72x_2 - 0,65x_3 = 1,27 \\ 0,74x_1 - 1,24x_2 - 1,73x_3 = 0,77 \\ 1,78x_1 + 2,32x_2 + 0,74x_3 = 1,16 \end{cases}$	30	$\begin{cases} 3,14x_1 - 2,12x_2 + 1,17x_3 = 1,27 \\ -2,12x_1 + 1,32x_2 - 2,45x_3 = 2,13 \\ 1,17x_1 - 2,45x_2 + 1,18x_3 = 3,14 \end{cases}$
31	$\begin{cases} 4,03x_1 + 2,71x_2 - 2,32x_3 = -1,60 \\ -2,45x_1 + 5,28x_2 - 0,36x_3 = 5,36 \\ 1,42x_1 + 2,91x_2 + 12,37x_3 = 5,75 \end{cases}$	32	$\begin{cases} 1,65x_1 - 2,27x_2 + 0,18x_3 = 2,25 \\ -2,27x_1 + 1,73x_2 - 0,46x_3 = 0,93 \\ 0,18x_1 - 0,46x_2 + 2,16x_3 = 1,33 \end{cases}$
33	$\begin{cases} -3,45x_1 - 1,25x_2 + 0,38x_3 = 5,01 \\ 2,15x_1 + 7,24x_2 - 0,39x_3 = 3,56 \\ 8,61x_1 + 12,94x_2 + 22,92x_3 = -4,72 \end{cases}$	34	$\begin{cases} 2,45x_1 + 1,75x_2 - 3,24x_3 = 1,23 \\ 1,75x_1 - 1,16x_2 + 2,18x_3 = 3,43 \\ -3,24x_1 + 2,18x_2 - 1,85x_3 = -0,16 \end{cases}$
35	$\begin{cases} -5,24x_1 + 2,66x_2 - 2,39x_3 = 9,11 \\ 2,47x_1 + 8,20x_2 + 2,31x_3 = 7,76 \\ 5,45x_1 - 6,27x_2 + 9x_3 = -9,37 \end{cases}$	36	$\begin{cases} 3,23x_1 + 1,62x_2 + 0,65x_3 = 1,28 \\ 1,62x_1 - 2,33x_2 - 1,43x_3 = 0,87 \\ 0,65x_1 - 1,43x_2 + 2,18x_3 = -2,87 \end{cases}$
37	$\begin{cases} -3,3x_1 + 2,1x_2 - 4,3x_3 = -0,21 \\ 4x_1 - 3,2x_2 + 5x_3 = 6 \\ 2x_1 + 1,23x_2 + 3,5x_3 = -1,2 \end{cases}$	38	$\begin{cases} 5,4x_1 - 2,46x_2 + 3,9x_3 = 5,51 \\ 2,57x_1 + 6,28x_2 - 1,3x_3 = 4,45 \\ 2,71x_1 - 0,76x_2 + 1,59x_3 = -3,57 \end{cases}$
39	$\begin{cases} 7,6x_1 + 5,8x_2 + 4,7x_3 = 10,01 \\ 3,8x_1 + 4,1x_2 + 2,7x_3 = 9,7 \\ 2,9x_1 + 2,1x_2 + 3,89x_3 = 7,37 \end{cases}$	40	$\begin{cases} 0,9x_1 + 2,7x_2 - 3,9x_3 = 2,41 \\ 2,51x_1 + 5,86x_2 - 0,5x_3 = 3,96 \\ 4,45x_1 - 2,57x_2 + 3,9x_3 = -1,28 \end{cases}$

11*. Написать программу, реализующую:

а) метод AA^T -минимальных итераций (A – произвольная матрица; минимизируется квадрат евклидовой нормы ошибки).

Инициализация: x_0 – произвольно, $r_0 = Ax_0 - b$, $s_1 = A^T r_0$;

Цикл: $r_i = r_{i-1} + \alpha_i As_i$, $s_{i+1} = A^T r_i + \beta_i s_i$, $x_i = x_{i-1} + \alpha_i s_i$,

где $\alpha_i = -\frac{(r_{i-1}, r_{i-1})}{(As_i, r_{i-1})} = -\frac{(r_{i-1}, r_{i-1})}{(s_i, s_i)}$, $\beta_i = -\frac{(r_i, As_i)}{(s_i, s_i)} = \frac{(r_i, r_i)}{(r_{i-1}, r_{i-1})}$;

б) метод $A^T A$ -минимальных итераций (A – произвольная матрица; минимизируется квадрат евклидовой нормы вектора невязки). Алгоритм счета такой же, как в предыдущем методе, но с числами

$$\alpha_i = -\frac{(A^T r_{i-1}, A^T r_{i-1})}{(As_i, As_i)}, \quad \beta_i = \frac{(A^T r_i, A^T r_i)}{(A^T r_{i-1}, A^T r_{i-1})}.$$

12*. Пусть $Ax = b$ – решаемая система. Применить метод сопряженных градиентов к эквивалентной системе $(L^{-1}AL^{-T})u = g$, где $u = L^T x$, $g = L^{-1}b$, L – предобусловленная, например, диагональная с элементами $l_{ii} = \sqrt{a_{ii}}$, матрица. Исследовать обусловленность старой и новой систем.

13*. Найти общее решение и фундаментальную систему решений для систем уравнений

$$\begin{cases} x_1 + 2x_2 + 4x_3 - 3x_4 = 0, \\ 3x_1 + 5x_2 + 6x_3 - 4x_4 = 0, \\ 4x_1 + 5x_2 - 2x_3 + 3x_4 = 0, \\ 3x_1 + 8x_2 + 24x_3 - 19x_4 = 0; \end{cases} \quad \begin{cases} 3x_1 + 4x_2 + x_3 + 2x_4 + 3x_5 = 0, \\ 5x_1 + 7x_2 + x_3 + 3x_4 + 4x_5 = 0, \\ 4x_1 + 5x_2 + 2x_3 + x_4 + 5x_5 = 0, \\ 7x_1 + 10x_2 + x_3 + 6x_4 + 5x_5 = 0. \end{cases}$$

14*. Исследовать совместность и найти общее решение

$$\begin{cases} x_1 + x_2 + 3x_3 - 2x_4 + 3x_5 = 1, \\ 2x_1 + 2x_2 + 4x_3 - x_4 + 3x_5 = 2, \\ 3x_1 + 3x_2 + 5x_3 - 2x_4 + 3x_5 = 1, \\ 2x_1 + 2x_2 + 8x_3 - 3x_4 + 9x_5 = 2; \end{cases} \quad \begin{cases} x_1 + 2x_2 + 3x_3 - 2x_4 + x_5 = 4, \\ 3x_1 + 6x_2 + 5x_3 - 4x_4 + 3x_5 = 5, \\ x_1 + 2x_2 + 7x_3 - 4x_4 + x_5 = 11, \\ 2x_1 + 4x_2 + 2x_3 - 3x_4 + 3x_5 = 6. \end{cases}$$

3. ЧИСЛЕННОЕ ДИФФЕРЕНЦИРОВАНИЕ

К численному дифференцированию чаще всего прибегают, когда приходится вычислять производные от функций, заданных таблично, или когда непосредственное дифференцирование функции затруднительно. Кроме того, формулы численного дифференцирования широко используются при разработке вычислительных методов решения многих задач (решение дифференциальных уравнений, поиск решений нелинейных уравнений, поиск точек экстремума функций и др.).

Пусть функция $f(x)$ задана таблицей своих значений в точках $x_0, x_1, \dots, x_N$. Выбрав из таблицы определенное множество $n+1$ узлов ($n \leq N$), построим по ним интерполяционный многочлен $P_n(x)$ для функции $f(x)$. Тогда в определенных границах изменения x производные от многочлена $P_n(x)$ могут быть использованы для приближенного вычисления соответствующих производных функции $f(x)$

$$f^{(m)}(x) \approx P_n^{(m)}(x), \quad m \leq n.$$

Пусть заданы равноотстоящие узлы x_k и значения функции $f(x_k) = y_k$. Конечными разностями первого порядка называются величины

$$\Delta f(x_k) = \Delta y_k = y_{k+1} - y_k, \quad k = 0, 1, 2, \dots$$

Конечные разности второго порядка определяются равенствами

$$\Delta^2 y_k = \Delta(\Delta y_k) = \Delta y_{k+1} - \Delta y_k, \quad k = 0, 1, 2, \dots$$

Конечные разности n -го порядка определяются через разности $(n-1)$ -го порядка по формуле

$$\Delta^n y_k = \Delta(\Delta y_k) = \Delta^{n-1} y_{k+1} - \Delta^{n-1} y_k, \quad k = 0, 1, 2, \dots$$

Конечные разности часто вычисляются по табл. 3.1. Имеет место равенство

$$f(x_n) = y_n = y_0 + \frac{n}{1!} \Delta y_0 + \frac{n(n-1)}{2!} \Delta^2 y_0 + \frac{n(n-1)(n-2)}{3!} \Delta^3 y_0 + \dots + \Delta^n y_0.$$

Таблица 3.1

k	y_k	Δy_k	$\Delta^2 y_k$	$\Delta^3 y_k$	$\Delta^4 y_k$	...
0	y_0	Δy_0				
1	y_1	Δy_1	$\Delta^2 y_0$	$\Delta^3 y_0$		
2	y_2	Δy_2	$\Delta^2 y_1$	$\Delta^3 y_1$	$\Delta^4 y_0$	
3	y_3	Δy_3	$\Delta^2 y_2$	$\Delta^3 y_2$	$\Delta^4 y_1$	
4	y_4	Δy_4	$\Delta^2 y_3$			
5	y_5					
...	...	...	...	...	...	...

Разностными отношениями первого порядка называются величины

$$f(x_0, x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \quad f(x_1, x_2) = \frac{f(x_2) - f(x_1)}{x_2 - x_1}, \quad \dots,$$

узлы x_k произвольные. По ним составляются разностные отношения второго порядка

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

$$f(x_1, x_2, x_3) = \frac{f(x_2, x_3) - f(x_1, x_0)}{x_3 - x_1}, \quad \dots$$

Разностные отношения порядка n ($n = 1, 2, \dots$) определяются при помощи разностных отношений предыдущего порядка $n-1$ по формуле

$$f(x_0, x_1, \dots, x_n) = \frac{f(x_1, x_2, \dots, x_n) - f(x_0, x_1, \dots, x_{n-1})}{x_n - x_0}.$$

При любом n верно равенство

$$f(x_0, x_1, \dots, x_n) = \sum_{i=0}^n \frac{f(x_i)}{(x_i - x_0) \cdots (x_i - x_{i-1})(x_i - x_{i+1}) \cdots (x_i - x_n)},$$

которое позволяет утверждать, что разностные отношения являются симметричными функциями своих узлов.

Для равноотстоящих узлов $x_k = x_0 + kh$ ($k = 0, 1, 2, \dots$) имеет место соотношение между разностными отношениями и конечными разностями

$$f(x_0, x_1, \dots, x_k) = \frac{\Delta^k y_0}{h^k k!}, \quad k = 0, 1, 2, \dots$$

Конечная разность порядка n от многочлена степени n равна постоянной величине, и, следовательно, все разности более высокого порядка равны нулю. Аналогичное свойство имеет место для разностных отношений. Таблица разностных отношений имеет вид табл. 3.2.

Таблица 3.2

k	x_k	$f(x_k)$	$\Delta^2 y_k$	$\Delta^3 y_k$	$\Delta^4 y_k$	$\Delta^5 y_k$	...
0	x_0	f_0					
1	x_1	f_1	$f(x_0, x_1)$				
2	x_2	f_2	$f(x_1, x_2)$	$f(x_0, x_1, x_2)$			
3	x_3	f_3	$f(x_2, x_3)$	$f(x_1, x_2, x_3)$	$f(x_0, x_1, x_2, x_3)$	$f(x_0, x_1, x_2, x_3, x_4)$	
4	x_4	f_4	$f(x_3, x_4)$	$f(x_2, x_3, x_4)$	$f(x_1, x_2, x_3, x_4)$	$f(x_1, x_2, x_3, x_4, x_5)$	
5	x_5	f_5	$f(x_4, x_5)$	$f(x_3, x_4, x_5)$	$f(x_2, x_3, x_4, x_5)$	...	
...	...	...	...	...	...		

С помощью разностных отношений строится формула интерполяционного многочлена – интерполяционный многочлен Ньютона

$$P_n(x) = f(x_0) + (x - x_0)f(x_0, x_1) + (x - x_0)(x - x_1)f(x_0, x_1, x_2) + \dots \\ \dots + (x - x_0)(x - x_1)\dots(x - x_{n-1})f(x_0, x_1, \dots, x_n), \quad (3.1)$$

где $P_n(x_k) = f(x_k)$ ($k = 0, 1, 2, \dots, n$). Остаточный член $r_n(x) = f(x) - P_n(x)$ в этом случае можно выразить через разделенную разность:

$$r_n(x) = f(x, x_0, x_1, \dots, x_n)(x - x_0)(x - x_1)\dots(x - x_n). \quad (3.2)$$

Формулой Ньютона удобно пользоваться при интерполировании одной функции с меняющейся системой узлов: при добавлении нового узла x_{n+1} нужно вычислить только одно слагаемое $(x - x_0) \dots (x - x_n) \cdot f(x_0, x_1, \dots, x_{n+1})$ и добавить его к предыдущей сумме.

В случае равноотстоящих узлов имеется много различных формул, построение которых зависит от расположения точки интерполирования x по отношению к узлам интерполирования.

Пусть функция $f(x)$ задана таблицей значений $f(x_k) = y_k$ в равноотстоящих точках $x_k = x_0 + kh$ ($k = 0, 1, 2, \dots$) и точка интерполирования x находится вблизи от начальной точки x_0 . Введем новую переменную t , положив $x = x_0 + th$, $t = (x - x_0)/h$. Используя (3.2) и соотношения (3.1) между разностными отношениями и конечными разностями, получим интерполяционную формулу Ньютона для интерполирования в начале таблицы

$$\begin{aligned} f(x_0 + th) = & y_0 + \frac{t}{1!} \Delta y_0 + \frac{t(t-1)}{2!} \Delta^2 y_0 + \dots \\ & \dots + \frac{t(t-1) \dots (t-k+1)}{k!} \Delta^k y_0 + r_k(x). \end{aligned} \quad (3.3)$$

Если точка интерполирования находится вблизи конца таблицы узлов, то узлы интерполирования естественно брать в порядке $x_n, x_n - h, x_n - 2h, \dots$. Введя переменную t и положив $x = x_n + th$, получим формулу Ньютона для интерполирования в конце таблицы

$$\begin{aligned} f(x_n + th) = & y_n + \frac{t}{1!} \Delta y_{n-1} + \frac{t(t+1)}{2!} \Delta^2 y_{n-2} + \dots \\ & \dots + \frac{t(t+1) \dots (t+k-1)}{k!} \Delta^k y_{n-k} + r_k(x). \end{aligned} \quad (3.4)$$

Если точка интерполирования лежит вблизи некоторого внутреннего узла x_n , то узлы интерполирования целесообразно брать в порядке удаленности от x_n : $x_n - h, x_n + h, \dots, x_n - kh, x_n + kh$, а сама формула (Стирлинга) имеет вид

$$f(x_n + th) = y_n + \frac{t}{1!} \frac{\Delta y_{n-1} + \Delta y_n}{2} + \frac{t^2}{2!} \Delta^2 y_{n-1} + \dots$$

$$\dots + \frac{t(t^2 - 1^2)}{3!} \frac{\Delta^3 y_{n-2} + \Delta^3 y_{n-1}}{2} + \frac{t^2(t^2 - 1^2)}{4!} \Delta^4 y_{n-2} + \dots$$

Дифференцируя эти многочлены по переменной x с учетом правила дифференцирования сложной функции, можно получить формулы для вычисления производных. Например, применяя это действие к формуле (3.3), будем иметь

$$y' \approx \frac{1}{h} \left(\Delta y_0 + \frac{2t-1}{2!} \Delta^2 y_0 + \frac{3t^2-6t+2}{3!} \Delta^3 y_0 + \right.$$

$$\left. + \frac{4t^3-18t^2+22t-6}{4!} \Delta^4 y_0 + \frac{5t^4-40t^3+105t^2-100t+24}{5!} \Delta^5 y_0 + \dots \right),$$

$$y'' \approx \frac{1}{h^2} \left[\Delta^2 y_0 + (t-1) \Delta^3 y_0 + \left(\frac{1}{2} t^2 - \frac{3}{2} t + \frac{11}{12} \right) \Delta^4 y_0 + \right.$$

$$\left. + \left(\frac{1}{6} t^3 - t^2 + \frac{7}{4} t - \frac{5}{6} \right) \Delta^5 y_0 + \dots \right].$$

Интерполяционные многочлены Ньютона дают выражения для производных через конечные разности. Однако при компьютерных вычислениях часто выгоднее выражать производные выражать не через конечные разности функции, а непосредственно через значения функции в узлах.

Запишем интерполяционный многочлен Лагранжа и его остаточный член для случая трех узлов интерполяции и найдем их производные

$$L(x) = \frac{1}{2h^2} [(x-x_1)(x-x_2)y_0 - 2(x-x_0)(x-x_1)y_1 + (x-x_0)(x-x_1)y_2],$$

$$R_L(x) = \frac{y_*'''}{3!} (x-x_0)(x-x_1)(x-x_2),$$

$$L'(x) = \frac{1}{2h^2} [(2x-x_1-x_2)y_0 - 2(2x-x_0-x_2)y_1 + (2x-x_0-x_1)y_2],$$

$$R'_L(x) = \frac{y_*'''}{3!} [(x-x_1)(x-x_2) + (x-x_0)(x-x_2) + (x-x_0)(x-x_1)].$$

Здесь y_*''' – значение производной третьего порядка в некоторой внутренней точке $x^* \in [x_0, x_2]$.

Запишем выражение для производной $y'(x)$ при $x=x_0$:

$$\begin{aligned} y'_0 &= L'(x) + R'_L(x) = \\ &= \frac{1}{2h^2} [(2x-x_1-x_2)y_0 - 2(2x-x_0-x_2)y_1 + (2x-x_0-x_1)y_2] + \\ &\quad + \frac{y_*'''}{3!} (x_0-x_1)(x_0-x_2) = \frac{1}{2h} (-3y_0 + 4y_1 - y_2) + \frac{h^2}{3} y_*'''. \end{aligned}$$

Аналогичные соотношения получаются при $x=x_1$ и $x=x_2$ для значений y'_1 и y'_2 соответственно

$$\begin{aligned} y'_1 &= \frac{1}{2h} (y_2 - y_0) - \frac{h^2}{6} y_*''', \\ y'_2 &= \frac{1}{2h} (y_0 - 4y_1 + 3y_2) + \frac{h^2}{3} y_*'''. \end{aligned}$$

Для каждой из этих формул значения y_*''' , вообще говоря, различны.

Используя интерполяционный многочлен Лагранжа и его остаточный член для случая четырех узлов ($n=3$), получим соответствующие аппроксимации производных:

$$\begin{aligned} y'_0 &= \frac{1}{6h} (-11y_0 + 18y_1 - 9y_2 + 2y_3) - \frac{h^3}{4} y_*^{\text{IV}}, \\ y'_1 &= \frac{1}{6h} (-2y_0 - 3y_1 + 6y_2 - y_3) + \frac{h^3}{12} y_*^{\text{IV}}, \\ y'_2 &= \frac{1}{6h} (y_0 - 6y_1 + 3y_2 + 2y_3) - \frac{h^3}{12} y_*^{\text{IV}}, \\ y'_3 &= \frac{1}{6h} (-2y_0 + 9y_1 - 18y_2 + 11y_3) + \frac{h^3}{4} y_*^{\text{IV}}. \end{aligned}$$

В случае пяти узлов получим

$$y'_0 = \frac{1}{12h}(-25y_0 + 48y_1 - 36y_2 + 16y_3 - 3y_4) + \frac{h^4}{5}y_*^V,$$

$$y'_1 = \frac{1}{12h}(-3y_0 - 10y_1 + 18y_2 - 6y_3 + y_4) - \frac{h^4}{20}y_*^V,$$

$$y'_2 = \frac{1}{12h}(y_0 - 8y_1 + 8y_3 - y_4) + \frac{h^4}{30}y_*^V,$$

$$y'_3 = \frac{1}{12h}(y_0 + 6y_1 - 18y_2 + 10y_3 + 3y_4) + \frac{h^4}{20}y_*^V,$$

$$y'_4 = \frac{1}{12h}(3y_0 - 16y_1 + 36y_2 - 48y_3 + 25y_4) + \frac{h^5}{5}y_*^V.$$

Таким образом, используя значения функции в $n+1$ узле, получаем аппроксимацию производных n -го порядка точности. Эти формулы можно использовать не только для узлов $x_1, x_2, \dots$, но и для любых узлов $x_i, x_{i+1}, \dots$, соответствующим образом изменяя значения индексов. Нетрудно убедиться, что наиболее простые выражения, да к тому же с наименьшими коэффициентами в остаточных членах, получаются для производных в центральных узлах при четных n .

$$y'_1 = \frac{1}{2h}(y_2 - y_0) - \frac{h^2}{6}y_*''', \quad n=2,$$

$$y'_2 = \frac{1}{12h}(y_0 - 8y_1 + 8y_3 - y_4) + \frac{h^4}{30}y_*^V, \quad n=4.$$

Они называются аппроксимациями производных центральными разностями и широко используются на практике.

На основе интерполяционных многочленов можно получить аппроксимации для старших производных, например для вторых производных:

$$y_0'' = \frac{1}{h^2}(y_2 - 2y_1 + y_0) + O(h),$$

$$y_1'' = \frac{1}{h^2}(y_2 - 2y_1 + y_0) + O(h^2),$$

$$y_2'' = \frac{1}{h^2}(y_2 - 2y_1 + y_0) + O(h).$$

В случае четырехточечной аппроксимации получим

$$y_0'' = \frac{1}{h^2}(2y_0 - 5y_1 + 4y_2 - y_3) + O(h^2),$$

$$y_1'' = \frac{1}{h^2}(y_0 - 2y_1 + y_2) + O(h^2),$$

$$y_2'' = \frac{1}{h^2}(y_1 - 2y_2 + y_3) + O(h^2),$$

$$y_3'' = \frac{1}{h^2}(-y_0 + 4y_1 - 5y_2 + 2y_3) + O(h^2).$$

Для пяти узлов имеем

$$y_0'' = \frac{1}{12h^2}(35y_0 - 104y_1 + 114y_2 - 56y_3 + 11y_4) + O(h^3),$$

$$y_1'' = \frac{1}{12h^2}(11y_0 - 20y_1 + 6y_2 + 4y_3 - y_4) + O(h^3),$$

$$y_2'' = \frac{1}{12h^2}(-y_0 + 16y_1 - 30y_2 + 16y_3 - y_4) + O(h^4),$$

$$y_3'' = \frac{1}{12h^2}(-y_0 + 4y_1 + 6y_2 - 20y_3 + 11y_4) + O(h^3),$$

$$y_4'' = \frac{1}{12h^2}(11y_0 - 56y_1 + 114y_2 - 104y_3 + 35y_4) + O(h^3).$$

Аппроксимации вторых производных с помощью центральных разностей при четных n также наиболее выгодны.

Пример. Продифференцировать функцию, заданную в виде таблицы с равномерным шагом.

x_k	0,0	0,25	0,5	0,75	1,0	1,25	1,50	1,75	2,0
y_k	1,849	1,960	2,028	2,048	2,020	1,944	1,825	1,670	1,489

Решение.

1 способ. Локальное применение интерполяционных формул типа Ньютона на соответствующих участках отрезка $[a, b]$.

```
> restart;
> n:=9: X:=vector(n); Y:=vector(n);
```

Вводим массивы координат точек:

```
> x:=[0, 0.25, 0.5, 0.75, 1., 1.25, 1.5, 1.75, 2];
> y:=[1.849, 1.960, 2.028, 2.048, 2.020, 1.944,
1.825, 1.670, 1.489];
```

Начало и конец отрезка $[a, b]$, на котором задана функция, и шаг h изменения абсцисс $\{x_k\}$:

```
> a:=0: b:=2: h:=(b-a)/(n-1);
```

Определяем конечные разности вплоть до 4-го порядка:

```
> dY:=[seq(Y[i+1]-Y[i], i=1..n-1)];
> dY2:=[seq(dY[i+1]-dY[i], i=1..n-2)];
> dY3:=[seq(dY2[i+1]-dY2[i], i=1..n-3)];
> dY4:=[seq(dY3[i+1]-dY3[i], i=1..n-4)];
```

Вблизи левой границы интервала изменения x используем интерполяционную формулу (3.3) при $k = 4$:

```
> fleft:=Y[1]+t*dY[1]+t*(t-1)/2*dY2[1]+t*(t-1)*
(t-2)/6*dY3[1]+t*(t-1)*(t-2)*(t-3)/24*dY4[1];
```

Аналогично для конечной части интервала применяется формула (3.4):

```
> fright:=Y[n]+t*dY[n-1]+t*(t+1)/2*dY2[n-2]+
t*(t+1)*(t+2)/6*dY3[n-3]+
t*(t+1)*(t+2)*(t+3)/24*dY4[n-4];
```

Для аппроксимации центральной части промежутка $[0, 2]$ воспользуемся формулой Стирлинга:

```
> m:=5; # номер узла, около которого строится аппроксимация
```

```
> fmid:=Y[m]+t*(dY[m]+dY[m-1])/2+t^2/2*dY2[m-1]+
  t*(t^2-1)/6*(dY3[m-1]+dY3[m-2])/2+
  (t^2)*(t^2-1)/24*dY4[m-2];
```

В найденных функциях перейдем к переменной x

```
> g1:=subs(t=(x-a)/h,fleft);
g2:=subs(t=(x-X[m])/h,fmid);
g3:=subs(t=(x-b)/h,fright);
```

Определим общую аппроксимирующую функцию в виде кусочной, сшивая три найденных аппроксимации, действующие на соответствующих промежутках:

```
> g:=piecewise(x<X[4],g1,x<X[6],g2,g3);
```

т.е. на участке от x_1 до x_4 берется **g1** (fleft), на участке от x_4 до x_6 – функция **g2** (fmid), на остальном промежутке – **g3** (fright).

Аналогично находим первую и вторую производные

```
> gd:=piecewise(x<X[4],diff(g1,x),x<X[6],
  diff(g2,x),diff(g3,x));
> gd2:=piecewise(x<X[4],diff(g1,x$2),x<X[6],
  diff(g2,x$2),diff(g3,x$2));
```

Вычертим графики аппроксимаций заданной функции и её производной.

```
> plot(g,x=a..b);
```

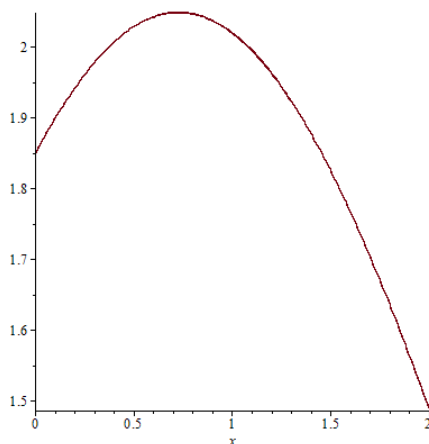


Рис. 3.1

```
> plot(gd,x=a..b);
```

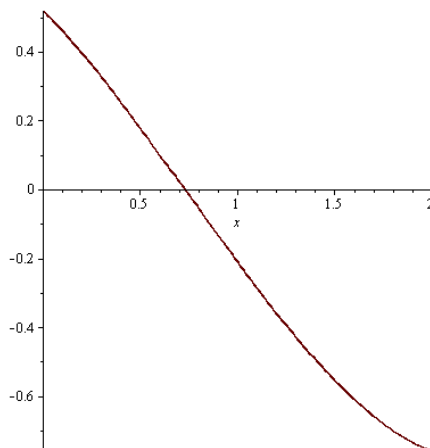


Рис. 3.2

Ниже приводится таблица значений функции и её производных.

```
> for t from a to b by h/2 do printf(`x=%5.3g
y=%8.5g    y'=%8.5g    y''=%g\n`,
t,subs(x=t,g),subs(x=t,gd),subs(x=t,gd2));od;
```

x= 0	y= 1.849	y'= 0.51833	y''=-0.534667
x=0.125	y= 1.9094	y'= 0.44567	y''=-0.624667
x= 0.25	y= 1.96	y'= 0.363	y''=-0.694667
x=0.375	y= 1.9998	y'= 0.27283	y''=-0.744667
x= 0.5	y= 2.028	y'= 0.17767	y''=-0.774667
x=0.625	y= 2.0441	y'= 0.08	y''=-0.784667
x= 0.75	y= 2.048	y'=-0.014333	y''=-0.774667
x=0.875	y= 2.0401	y'= -0.112	y''=-0.784667
x= 1	y= 2.02	y'= -0.20967	y''=-0.774667
x= 1.12	y= 1.9878	y'= -0.30483	y''=-0.744667
x= 1.25	y= 1.944	y'= -0.39367	y''=-0.692
x= 1.38	y= 1.8895	y'= -0.47717	y''=-0.642
x= 1.5	y= 1.825	y'= -0.55367	y''=-0.58
x= 1.62	y= 1.7514	y'= -0.62167	y''=-0.506
x= 1.75	y= 1.67	y'= -0.67967	y''=-0.42
x= 1.88	y= 1.582	y'= -0.72617	y''=-0.322
x= 2	y= 1.489	y'= -0.75967	y''=-0.212

2 способ. Использование формул аппроксимации производных по набору значений функции для близлежащих узлов.

Вводим исходные данные.

```
> restart;  
> n:=9: X:=vector(n);Y:=vector(n);  
> X:=[0,0.25,0.5,0.75,1.,1.25,1.5,1.75,2.];  
> Y:=[1.849,1.960,2.028,2.048,2.020,1.944,1.825,  
1.670, 1.489];
```

Сначала воспользуемся трёхточечной аппроксимацией производной.

```
> diF3:=vector(n);# массив для производных в узловых точках  
> diF3[1]:=(-3*Y[1]+4*Y[2]-Y[3])/(2*h);  
> for j from 2 to n-1 do  
    diF3[j]:=(Y[j+1]-Y[j-1])/(2*h);  
> od;  
> diF3[n]:=(Y[n-2]-4*Y[n-1]+3*Y[n])/(2*h);
```

В результате получим массив производных:

```
> eval(diF3);  
[0.5300000000,0.3580000000,0.1760000000,-0.01600000000,  
-0.2080000000,-0.3900000000,-0.5480000000,-0.6720000000,  
-0.7760000000]
```

Теперь вычислим производные в узловых точках на основе пятиточечной аппроксимации.

```
> diF5:=vector(n);  
> diF5[1]:=(-25*Y[1]+48*Y[2]-36*Y[3]+16*Y[4]-  
3*Y[5])/(12*h);  
> diF5[2]:=(-3*Y[1]-10*Y[2]+18*Y[3]-  
6*Y[4]+Y[5])/(12*h);  
> for j from 3 to n-2 do  
    diF5[j]:=(Y[j-2]-8*Y[j-1]+8*Y[j+1]-  
                Y[j+2])/(12*h);  
> od;  
> diF5[n-1]:=(-Y[n-4]+6*Y[n-3]-18*Y[n-2]+  
                10*Y[n-1]+3*Y[n])/(12*h);
```



```
> diF5[n] := (3*Y[n-4] - 16*Y[n-3] + 36*Y[n-2] -
              48*Y[n-1] + 25*Y[n]) / (12*h) ;
DiF51 := -0.7596666667
```

Вывод массива значений производных:

```
> eval(diF5) ;
[0.5300000000, 0.3580000000, 0.1760000000, -0.01600000000,
 -0.2080000000, -0.3900000000, -0.5480000000, -0.6720000000,
 -0.7760000000]
```

Вычисление вторых производных читателю предлагается провести самостоятельно.

Упражнения

Найти численно первые и вторые производные функции $y=f(x)$ по ее табулированным значениям на заданном отрезке с шагом $h = 0,1$. Сравнить с точным значением.

№	$y=f(x)$	$[a, b]$	№	$y=f(x)$	$[a, b]$
1	$y = x - \frac{1}{x}$	[2; 3]	2	$y = \operatorname{ch} 2x$	[0; 1]
3	$y = \frac{\operatorname{sh} 2x}{x}$	[0,1; 1,1]	4	$y = \frac{\sin 4x}{x}$	[1; 2]
5	$y = \frac{\operatorname{th} 4x}{x}$	[0;1]	6	$y = \frac{1}{1 + 4x^2}$	[-1; 0]
7	$y = \operatorname{tg} x - \frac{1}{x}$	[-1, 0]	8	$y = e^{-3,4x^2}$	[0; 1]
9	$y = \operatorname{tg} x - \frac{1}{x^2}$	[2; 3]	10	$y = \frac{1}{3x + 5}$	[-2; -1]
11	$y = \frac{x^3}{\sin x + 1,3}$	[0,1; 1,1]	12	$y = \lg x$	[0,1; 1,1]
13	$y = \frac{x^2 - 2}{\ln x + 5}$	[0,1; 1,1]	14	$y = \frac{\lg x}{5x}$	[0,5; 1,5]

№	$y=f(x)$	$[a, b]$	№	$y=f(x)$	$[a, b]$
15	$y = \lg x + 1,4$	$[0,5; 1,5]$	16	$y = \frac{-0,5}{3x + 0,2}$	$[0; 1]$
17	$y = \frac{5}{\ln x}$	$[1,1; 2,1]$	18	$y = \operatorname{sh} x$	$[2; 3]$
19	$y = x - \frac{1}{x^2}$	$[2; 3]$	20	$\frac{\operatorname{arctg} x}{x}$	$[0,2; 2,2]$

Указание. Ниже показан пример табулирования функции для $y = \sin x$ на отрезке $[0, 1]$ с шагом $h = 0,1$. Результатом является создание двух массивов – абсцисс и ординат точек $(x_k, y(x_k))$ кривой $y = \sin x$.

```
> a:=0: b:=1: h:=0.1: m:=(b-a)/h: n:=m+1:
> f:=x->sin(x):
> X:=[seq(a+(i-1)*h,i=1..n)];
      X:= [0., 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]
> Y:=[seq(f(X[i]),i=1..n)];
      Y:= [0., 0.09983341665, 0.1986693308, 0.2955202067,
            0.3894183423, 0.4794255386, 0.5646424734, 0.6442176872,
            0.7173560909, 0.7833269096, 0.8414709848]
```

4. ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ

4.1. Постановка задачи

Пусть на отрезке $[a, b]$ задана непрерывная функция $y=f(x)$. Точками $a = x_0, x_1, x_2, \dots, b = x_n$ разобьем исходный отрезок на n элементарных отрезков длиной $\Delta x_i = x_i - x_{i-1}$. На каждом из этих отрезков выберем произвольную точку ξ_i ($x_{i-1} \leq \xi_i \leq x_i$) и составим интегральную сумму

$$I = \sum_{i=1}^n f(\xi_i) \Delta x_i. \quad (4.1)$$

По определению

$$\int_a^b f(x) dx = \lim_{\max \Delta x_i \rightarrow 0} \sum_{i=1}^n f(\xi_i) \Delta x_i.$$

Как известно, геометрический смысл определенного интеграла от неотрицательной на промежутке $[a, b]$ функции $f(x)$ – площадь криволинейной трапеции. Для непрерывной функции $f(x)$ первообразная всегда существует, и определенный интеграл вычисляется по формуле Ньютона-Лейбница:

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

где $F(x)$ – одна из первообразных для данной функции $f(x)$.

Однако во многих случаях первообразная $F(x)$ не выражается в элементарных функциях и не может быть найдена с помощью стандартных средств или является слишком сложной. Например, таким способом не удастся вычислить интегралы

$$\int_a^b \frac{\sin x}{x} dx, \quad \int_a^b \frac{dx}{\ln x}, \quad \int_a^b e^{-x^2} dx, \quad \text{и т.д.}$$

Кроме того, на практике функция $f(x)$ часто задается таблично, и тогда само понятие первообразной теряет смысл. В таких случаях используются методы численного интегрирования, состоящие в том, что данную функцию $f(x)$ на отрезке $[a, b]$, заменяют аппроксимирующей (чаще интерполирующей)

функцией более простого вида. Такие специальные приближенные формулы для вычисления определенных интегралов называют квадратурными формулами или формулами численного интегрирования.

Наиболее часто приближенное значение интеграла ищут в виде линейной комбинации значений функции $f(x)$ на отрезке $[a, b]$:

$$\int_a^b f(x)dx \approx \sum_{k=1}^n A_k f(x_k). \quad (4.2)$$

Приближенное равенство (4.2) называют квадратурной формулой, определяемой узлами и весовыми коэффициентами (весами), выражение в правой части (4.2) называется квадратурной суммой, а разность

$$R_n(f) = \int_a^b f(x)dx - \sum_{k=1}^n A_k f(x_k)$$

называют остаточным членом или остатком этой квадратурной формулы.

Нередко возникает потребность нахождения численного значения интеграла, когда заранее известна или по каким-то причинам составляется таблица значений $f(x_k)$ интегрируемой функции $f(x)$ в некоторой системе точек x_k ($k = 1, 2, \dots, n$). Если A_k выбирать так, чтобы квадратурная формула была точной для всех многочленов наивысшей возможной степени, то придем к интерполяционным квадратурам. Источником получения таких формул могут служить интерполяционные многочлены Лагранжа. По заданной системе узлов x_k и таблице значений $f(x_k)$ ($k = 1, 2, \dots, n$) построим интерполяционный многочлен Лагранжа

$$L_{n-1}(x) = \sum_{k=1}^n f(x_k) l_k(x), \quad l_k(x) = \frac{\omega_k(x)}{\omega_k(x_k)},$$

где $\omega_k(x) = \frac{1}{x - x_k} \prod_{i=1}^n (x - x_i)$, и положим в интеграле (4.2)

$$f(x) \approx L_{n-1}(x).$$

Получим приближенное равенство

$$\int_a^b f(x)dx \approx \sum_{k=1}^n A_k f(x_k), \quad (4.3)$$

где коэффициенты представляют собой интегралы от полиномов $l_k(x)$, т.е.

$$A_k = \int_a^b l_k(x)dx. \quad (4.4)$$

Квадратурные формулы (4.3), у которых коэффициенты определяются равенствами (4.4), называются интерполяционными.

В случае равноотстоящих узлов интерполяционная квадратурная формула (4.3)-(4.4) может быть записана в виде

$$\int_a^b f(x)dx \approx (b-a) \sum_{k=1}^n B_k^n f(a+kh), \quad (4.5)$$

где

$$B_k^n = \frac{(-1)^{n-k}}{nk!(n-k)!} \int_0^n t(t-1)\dots(t-k+1)(t-k-1)\dots(t-n)dt,$$

а шаг, как обычно, $h = (b-a)/n$.

Квадратурные формулы (4.5) называют формулами Ньютона-Котеса.

Установлено, что при больших n в формуле (4.5) будут встречаться как положительные, так и отрицательные коэффициенты, превосходящие сколь угодно большое число. Отсюда, в частности, следует, что при больших n малые ошибки в значениях функции $f(a+kh)$ могут дать большую погрешность в квадратурной сумме. Поэтому формулы Ньютона-Котеса мало пригодны для вычислений при большом числе узлов.

На практике используют простейшие квадратурные формулы с небольшими числами n (обычно в пределах $1 \div 4$), что в редких случаях может дать удовлетворительную точность. Для уменьшения погрешности предварительно разбивают отрезок $[a, b]$ на достаточно большое число интервалов, и к каждому из них применяют простейшую квадратурную формулу.

4.2. Метод прямоугольников

Простые квадратурные формулы можно вывести непосредственно из определения интеграла, т.е. из формулы (4.1). Зафиксировав там некоторое $n \geq 1$, будем иметь

$$I \approx \sum_{i=1}^n f(\xi_i) \Delta x_i. \quad (4.6)$$

Это приближенное равенство называют общей формулой прямоугольников (площадь криволинейной трапеции приближенно заменяется площадью ступенчатой фигуры, составленной из прямоугольников, основаниями которых служат отрезки $[x_{i-1}, x_i]$, а высотами – ординаты $f(\xi_i)$) (рис. 4.1).

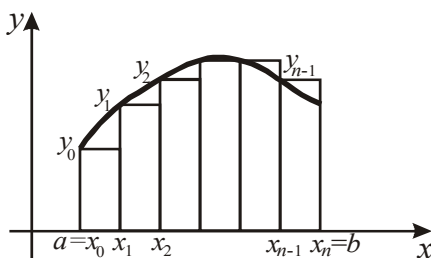


Рис. 4.1

Условимся в дальнейшем пользоваться равномерным разбиением отрезка $[a, b]$ на n частей точкам x_i с шагом $h = (b - a)/n$, полагая $x_0 = a$, $x_1 = a + h$, $x_2 = a + 2h$, ..., $x_{n-1} = a + (n - 1)h$, $x_n = b$.

При таком разбиении формула (4.6) приобретает вид

$$I \approx h \sum_{i=1}^n f(\xi_i), \quad \xi_i \in [x_{i-1}, x_i].$$

В качестве промежуточных точек ξ_i можно выбрать левые ($\xi_i = x_{i-1}$) или правые ($\xi_i = x_i$) концы элементарных отрезков $[x_{i-1}, x_i]$. Соответственно получим две интегральные суммы:

$$I \approx I_{\Pi-} = h \sum_{i=1}^n f(x_{i-1}) = h(y_0 + y_1 + y_2 + \dots + y_{n-2} + y_{n-1}), \quad (4.7)$$

$$I \approx I_{\Pi+} = h \sum_{i=1}^n f(x_i) = h(y_1 + y_2 + y_3 \dots + y_{n-1} + y_n), \quad (4.8)$$

которые называются квадратурными формулами левых и правых прямоугольников.

Часто применяется еще одна формула прямоугольников, использующая значения функции $f(x)$ в середине элементарных отрезков, т.е. $\xi_i = \frac{1}{2}(x_{i-1} + x_i)$. В результате имеем квадратурную формулу средних прямоугольников

$$\begin{aligned} I \approx I_{\Pi} &= h \sum_{i=1}^n f\left(x_{i-1} + \frac{h}{2}\right) = h \sum_{i=1}^n f\left(x_i - \frac{h}{2}\right) = \\ &= h[f(a + h/2) + f(a + h + h/2) + \dots + f(a + (n-1)h + h/2)]. \end{aligned} \quad (4.9)$$

Для формул (4.7) и (4.8) может быть получена следующая оценка погрешности

$$|R| \leq \frac{b-a}{2} h M_1, \quad (4.10)$$

а для формулы (4.9)

$$|R| \leq \frac{b-a}{24} h^2 M_2, \quad (4.11)$$

где M_1 и M_2 – наибольшее на отрезке $[a, b]$ значение модуля соответственно первой и второй производной функции $f(x)$:

$$M_1 = \max_{x \in [a, b]} |f'(x)|, \quad M_2 = \max_{x \in [a, b]} |f''(x)|.$$

Как видно из формулы (4.11), при увеличении числа n элементарных отрезков, на которые разбивается промежуток интегрирования $[a, b]$, погрешность интегрирования по формуле (4.9) убывает пропорционально квадрату шага h . А погрешность интегрирования по формулам (4.7)-(4.8) убывает лишь по линейному закону. Таким образом, формула средних прямоугольников имеет более высокий порядок точности.

Пример. Вычислить по формуле прямоугольников $I = \int_1^2 \sqrt{x} dx$, разбив интервал интегрирования на 10 частей.

Оценить погрешность.

Решение. Здесь $f(x) = \sqrt{x}$; отрезок интегрирования $[1, 2]$, при $n = 10$ имеем $h = \frac{2-1}{10} = 0.1$. Точками деления служат $x_0 = 1$, $x_1 = x_0 + h = 1.1$, $x_2 = x_0 + 2h = 1.2$, ..., $x_9 = 1.9$.

Найдем соответствующие значения подынтегральной функции: $y_0 = f(x_0) = \sqrt{1} = 1$, $y_1 = f(x_1) = \sqrt{1.1} = 1.049$, $y_2 = 1.095$, $y_3 = 1.140$, $y_4 = 1.183$, $y_5 = 1.225$, $y_6 = 1.265$, $y_7 = 1.304$, $y_8 = 1.342$, $y_9 = 1.378$.

Используя формулу левых прямоугольников, получим:

$$I_{\Pi} = 0.1(1 + 1.049 + 1.095 + 1.140 + 1.183 + 1.225 + \\ + 1.265 + 1.304 + 1.342 + 1.378) = 0.1 \cdot 11.981 \approx 1.20.$$

Оценим погрешность. В данном случае $f'(x) = \frac{1}{2\sqrt{x}}$; на отрезке $[1, 2]$ достигает наибольшего значения, равного 0.5, при $x = 1$. Таким образом, $|f'(x)| \leq M_1 = 1/2$. По формуле (13) находим: $|R| \leq \frac{1}{2} \cdot 0.1 \cdot \frac{1}{2} \approx 0.025$. Следовательно, $I = 1.20 \pm 0.025$.

По формуле средних прямоугольников находим

$$I_{\Pi} = h [f(1.05) + f(1.15) + \dots + f(1.95)] = 0.1(\sqrt{1.05} + \sqrt{1.15} + \sqrt{1.25} + \\ + \sqrt{1.35} + \sqrt{1.45} + \sqrt{1.55} + \sqrt{1.65} + \sqrt{1.75} + \sqrt{1.85} + \sqrt{1.95}) \approx 1.219.$$

$$f''(x) = -\frac{1}{4\sqrt{x^3}}, |f''(x)| \leq M_2 = 1/4, |R| \leq \frac{1}{24} \cdot 0.01 \cdot \frac{1}{4} \approx 10^{-4}.$$

Точное значение 1.21895.

4.3. Метод трапеций

Для получения других формул численного интегрирования, как говорилось ранее, подынтегральную функцию $f(x)$ на отрезке интегрирования $[a, b]$ заменяют интерполирующей или аппроксимирующей функцией более простого вида. Заменяем подынтегральную функцию $y=f(x)$ интерполяционным многочленом Лагранжа 1-го порядка

$$\begin{aligned} I &\approx \int_a^b \left(\frac{x-x_1}{x_0-x_1} y_0 + \frac{x-x_0}{x_1-x_0} y_1 \right) dx = \\ &= \frac{y_0}{(-h)} \int_{x_0}^{x_0+h} (x-x_1) dx + \frac{y_1}{h} \int_{x_0}^{x_0+h} (x-x_0) dx = \frac{h}{2} (y_0 + y_1) \quad (4.12) \end{aligned}$$

Здесь $a = x_0$, $b = x_1$, $h = x_1 - x_0$, $y_0 = f(x_0)$, $y_1 = f(x_1)$.

Геометрически этот результат совершенно очевиден. Действительно, подставляя вместо $f(x)$ интерполяционный многочлен первой степени, фактически заменяем участок кривой хордой (рис. 4.2). При этом интеграл заменяется площадью обычной прямолинейной трапеции. Очевидно, что формула (4.12) будет точной, если $f(x)$ – линейная функция.

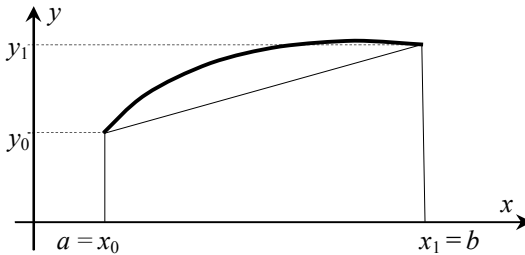


Рис. 4.2

Полученную формулу

$$\int_a^b f(x) dx \approx \frac{h}{2} (y_0 + y_1) = \frac{(b-a)}{2} [f(a) + f(b)] \quad (4.13)$$

называют квадратурной формулой трапеций. Она также может быть получена из общей формулы Ньютона-Котеса при $n=1$.

Для формулы (4.13) может быть получена следующая оценка погрешности

$$|R| \leq \frac{(x_1 - x_0)^3}{12} \max_{x \in [x_0, x_1]} |f''(x)|.$$

Так как ошибка вычислений возрастает с увеличением длины отрезка интегрирования, то для уменьшения этой ошибки поступают следующим образом. Разбив интервал $[a, b]$, на n частей, можно применять формулу (4.13) для каждого из этих участков в отдельности, т.е. рассматривать не один интерполяционный многочлен степени n на всем интервале $[a, b]$, а n интерполяционных многочленов первой степени, различных на каждом из отдельных участков. При этом кривая заменяется ломаной линией (рис. 4.3).

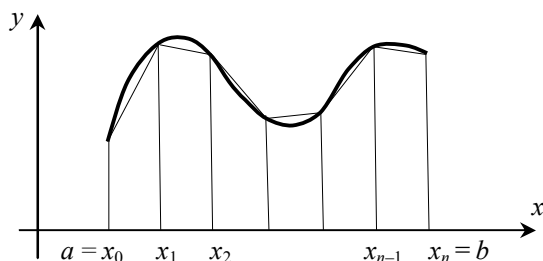


Рис. 4.3

Применяя формулу (4.12) к участкам (x_{i-1}, x_i) , получаем

$$\int_{x_0}^{x_1} y dx = \frac{h}{2} (y_0 + y_1), \int_{x_1}^{x_2} y dx = \frac{h}{2} (y_1 + y_2), \dots, \int_{x_{n-1}}^{x_n} y dx = \frac{h}{2} (y_{n-1} + y_n). \quad (4.14)$$

Сложив все формулы (4.14), придем к общей формуле, дающей приближенное выражение для интеграла:

$$\int_a^b f(x) dx \approx h \left(\frac{y_0 + y_n}{2} + y_1 + y_2 + \dots + y_{n-1} \right). \quad (4.15)$$

Эта формула дает при достаточно малых h , т.е. при большом числе n отрезков деления, довольно хорошие резуль-

таты. Формула (4.15) носит название составной формулы трапеций, что вполне объясняется ее геометрическим смыслом.

Ошибка вычислений по формуле (4.15) складывается из ошибок вычислений на каждом из отрезков $[x_{i-1}, x_i]$. Поэтому

$$|R| \leq \frac{(b-a)h^2}{12} \max_{x \in [a,b]} |f''(x)| = \frac{(b-a)^3}{12n^2} \max_{x \in [a,b]} |f''(x)|. \quad (4.16)$$

4.4. Параболическая интерполяция. Формула Симпсона

Для получения еще одной квадратурной формулы численного интегрирования подставим в интеграле вместо $f(x)$ интерполяционный полином второго порядка, построенный по трем равноотстоящим узлам. Тогда

$$\begin{aligned} I &\approx \int_a^b \left(\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} y_0 + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} y_1 + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} y_2 \right) dx = \\ &= \frac{y_0}{2h^2} \int_{x_0}^{x_2} (x-x_1)(x-x_2) dx - \frac{y_1}{h^2} \int_{x_0}^{x_2} (x-x_0)(x-x_2) dx + \\ &\quad + \frac{y_2}{2h^2} \int_{x_0}^{x_2} (x-x_0)(x-x_1) dx = \frac{h}{3} (y_0 + 4y_1 + y_2). \end{aligned}$$

Здесь положено $a=x_0$, $b=x_2$, $h=x_1-x_0=x_2-x_1$, $y_0=f(x_0)$, $y_1=f(x_1)$, $y_2=f(x_2)$. Тем самым получена формула

$$\int_a^b f(x) dx \approx \frac{(b-a)}{6} \left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right], \quad (4.17)$$

называемая квадратурной формулой Симпсона или формулой парабол. Она совпадает с формулой Ньютона-Котеса при $n=2$. Ее геометрический смысл заключается в том, что в интервале интегрирования $[x_0, x_0+2h]$ функция $y=f(x)$ заменяется обычной параболой второй степени $y=Ax^2+Bx+C$, проходящей через три точки кривой с абсциссами x_0, x_0+h, x_0+2h (рис. 4.4, тонкая линия). При этом площадь криволинейной трапеции заменяется площадью параболической трапеции.

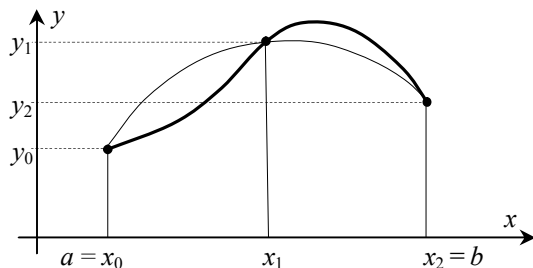


Рис. 4.4

Для повышения точности интегрирования разобьем интервал $[a, b]$ на четное число $n = 2m$ равных отрезков и для каждого из отрезков $[x_0, x_2]$, $[x_2, x_4]$, ..., $[x_{2n-2}, x_{2n}]$ применим формулу (4.17)

$$\begin{aligned} \int_{x_0}^{x_2} y dx &= \frac{h}{3}(y_0 + 4y_1 + y_2), \\ \int_{x_2}^{x_4} y dx &= \frac{h}{3}(y_2 + 4y_3 + y_4), \\ &\dots\dots\dots \\ \int_{x_{2n-2}}^{x_{2n}} y dx &= \frac{h}{3}(y_{2n-2} + 4y_{2n-1} + y_{2n}). \end{aligned} \quad (4.18)$$

Просуммировав все формулы (4.18), получим так называемую составную формулу Симпсона:

$$\int_a^b f(x) dx = \frac{h}{3} [y_0 + y_{2n} + 4(y_1 + y_3 + \dots + y_{2n-1}) + 2(y_2 + y_4 + \dots + y_{2n-2})]. \quad (4.19)$$

Если подынтегральная функция $f(x)$ имеет непрерывную производную четвертого порядка на $[a, b]$, то справедлива такая оценка погрешности формулы Симпсона

$$|R_h| \leq \frac{(b-a)h^4}{180} \max_{x \in [a, b]} |f^{IV}(x)|,$$

или

$$|R_n| \leq \frac{(b-a)^5}{180n^4} \max_{x \in [a,b]} |f^{IV}(x)|. \quad (4.20)$$

Пример. При $n=10$ вычислить интеграл $\int_0^1 \frac{1}{1+x^2} dx$ по формулам трапеций и Симпсона.

Решение. Вычисления можно организовать следующим образом. Составим таблицу значений $f(x) = (1+x^2)^{-1}$.

Таблица 4.1

k	$x_k = a + kh$	$f(x_k), k = 1, \dots, 9$	$f(x_k), k = 0, 10$
0	0,0		1,000
1	0,1	0,990	
2	0,2	0,962	
3	0,3	0,917	
4	0,4	0,862	
5	0,5	0,800	
6	0,6	0,735	
7	0,7	0,671	
8	0,8	0,610	
9	0,9	0,552	
10	1,0		0,500
Суммы		7,099	1,500

По формуле трапеций при $n=10$ имеем

$$I = \frac{1}{10} (7,099 + \frac{1}{2} \cdot 1,500) = 0,7849.$$

Оценим погрешность из неравенства (4.16). В специальной литературе показано, что $\left(\frac{1}{1+x^2}\right)^{(n)} \leq n!$, откуда следует, что

$$\left| \left(\frac{1}{1+x^2} \right)^{(2)} \right| \leq 2, \quad x \in [0, 1].$$

(Разумеется, этот результат можно получить обычным способом, взяв производную 2-го порядка от функции $f(x) = (1+x^2)^{-1}$, и определив ее наибольшее значение на заданном промежутке).

$$|R_n| \leq \frac{1}{12 \cdot 10^2} \cdot 2 < 0,2 \cdot 10^{-2}.$$

Точное значение $I = \frac{\pi}{4} = 0,78539816$. Из сравнения точного и приближенного значений видно, что две первые цифры совпадают.

Для применения формулы Симпсона также составим таблицу. При этом и число узлов n и шаг h остаются прежними, и вычислительные затраты примерно такие же, но значение интеграла получается заметно точнее, чем с помощью формулы трапеций.

Таблица 4.2

k	$x_k = a + kh$	$f(x_k), k=1(1)10$	$f(x_k), k=1(2)9$	$f(x_k), k=2(2)8$
0	0,0	1,00000		
1	0,1		0,99010	
2	0,2			0,96153
3	0,3		0,91743	
4	0,4			0,86206
5	0,5		0,80000	
6	0,6			0,73529
7	0,7		0,67114	
8	0,8			0,60975
9	0,9		0,55249	
10	1,0	0,50000		
Суммы		1,50000	3,93116	3,16865

Действительно, по формуле Симпсона находим

$$\int_0^1 \frac{1}{1+x^2} dx \approx \frac{1}{3 \cdot 10} (1,50000 + 4 \cdot 3,93116 + 2 \cdot 3,16865) = 0,78539.$$

С учетом того, что

$$\left| \left(\frac{1}{1+x^2} \right)^{(4)} \right| \leq 24, \quad x \in [0, 1],$$

получим оценку погрешности формулы:

$$|R_n| \leq \frac{1}{180 \cdot 10^4} \cdot 24 < 0,2 \cdot 10^{-4}.$$

Т.е. результат содержит как минимум четыре верных цифры после запятой. На самом деле вычисленное значение оказалось точнее, чем дает оценка (4.16), и это обстоятельство встречается нередко на практике, что позволяет говорить о грубости подобных оценок. Тем не менее, с помощью неравенств (4.16) можно осуществлять приближенное вычисление интегралов с наперед заданной точностью.

Пример. Вычислить с двумя верными цифрами после запятой значение интеграла $\int_0^4 \frac{1}{2+x} dx$.

Решение. Естественно ожидать, что формула Симпсона дает необходимую точность при меньших затратах вычислительного труда по сравнению с методом трапеций. В самом деле,

$$f(x) = \frac{1}{2+x}, \quad f''(x) = \frac{2}{(2+x)^3}, \quad \max_{0 \leq x \leq 4} |f''(x)| = f''(0) = \frac{1}{4},$$

$$f^{IV}(x) = \frac{24}{(2+x)^5}, \quad \max_{0 \leq x \leq 4} |f^{IV}(x)| = f^{IV}(0) = \frac{3}{4}.$$

Перепишем неравенства (4.16) и (4.20) для нашего случая. Для формулы трапеций

$$|R_n| \leq \frac{4^3}{12n^2} \cdot \frac{1}{4} = \frac{4}{3n^2}.$$

Для формулы Симпсона

$$|R_n| \leq \frac{4^4}{180n^4} \cdot \frac{3}{4} = \frac{4}{3n^2} \cdot \frac{4^2}{5n^2}.$$

Из этих неравенств видно, что если для получения необходимой точности взять $n \geq 2$, то формула Симпсона даст значение интеграла с большей точностью, чем формула трапеций.

Если пользоваться формулой трапеций, то для достижения заданной точности достаточно потребовать $\frac{4}{3n^2} \leq \frac{1}{2} \cdot 10^{-2}$, поэтому достаточно взять $n=17$. Для формулы Симпсона $\frac{64}{15n^4} \leq \frac{1}{2} \cdot 10^{-2}$, и n можно взять равным 6.

Применяя формулу Симпсона при $n=6$, имеем

$$\int_0^4 \frac{1}{2+x} dx \approx \frac{4}{3 \cdot 6} \left(\frac{1}{2} + 4 \cdot \frac{3}{8} + 2 \cdot \frac{3}{10} + 4 \cdot \frac{1}{4} + 2 \cdot \frac{3}{14} + 4 \cdot \frac{3}{16} + \frac{1}{6} \right) = 1,099.$$

4.5. Использование сплайнов

Одним из методов численного интегрирования, особенно эффективным при строго ограниченном числе узлов, является метод, использующий интерполяцию сплайнами.

Разобьем отрезок интегрирования $[a, b]$ на n частей точками x_i . Пусть $x_i - x_{i-1} = h_i$, $i=1, 2, \dots, n$. На каждом элементарном отрезке интерполируем подынтегральную функцию $f(x)$ с помощью какого-либо сплайна, например кубического:

$$p_i(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3, \\ x_{i-1} \leq x \leq x_i, \quad i = 1, \dots, n.$$

Выражение для интеграла представим в виде

$$I = \int_a^b f(x) dx = \sum_{i=1}^n \int_{x_{i-1}}^{x_i} f(x) dx \approx \sum_{i=1}^n \int_{x_{i-1}}^{x_i} p_i(x) dx.$$

После подстановки выражения для $p_i(x)$ в результате вычисления интегралов получим

$$I \approx \sum_{i=1}^n \left(a_i h_i + \frac{1}{2} b_i h_i^2 + \frac{1}{3} c_i h_i^3 + \frac{1}{4} d_i h_i^4 \right).$$

Эту формулу можно представить в удобном для практических расчетов виде

$$I \approx \frac{1}{2} \sum_{i=1}^n h_i (y_{i-1} + y_i) - \frac{1}{12} \sum_{i=1}^n h_i^3 (c_i + c_{i+1}).$$

Отметим, что при использовании сплайнов представление в виде квадратурной формулы (4.2) невозможно, поскольку коэффициенты A_k в этом случае зависят от всех значений y_i .

4.6. Квадратурные формулы Гаусса

В квадратурных формулах, полученных на основе метода Ньютона-Котеса, используются равноотстоящие узлы. В квадратурных формулах Гаусса распределение узлов на интервале интегрирования не является равномерным. Идея формул Гаусса состоит в том, чтобы при возможно наименьшем числе узлов точно интегрировать многочлены наивысшей возможной степени. Это как раз и достигается за счет специального выбора положения узлов x_k на промежутке интегрирования.

Рассмотрим функцию $y=f(x)$, заданную на стандартном интервале $[-1, 1]$. Любой интервал $[a, b]$ путем замены переменных может быть сведен к стандартному. Поставим задачу так: как распределить точки $x_1, x_2, \dots, x_n$ на интервале $[-1, 1]$ и подобрать коэффициенты $A_1, A_2, \dots, A_n$, чтобы квадратурная формула

$$\int_{-1}^1 f(x) dx \approx \sum_{k=1}^n A_k f(x_k) \quad (4.21)$$

была точной для всех полиномов $f(x)$ максимально возможной степени N .

Для решения этой задачи необходимо определить $2n$ постоянных $\{x_k\}$, $\{A_k\}$. С помощью этого количества коэффициентов можно определить полином степени $2n-1$. Очевидно, что это и будет наивысшая возможная степень полинома N . Можно показать, что для выполнения равенства (4.21) необходимо и достаточно, чтобы оно было верным при $f(x)=1$, $f(x)=x$, $f(x)=x^2$, $\dots$, $f(x)=x^{2n-1}$.

Подставляя в (8.3) вместо $f(x)$ последовательно указанные функции x^k , $k=0, 1, \dots, 2n-1$, получим систему $2n$ нелинейных уравнений

$$\sum_{k=1}^n A_k x_k^{2n-1} = A_1 x_1^{2n-1} + A_2 x_2^{2n-1} + \dots + A_n x_n^{2n-1} = 0.$$

$$P_n(x) = \frac{1}{2^n n!} \frac{d^n (x^2 - 1)^n}{dx^n}.$$
$$A_k = \frac{2}{(1-x_k^2)[P'_v(x_k)]^2}, \quad k=1, 2, \dots, n.$$

Таблица 4.3

$n=1$	$x_1=0$	$A_1=2$
$n=2$	$x_2=-x_1=0,5773502692$	$A_1=A_2=1$
$n=3$	$x_3=-x_1=0,7745966692$ $x_2=0$	$A_1=A_3=1/9$ $A_2=8/9$
$n=4$	$x_4=-x_1=0,8611363116$ $x_3=-x_2=0,3399810436$	$A_1=A_4=0,3478548452$ $A_2=A_3=0,6521451548$

$n=5$	$x_5=-x_1=0,9061798458$ $x_4=-x_2=0,5384693101$ $x_3=0$	$A_5=A_1=0,2369268851$ $A_4=A_2=0,4786286705$ $A_3=0,5688888889$
$n=6$	$x_6=-x_1=0,9324695142$ $x_5=-x_2=0,6612093865$ $x_4=-x_3=0,2386191861$	$A_6=A_1=0,1713244924$ $A_5=A_2=0,3607615730$ $A_4=A_3=0,4679139346$

Остаточный член формулы Гаусса (4.21) с n узлами определяется таким образом

$$R_n = \frac{2^{2n+1}(n!)^4}{(2n+1)[(2n)!]^3} f^{(2n)}(\xi), \quad \xi \in [-1, 1].$$

При вычислении интеграла $\int_a^b f(t)dt$ следует сделать замену переменной интегрирования

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

Тогда

$$\int_a^b f(t)dt = \frac{b-a}{2} \int_{-1}^1 f\left(\frac{b-a}{2}x + \frac{a+b}{2}\right)dx.$$

Применяя к правой части формулу (4.21), находим

$$\int_a^b f(t)dt = \frac{b-a}{2} \sum_{k=1}^n A_k f(t_k) + \tilde{R}_n, \quad (4.22)$$

где $t_k = \frac{b-a}{2}x_k + \frac{a+b}{2}$; x_k – узлы квадратурной формулы Гаусса для отрезка $[-1, 1]$ и A_k – соответствующие им коэффициенты. Остаток квадратуры (4.22) составляет

$$\tilde{R}_n = \frac{(b-a)^{2n+1}(n!)^4}{(2n+1)[(2n)!]^3} f^{(2n)}(\xi), \quad \xi \in [a, b].$$

Применение формулы Гаусса позволяет получить точный результат при малом числе узлов интегрирования. В тех случаях, когда для вычисления подынтегральной функции в каждом узле требуется много времени, применение формулы Га-

усса особенно выгодно. Недостатком формулы Гаусса является то, что и абсциссы узлов интегрирования x_k , и коэффициенты A_k являются в общем случае иррациональными числами.

Если один и тот же интеграл необходимо вычислять с разным количеством узлов, то использование метода Гаусса, несмотря на его большую точность нецелесообразно, так как нельзя использовать ранее вычисленные ординаты, ибо они находятся в неподходящих узлах. В подобной ситуации лучше использовать формулу Симпсона.

При обработке экспериментальных данных метод Гаусса целесообразно использовать лишь в том случае, если абсциссы экспериментальных данных согласуются с абсциссами, выбранными в методе Гаусса. Однако такое согласование встречается достаточно редко. Если же узлы интегрирования расположены случайным образом, то для интегрирования можно применить лишь формулу трапеций. Формула Симпсона обеспечивает достаточную точность при умеренном числе ординат.

4.7. Двойной пересчет

Так как отыскание производных высокого порядка для формирования остаточного члена часто приводит к громоздким вычислениям и является неформализуемым действием, то на практике часто используют следующий прием: интеграл вычисляют сначала с начальным шагом h (что соответствует числу n частичных отрезков), а затем с шагом $h/2$ ($2n$ отрезков). Если $|I_{2n} - I_n| \geq \varepsilon$, то расчет повторяют с шагом $h/4$. В противном случае вычисления останавливают, и за приближенное значение интеграла берут любое из найденных значений. Указанный приём широко используется при компьютерных вычислениях, так как он позволяет осуществлять автоматический выбор шага при заданной точности с одновременным контролем.

Для приближенной оценки погрешности усечения можно использовать также принцип Рунге, согласно которому

$$\Delta \approx \frac{1}{3} |I_{2n} - I_n| \quad \text{для формулы трапеций,}$$

$$\Delta \approx \frac{1}{15} |I_{2n} - I_n| \quad \text{для формулы Симпсона.}$$

Однако в случае, если подынтегральная функция имеет много нулей и экстремумов на отрезке интегрирования, то точность квадратурных формул значительно снижается за счет больших значений производных высокого порядка. В этом случае шаг h следует выбирать намного меньше расстояния между соседними нулями функции и ее производной.

4.8. Метод статистических испытаний Монте–Карло

Идею метода статистических испытаний сначала проиллюстрируем на примере приближенного вычисления определенного интеграла от функции $\varphi(x)$.

Пусть X – равномерно распределенная на отрезке $[a, b]$ случайная величина. Это означает, что её плотность распределения задается соотношением

$$f_X(x) = \begin{cases} \frac{1}{b-a}, & \text{если } a \leq x \leq b, \\ 0, & \text{в противном случае.} \end{cases} \quad (4.23)$$

Тогда любая функция $Y = \varphi(X)$ также будет случайной величиной, и ее математическое ожидание равно

$$M[Y] = \int_{-\infty}^{+\infty} \varphi(x) f_X(x) dx = \int_a^b \varphi(x) \frac{1}{b-a} dx = \frac{1}{b-a} \int_a^b \varphi(x) dx.$$

Следовательно, читая это равенство в обратном порядке, приходим к выводу, что интеграл $I = \int_a^b \varphi(x) dx$ может быть вычислен как оценка математического ожидания (с точностью до множителя $b-a$) некоторой случайной величины Y , которая

является функцией случайной величины X с равномерным законом распределения.

Пусть X_k ($k = 1, 2, \dots$) – независимые случайные величины, распределенные по одному и тому же закону (4.23), тогда случайную величину

$$I_n = (b-a) \frac{1}{n} \sum_{k=1}^n \varphi(X_k)$$

можно рассматривать как приближенное значение интеграла I . Из закона больших чисел следует (при условии конечности дисперсии $D[\varphi(X_k)]$), что I_n стремится по вероятности к I при $n \rightarrow \infty$, т.е. вероятности больших отклонений I_n от $M[I_n] = I$ малы при больших n . Итак,

$$\int_a^b \varphi(x) dx \approx \frac{b-a}{n} \sum_{k=1}^n \varphi(X_k).$$

Аналогичный подход применяется к вычислению кратных интегралов. Для тройного интеграла получим

$$\iiint_{\Omega} \varphi(x, y, z) dx dy dz \approx \frac{V}{n} \sum_{k=1}^n \varphi(X_k, Y_k, Z_k).$$

где V – объем тела Ω . Здесь под $\{X_k, Y_k, Z_k\}$ понимается случайный вектор, распределенный равномерно в трехмерной области Ω . Его плотность распределения равна

$$f(x, y, z) = \begin{cases} \frac{1}{V}, & \text{если } (x, y, z) \in \Omega, \\ 0, & \text{в противном случае.} \end{cases}$$

При программировании метода Монте-Карло значения случайной величины, распределенной равномерно в интервале $[0, 1]$, обычно генерируются с помощью процедуры **random**. В случае тройного интеграла следует учесть, что три независимых равномерных случайных величин образуют случайный вектор, распределенный равномерно в кубе $0 \leq x \leq 1$, $0 \leq y \leq 1$, $0 \leq z \leq 1$. Троекратное обращение к функции **random** дает реализацию (x_k, y_k, z_k) этого вектора. Пусть, например, областью

интегрирования V является тетраэдр, занимающий $1/6$ часть от данного куба. Тогда, если учитывать только точки (x_k, y_k, z_k) , попадающие в этот тетраэдр, то получим случайный вектор, равномерно распределенный в тетраэдре $x \geq 0, y \geq 0, z \geq 0, x + y + z \leq 1$ (что и требуется в методе Монте-Карло). Объем этого тетраэдра равен $V=1/6$. Вычисления интеграла проводятся по формуле

$$\iiint_{\Omega} \frac{1}{(1+x+y+z)^3} dx dy dz \approx \frac{(1/6)}{n} \sum_{k=1}^n \frac{1}{(1+x_k+y_k+z_k)^3}. \quad (4.24)$$

В качестве условия окончания счета можно принять следующий подход. Пусть m – число реализаций случайного вектора, попадающих в куб (из них затем отбираются попадающие в тетраэдр). Вычисляются значения I при $m=1000, 2000, 4000, \dots$. Если $|I_{2m} - I_m| < \varepsilon$, то выполнение программы прекращается, а в качестве I берется значение, вычисленное по формуле (4.24), с полным числом n всех точек, прошедших отбор.

4.9. Приближенное вычисление интегралов в системе Maple

Пример 1. Проинтегрировать функцию, заданную таблично.

x_k	0	0,227	0,591	0,915	1,264	1,612	1,800
y_k	0,8632	0,9528	0,9733	0,7148	0,4189	0,3384	0,4481

Решение. Заметим, что функция задана не в равноотстоящих точках. Формулы Ньютона-Котеса (и, в частности, формула Симпсона) здесь неприменимы. Поэтому необходимо использовать общий подход, состоящий в аппроксимации функции на основе либо глобальной, либо кусочной интерполяции, а затем аналитическом интегрировании этой аппроксимации.

1 способ. Вычисление с помощью глобального интерполяционного многочлена

```
> restart;
> with (plots) :
> x:=[0,0.227,0.591,0.915,1.264,1.612,1.800] :
> y:=[0.8632,0.9528,0.9733,0.7148,0.4189,
```

```
0.3384,0.4481];
```

Имеем 7 узлов $\{x_k\}$, которые расположены в порядке возрастания (это обязательное условие); отрезок интегрирования – от первой точки до последней.

```
> n:=7: a:=X[1]: b:=X[n]:
```

Ниже получаем интерполяцию сразу по всем узловым точкам и для контроля выводим графические построения

```
> f:=interp(X,Y,t);
```

$$f := 0.5338789920 t^6 - 3.454406447 t^5 + 8.666756147 t^4 - 9.635910292 t^3 + 3.700870772 t^2 - 0.0413804724 t + 0.8632$$

```
> fu:=unapply(f,t):# функция-интерполант
```

```
> g1:=plot(fu(x),x=a..b):
```

```
> g2:=plot([X[k],Y[k]]$k=1..n, x=a..b,  
style=POINT,symbol=CIRCLE):
```

```
> display(g1,g2);
```

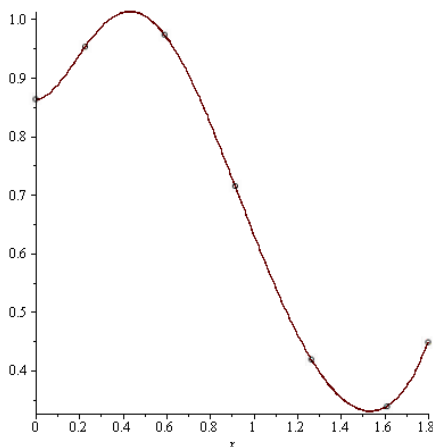


Рис. 4.5

Теперь осталось вычислить интеграл от интерполирующей функции, что не составляет труда, поскольку эта функция является полиномом, в данном случае, 6-й степени.

```
> int(fu(x),x=a..b);
```

```
1.232881868
```


2 способ. Вычисление с помощью локальной интерполяции

Глобальная интерполяция с увеличением числа узлов n может вносить существенную погрешность, поэтому на практике чаще используют локальную (кусочную) интерполяцию, при которой строят интерполирующие полиномы невысоких степеней на соответствующих частичных отрезках области определения функции. Одним из наиболее успешных видов локальной интерполяции является аппроксимация сплайнами.

Применим интерполяцию кубическим сплайном. Заметим, что при этом аппроксимирующая функция получается кусочно заданной (само её выражение здесь не приводится).

```
> readlib(spline);  
      proc(XX, YY, z, d) ... end proc  
> f1:=spline(X,Y,x,cubic); # функция – интерполант  
> g3:=plot(f1,x=a..b,color=black):  
> g4:=plot([ [X[k],Y[k]]$k=1..n],x=a..b,  
style=POINT,symbol=CIRCLE,color=black):  
> display(g3,g4);
```

Вычисляем интеграл:

```
> int(f1,x=a..b);  
1.232975570
```

Найдем приближенное значение интеграла также с помощью составной формулы трапеции в варианте с нефиксированным шагом $h_i = x_{i+1} - x_i$:

$$\int_a^b y dx \approx \frac{1}{2} \sum_{i=1}^{n-1} (y_{i+1} + y_i) h_i.$$

```
> sum((Y[j]+Y[j+1])*(X[j+1]-X[j])/2,j=1..n-1);  
1.233635400
```

Такое вычисление соответствует случаю кусочно-линейной интерполяции, которую, как легко в этом убедиться, обеспечивает сплайн 1-го порядка. Для этого в Maple-функции **spline** параметр **cubic** следует поменять на **linear** и перезапустить весь приведенный фрагмент.

Для более полной картины вычислим также интеграл с помощью сплайна 2-го порядка:

```
> f1:=spline(X,Y,x,quadratic);
[...]
```

1.234524598

Итак, обобщая полученные результаты, заключаем, что искомый интеграл примерно равен $I \approx 1.23$.

Пример 2. Проинтегрировать функцию, заданную в виде таблицы с равномерным шагом.

x_k	0,0	0,25	0,5	0,75	1,0	1,25	1,50	1,75	2,0
y_k	1,849	1,960	2,028	2,048	2,020	1,944	1,825	1,670	1,489

Решение.

```
> restart;
> n:=9: X:=vector(n);Y:=vector(n);
> X:=[0,0.25,0.5,0.75,1.,1.25,1.5,1.75,2.];
> Y:=[1.849, 1.960, 2.028, 2.048, 2.020, 1.944,
1.825, 1.670, 1.489];
> a:=X[1]: b:=X[n]: h:=(b-a)/(n-1);
```

Найдём значение интеграла по формуле трапеций.

```
> sum((Y[k]+Y[k+1])*h/2,k=1..n-1);
```

3.791000000

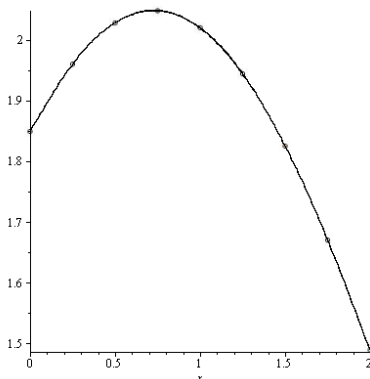
Затем – по формуле Симпсона:

```
> s:=0: for j from 1 to n-2 by 2 do
      s:=s+(Y[j]+4*Y[j+1]+Y[j+2])*h/3 od:
> s;
```

3.797666667

Вычислим интеграл также на основе сплайн-интерполяции.

```
> readlib(spline);
f1:=spline(X,Y,x,3);
> g4:=plot(f1,x=a..b,
color=black):
```



```

g5:=plot([[X[k],Y[k]]$k=1..n],
x=a..b,style=POINT,symbol=
CIRCLE, color=black):
> plots[display](g4,g5);
> int(f1,x=a..b);
3.797371134

```

Три совпадающие значащие цифры можно принять за искомое значение: $I \approx 3,79$.

Пример 3. Вычислить число $\int_0^1 \cos x^2 dx$ Рис. 4.6

Решение.

Расчёт выполним несколькими различными способами. Число частичных отрезков n везде возьмем одно и то же – равное 10.

```

> restart;
> f:=t->1/cos(t^2); a:=0:b:=1: # эти определения общие
# для всех способов решения
# 1-й способ – с помощью встроенной функции Maple
> evalf(int(f(x),x=a..b));
1.132697500

# 2-й способ – методом прямоугольников
> n:=10:h:=(b-a)/n:
> S1:=0: S2:=0: S3:=0:
for i from 1 to n do
    S1:=evalf(S1+f(a+(i-1)*h));
    S2:=evalf(S2+f(a+i*h));
    S3:=evalf(S3+f(a+(i-1/2)*h));
od:
> Irecleft:=S1*h;Irecright:=S2*h;Irecmiddle:=S3*h;
Irecleft:=1.094885378
Irecright:=1.179966950
Irecmiddle:=1.130360995

```

(Результаты, полученные по методам левых, правых и средних прямоугольников соответственно).

3-й способ – методом трапеций

```

> n:=10:h:=(b-a)/n:
> S:=0: for i from 1 to n do
    S:=evalf(S+f(a+(i-1)*h)+f(a+i*h));
od:
> Itrap:=S*h/2;

```

Itrap := 1.137426163

4-й способ – методом Симпсона

```

> n:=10: h:=(b-a)/n:
> S:=0: for i from 1 to n by 2 do
S:=evalf(S+f(a+(i-1)*h)+4*f(a+i*h)+f(a+(i+1)*h));
od:
> Isimp:=S*h/3;

```

Isimp := 1.132943275

Проверим формулу $I \approx \frac{2I_{\text{пр}} + I_{\text{тр}}}{3}$.

Выполнив расчет по формуле Симпсона для вдвое большего числа отрезков деления ($n=20$), получим

```

> n:=20: h:=(b-a)/n:
> S:=0: for i from 1 to n by 2 do
S:=evalf(S+f(a+(i-1)*h)+4*f(a+i*h)+f(a+(i+1)*h));
od:
> Isimp20:=S*h/3;

```

Isimp20 := 1.132716051

С другой стороны,

```

> (2*Irecmiddle+Itrap)/3;

```

1.132716051

Результаты совпадают. Таким образом, по двум значениям интеграла, полученным с помощью формул трапеций и средних прямоугольников для шага h , можно найти более точное значение, соответствующее формуле Симпсона для половинного шага $h/2$, т.е. с удвоенным числом n .

Проверим также правило Рунге, согласно которому, зная два вычисленных численно значения интеграла для n и $2n$ частичных отрезков, можно оценить погрешность, а также уточ-

нить результат на один порядок формулы численного интегрирования. Для метода Симпсона имеет место формула

$$I = I_{2n} + \frac{I_{2n} - I_n}{15} + O(h^5).$$

Имеем

```
> Isimp20+(Isimp20-Isimp)/15;
1.132700902
```

Если отталкиваться от результата, рассчитанного с помощью встроенной функции Maple, то это число ближе к правильному, чем значение по формуле Симпсона при $n=20$.

И в заключение проведем уточнение на основе процесса Эйткена, согласно которому

$$I \approx I_n + \frac{(I_{2n} - I_n)^2}{I_n - 2I_{2n} + I_{4n}}.$$

Взяв за n в этой формуле число 10, замечаем, что I_{10} , I_{20} уже вычислены, и осталось найти недостающее значение I_{40} . Для этого запустим программу метода Симпсона для $n=40$:

```
> n:=40: h:=(b-a)/n:
> S:=0:
  for i from 1 to n by 2 do
    S:=evalf(S+f(a+(i-1)*h)+4*f(a+i*h)+f(a+(i+1)*h));
  od:
> Isimp40:=S*h/3;
```

```
Isimp40 := 1.132698732
```

```
> Isimp-(Isimp20-Isimp)^2/(Isimp-
2*Isimp20+Isimp40);
```

```
1.132697303
```

Видно, что получилось число более точное, чем I_{40} .

Таким образом, все рассмотренные здесь процедуры уточнения решения действительно работают.

Пример 4. Вычислить интеграл $\int_0^1 e^{-x^2} dx$ по формулам

Симпсона и трапеций с четырьмя верными знаками после запятой.

Решение.

```
> restart;
```

Зададим пределы интегрирования и подынтегральную функцию.

```
> a:=0: b:=1:
```

```
> f:=t->exp(-t^2);
```

Вычислим четвертую производную

```
> df4:=diff(f(x),x$4);
```

$$df4 := 12 e^{-x^2} - 48 x^2 e^{-x^2} + 16 x^4 e^{-x^2}$$

Определим её наибольшее по модулю значение на отрезке $[0, 1]$.

```
> maximize(abs(df4),x,{x=a..b});
```

12

Замечание. В Maple 5.4 функция **maximize** в сочетании с модулем **abs** может дать неверный результат (чтобы это проверить, постройте график **abs(df4)**), и для нахождения требуемого максимума рекомендуется применить надежный, но более громоздкий способ:

```
> mxd4:=max(evalf(abs(minimize(df4,x,{x=a..b}))),  
evalf(abs(maximize(df4,x,{x=a..b}))));
```

12.

Таким образом, $\max_{0 \leq x \leq 1} |f(x)| = 12$. Поэтому

$$|R_n| \leq \frac{1}{180n^4} \cdot 12,$$

и чтобы было четыре верных цифры после запятой, необходимо потребовать

$$\frac{1}{180n^4} \cdot 12 \leq \frac{1}{2} \cdot 10^{-4}.$$

Решим это неравенство относительно n .

```
> solve((b-a)^5*mxd4/180/m^4<0.5e-4,m);
```

RealRange(-∞, Open(-6.042750795)), RealRange(Open(6.042750795), ∞)

Учитывая, что n в формуле Симпсона может быть только четным числом, получаем, что достаточно взять $n=8$, при этом $|R_8| < 0,2 \cdot 10^{-4}$.

```
> n:=8: h:=(b-a)/n;
```

Далее – цикл, суммирующий значения по формуле Симпсона.

```
> S:=0: for i from 1 to n by 2 do
S:=evalf(S+f(a+(i-1)*h)+4*f(a+i*h)+f(a+(i+1)*h));
od:
```

Вывод окончательного результата:

```
> Isimp:=S*h/3;
```

```
Isimp := 0.7468261197
```

Вычислим тот же самый интеграл с помощью встроенной функции Maple. Поскольку у него первообразная в конечном виде не выражается, результат будет получен приближенно и тоже численным интегрированием.

```
> evalf(int(f(x), x=a..b));
0.7468241330
```

Совершенно аналогично можно выполнить шаги по реализации метода трапеций:

```
> restart;
> f:=t->exp(-t^2); a:=0:b:=1:eps:=1e-4:
> df2:=diff(f(x), x$2);
```

$$df2 := -2 e^{-x^2} + 4 x^2 e^{-x^2}$$

```
> mxd2:=maximize(abs(df2), x=a..b);
mxd2 := 2
```

(См. замечание выше относительно функции **maximize**).

```
> solve((b-a)^3*mxd2/12/m^2<0.5*eps, m);
RealRange(-∞, Open(-57.73502692)), RealRange(Open(57.73502692), ∞)
```

Как видно, число узлов здесь по сравнению с методом Симпсона достаточно велико; n надо брать не меньше 58.

```
> n:=58:h:=(b-a)/n:
> S:=0: for i from 1 to n do
```

```

S:=evalf(S+f(a+(i-1)*h)+f(a+i*h));
od:
> Itrap:=S*h/2;

Itrap:=0.7468059065

```

Таким образом, $\int_0^1 e^{-x^2} dx \approx 0,7468$.

Пример 5. Применяя процедуру двойного пересчета к методам численного интегрирования, вычислить значение постоянной Каталана с $\varepsilon = 10^{-4}$:

$$\int_0^1 \frac{\arctg x}{x} dx$$

Решение.

```
> restart;
```

Следует учесть, что если подынтегральную функцию задать стандартным способом в виде

```
> f:=x->arctan(x)/x;
```

то в дальнейшем Maple будет сообщать об ошибке (division by zero), так как в точке $x=0$ она не определена, а в вычислениях по формулам прямоугольников, трапеций и Симпсона эта точка участвует. Но поскольку предел ($\arctg x/x$ при $x \rightarrow 0$ равен 1, то надо соответствующим образом переопределить функцию в данной точке, и тогда особенность исчезнет. Для этого используем кусочную функцию:

```
> f:=x->piecewise(x<>0,arctan(x)/x,1);
```

$$f:=x \rightarrow \text{piecewise}\left(x \neq 0, \frac{\arctan(x)}{x}, 1\right)$$

(т.е. если $x \neq 0$, то функция равна $\arctg x/x$, в противном случае равна 1). При этом следует отметить, что такой способ задания функции может не сработать в раннем выпуске пакета Maple, например версии V release 4. В этом случае кусочную функцию можно ввести с помощью подпрограммы (процедуры):

```
> f:=proc(x) local g; if x=0 then g:=1;else
g:=evalf(arctan(x)/x); fi;RETURN(g) end;
```

```
# инициализация переменных
```



```

> n:=1: eps:=1e-4: a:=0: b:=1: h:=(b-a)/n:
Ir:=1: Ir2:=0: It:=1: It2:=0:
# основной цикл методов прямоугольников и трапеций
> while (n<4) or (evalf(abs(Ir2-Ir))>eps) or
(evalf(abs(It2-It)))>eps do
if n>=2 then Ir:=Ir2; It:=It2; fi;
> S1:=0; S2:=0;n:=2*n;h:=h/2;
for i from 1 to n do S1:=evalf(S1+f(a+(i-1)*h+h/2));
S2:=evalf(S2+f(a+(i-1)*h)+f(a+i*h)); od;
Ir2:=S1*h;
It2:=S2*h/2;
> od:
> It2; # значение интеграла по методу прямоугольников
0.9159423680
> Ir2; # значение интеграла по методу трапеций
0.9159772072
> n; # конечное число отрезков деления
32
> evalf(int(f(x),x=a..b));
0.9159655940 - 2. 10-10 I
# ниже – цикл метода Симпсона
> n:=2: J:=0: J2:=1: h:=(b-a)/n:
> while (n<8) or evalf(abs(J2-J))>eps do
if n>2 then J:=J2 fi;
> S3:=0; n:=2*n;h:=h/2;
for i from 2 to n do
if (i mod 2)=0 then
S3:=evalf(S3+4*f(a+(i-1)*h))
else S3:=evalf(S3+2*f(a+(i-1)*h));
fi;
od;
J2:=evalf((f(a)+f(b)+S3)*h/3);
> od:
> J;J2; # значения интеграла по методу Симпсона с числом
# частичных отрезков n и 2n
0.9159710936

```

0.9159659775

> n;

8

уточнение результата согласно правилу Рунге

> Jm:=J2+ ((J2-J) /15) ;

Jm := 0.9159656364

Пример 6. По формуле Гаусса при $n=5$ вычислить интеграл $\int_0^{\pi/2} e^{-x^2} \cos x dx$.

Решение.

Для программирования квадратурной формулы Гаусса необходимо при заданном n определить два массива чисел – узлы и коэффициенты – по n элементов каждый. Эти числа иррациональны, требуют учета большого числа знаков, поэтому вводить их вручную довольно утомительно и чревато ошибками. К счастью, Maple имеет в своем составе полиномы Лежандра, с помощью которых можно без лишних затрат организовать автоматический ввод указанных чисел, причем для произвольного числа n . Рассмотрим подробнее такую возможность.

> restart;

Digits:=15:

> n:=5;

Ln:=orthopoly[P](n,x);# полином Лежандра n -й степени

n:=5

$$L_n := \frac{63}{8} x^5 - \frac{35}{4} x^3 + \frac{15}{8} x$$

> A:=vector(n) :

> z:=[evalf(solve(Ln,x))]: # корни полинома Лежандра

> for i from 1 to n do # коэффициенты формулы Гаусса

z[i]:=Re(z[i]);

A[i]:=2/(1-z[i]^2)/subs(x=z[i],diff(Ln,x))^2;

od:

> z;evalm(A); # вывод искоемых узлов и коэффициентов

```
[0., 0.538469310105681, -0.538469310105681,
 0.906179845938666, -0.906179845938666]
[0.568888888888888, 0.478628670499370, 0.478628670499370,
 0.236926885056184, 0.236926885056184]
```

Программирование самой квадратурной формулы не составляет большого труда, но при этом следует помнить, что в формуле Гаусса интеграл берется по стандартному промежутку $[-1, 1]$, поэтому в случае произвольного отрезка $[a, b]$ требуется соответствующая замена переменной.

```
> f:=x->exp(-x^2)*cos(x); a:=0: b:=Pi/2:
```

$$f:=x \rightarrow e^{-x^2} \cos(x)$$

```
> Igauss:=evalf((b-a)/2*sum(A[k]*
f((b-a)*z[k]/2+(a+b)/2),k=1..n)):
> Int(f(x),x=a..b)=Igauss;
```

$$\int_0^{\frac{1}{2}\pi} e^{-x^2} \cos(x) dx = 0.695686650994082$$

Ниже для контроля интеграл вычисляется встроенной функцией Maple и выводится график функции на отрезке интегрирования (сам график не приводится).

```
> evalf(Int(f(x),x=a..b));
0.695689575245514
> plot(f(x),x=a..b);
```

Оценим погрешность квадратуры, используя неравенство

$$|R_n| \leq \frac{2^{2n+1}(n!)^4}{(2n+1)[(2n)!]^3} \max_{a \leq x \leq b} f^{(2n)}(x).$$

```
> df10:=diff(f(x),x$10); # 10-я производная, т.к. n=5
> mxdf10:=evalf(maximize(df10,x=a..b)); # max f^(10)
mxdf10:=1.09355595816162 10^5
> R:=evalf((b-a)^(2*n+1)*(n!)^4*mxdf10/(2*n+1)/
((2*n)!^3));
```

$R := 0.00000619730483490134$

Т.е. предельная погрешность не превосходит $6,2 \cdot 10^{-6}$.
Построив интервал

```
> [Igauss-R, Igauss+R] ;  
[0.695680453689247, 0.695692848298917]
```

видим, что значение, вычисленное встроенными средствами Maple, которое принимаем как наиболее близкое к точному, попадает в этот интервал.

Пример 7. Методом Монте-Карло вычислить интеграл $\int_{-1}^1 \frac{1}{2+x^6} dx$.

Решение.

```
> restart;  
> Digits:=15:
```

Зададим подынтегральную функцию и пределы интегрирования

```
> f:=x->1/(2+x^6); a:=-1: b:=1:
```

По умолчанию генератор случайных чисел Maple выдает числа в диапазоне от 0 до 10^{12} . Поэтому для получения стандартного промежутка $[0, 1]$ требуется использовать соответствующий масштабный множитель. Этой цели как раз и вводится переменная **w**.

```
> w:=0.1e13;
```

Ниже вызывается функция, которая инициализирует генератор случайных чисел в соответствии с системными часами.

```
> Seed:=readlib(randomize)();
```

Вычисления организуем блоками по 125000 случайных чисел, число блоков возьмем 10. Для каждого блока выведем приближенное значение в соответствии с алгоритмом Монте-Карло. Параллельно будем вести подсчет общей суммы для всех генерируемых чисел.

```
> kmax:=10: m:=125000:  
> g1:=0: s1:=0: n:=0:  
> for k from 1 to kmax do
```

```

s:=0;
for i from 1 to m do
    x:=(b-a)*rand()/w+a;
    z:=evalf(f(x));
    s:=s+z;
od;
g1:=(b-a)*s/m;
s1:=s1+s; n:=n+m;
printf(`%g  %g  n=%g  I=%g\n`,m,g1,n,(b-a)*s1/n);
od:
125000 .943289    n=125000    I=.943289
125000 .942868    n=250000    I=.943078
125000 .943175    n=375000    I=.943111
125000 .943109    n=500000    I=.943110
125000 .942754    n=625000    I=.943039
125000 .943594    n=750000    I=.943132
125000 .942688    n=875000    I=.943068
125000 .942860    n=1e+06     I=.943042
125000 .943037    n=1.125e+06 I=.943042
125000 .942873    n=1.25e+06  I=.943025
> Imk:=(b-a)*s1/n;
      Imk := .943025151962280

```

Для сравнения, как обычно, вычислим интеграл с помощью стандартной функции Maple.

```
> Int(f(t), t=a..b)=evalf(int(f(t), t=a..b));
```

$$\int_{-1}^1 \frac{1}{t^6 + 2} dt = 0.943011015521611$$

Пример 8. Составить таблицу и построить график функции интегральный синус, являющейся интегралом с переменным верхним пределом

$$\text{Si}(x) = \int_0^x \frac{\sin t}{t} dt$$

на промежутке $[0, 16]$ с шагом $p = 1$.

Решение.

Алгоритм решения проиллюстрируем следующей схемой:

при $x_1 = a + p$ вычислить $y_1 = \int_a^h f(t)dt$;

при $x_2 = a + 2p$: $y_2 = \int_a^{x_2} f(t)dt = \int_a^{x_1} f(t)dt + \int_{x_1}^{x_2} f(t)dt = y_1 + \int_{x_1}^{x_2} f(t)dt$;

при $x_3 = a + 3p$: $y_3 = \int_a^{x_3} f(t)dt = \int_a^{x_2} f(t)dt + \int_{x_2}^{x_3} f(t)dt = y_2 + \int_{x_2}^{x_3} f(t)dt$;

и т.д. до достижения $x_m = b$.

Таким образом, вычислив y_1 при $x = a + p$, остальные значения при $x_k = a + kp$ вычисляем по рекуррентной формуле

$$y_k = y_{k-1} + \int_{x_{k-1}}^{x_k} \frac{\sin t}{t} dt, \quad k=2, \dots, m.$$

Для вычисления всех интегралов по интервалам (x_{k-1}, x_k) используем какой-либо метод численного интегрирования. Вот как может выглядеть программа с вычислением каждого частичного интеграла по методу трапеций.

> restart;

Подынтегральная функция имеет устранимую особенность при $t=0$, поэтому функцию определяем как кусочную:

> f:=x->piecewise(x<>0,1,sin(x)/x);

В Maple5.4 это делаем через процедуру:

**> f:=proc(x) local g; if x=0 then g:=1;else
g:=evalf(sin(x)/x); fi;RETURN(g) end;**

> a:=0: b:=16: # пределы интегрирования

Вычисление по формуле трапеций производим в составном варианте, т.е. каждый промежуток $[x_{k-1}, x_k]$ длиной **p** делится на частичные отрезки, число которых задается переменной **n**. Внутренний шаг формулы – переменная **h** – есть длина такого частичного отрезка.

> p:=1: m:=(b-a)/p: n:=8; h:=p/n:

n:=8

> X:=vector(m+1): Y:=vector(m+1):

X[1]:=a: Y[1]:=0: i:=2: w:=a: s:=0:

```

> while w<b do
  X[i]:=w+p;
  Ip:=evalf(p/(2*n)*(f(w)+f(w+h)));
  s:=s+Ip;
  Ip:=evalf(p/(2*n)*sum(f(w+(k-1)*h)+f(w+k*h),
k=2..n));
  s:=s+Ip;
  Y[i]:=s;
  w:=w+p;i:=i+1;
> od:

```

Табличные данные хранятся в двух массивах **X** и **Y**. Аппроксимируем функцию по методу сплайн-интерполяции. Используем кубический сплайн.

```

> readlib(spline):
> f1:=spline(X,Y,x,cubic):
> g5:=plot(f1,x=a..b,color=black):
g6:=plot([X[j],Y[j]]$j=1..m+1,x=a..b,style=POINT,
symbol=CIRCLE,color=black):
> g7:=plot(Si(x),x=a..b):

```

Совместим изображения найденных дискретных точек, построенной по ним аппроксимации и графика встроенной функции $Si(x)$.

```

> plots[display](g5,g6,g7);

```

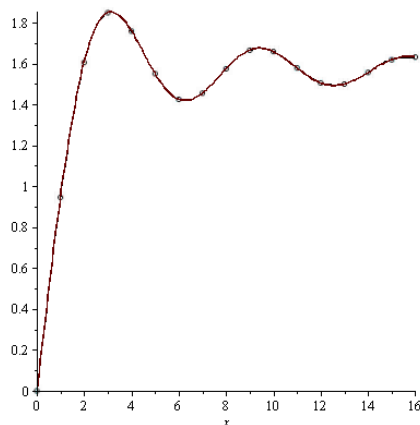


Рис. 4.7

Вывод таблицы значений функции с заданным шагом:

```
> for i from 1 to m+1 do
printf(`x=%g I=%g Si:=%g\n`,
X[i],Y[i],evalf(Si(X[i])));
od;
```

```
x=0 I=0 Si:=0
x=1 I=0.945691 Si:=0.946083
x=2 I=1.60485 Si:=1.60541
x=3 I=1.8482 Si:=1.84865
x=4 I=1.75805 Si:=1.7582
x=5 I=1.55006 Si:=1.54993
x=6 I=1.42491 Si:=1.42469
x=7 I=1.45472 Si:=1.4546
x=8 I=1.57414 Si:=1.57419
x=9 I=1.6649 Si:=1.66504
x=10 I=1.65825 Si:=1.65835
x=11 I=1.57832 Si:=1.57831
x=12 I=1.50507 Si:=1.50497
x=13 I=1.49945 Si:=1.49936
x=14 I=1.55622 Si:=1.55621
x=15 I=1.61812 Si:=1.61819
x=16 I=1.63123 Si:=1.6313
```

Соответствие хорошее. Совпадают четыре значащие цифры.

Как изменится точность, если уменьшать число n в формуле трапеций? Если взять меньше точек табулирования, т.е. увеличить шаг p ?

Упражнения

1. Вычислить интегралы по формулам трапеций и Симпсона и оценить погрешности.

1	$\int_1^2 \frac{dx}{x}, n=10$	2	$\int_1^3 \frac{dx}{1+x}, n=4$	3	$\int_0^1 \frac{dx}{1+x^2}, n=10$
4	$\int_{0,6\sqrt{2x^2+0,3}}^{1,3} \frac{1}{\sqrt{2x^2+0,3}} dx, n=17$	5	$\int_0^{1,2} \ln(1+x^2) dx, n=6$	6	$\int_{0,5}^{3,5} \sqrt{2x^2+3} dx, n=3, 6, 9$

7	$\int_0^1 \frac{dx}{1+x^3}, n=12$	8	$\int_4^{5,2} \ln x dx, n=6$	9	$\int_1^3 \sqrt{6x-5} dx, n=8$
10	$\int_0^1 e^{x^2} dx, n=10$	11	$\int_0^1 x^2 e^{-x^2} dx, n=10$	12	$\int_{0,1}^{1,4} \frac{\sin x}{x} dx, n=10$
13	$\int_0^1 \sin x^2 dx, n=10$	14	$\int_0^1 \cos x^2 dx, n=10$	15	$\int_0^{\pi/2} \sin x dx, n=4$
16	$\int_0^{\pi/2} \sqrt{1+\sin x^2} dx, n=10$	17	$\int_0^{\pi/2} \sqrt{1+\cos x^2} dx, n=10$	18	$\int_2^6 \ln(x^2+3,5) dx, n=8$
19	$\int_0^{2/3} \frac{\arctg x}{1+x} dx$	20	$\int_0^{\pi} \frac{\cos x}{\sqrt{1+x^2}} dx$	21	$\int_0^1 x^{3/2} e^{-x} dx$

2. Вычислить интегралы с точностью до $\frac{1}{2} \cdot 10^{-4}$ по формулам трапеций и Симпсона с заданной точностью, определяя шаг интегрирования h по оценке остаточного члена.

2.1. $\int_0^1 \frac{dx}{1+x}$.

2.2. $\int_0^1 \frac{dx}{1+x^3}$.

2.3. $\int_0^1 \sqrt{x} dx$.

2.4. $\int_0^1 e^{x^2} dx$.

2.5. $\int_2^4 e^{\frac{1}{x}} dx$.

2.6. $\int_0^{\pi/4} \sin x^2 dx$.

2.7. $\int_0^1 \cos x^2 dx$.

2.8. $\int_0^{\pi} \sqrt{3+\cos x} dx$.

2.9. $\int_0^{\pi/2} \sqrt{1+\cos x^2} dx$.

2.10. $\int_0^{\pi/2} \sqrt{1+\sin x^2} dx$.

2.11. $\int_0^{\pi/2} \sqrt{1-\frac{1}{4}\sin^2 x} dx$.

2.12. $\int_0^{2\pi} x \sin x dx$.

3. Вычислить интегралы по формулам прямоугольников, трапеций и Симпсона с точностью до $\frac{1}{2} \cdot 10^{-4}$, определяя шаг интегрирования с помощью двойного пересчета.

1	$\int_0^1 \frac{dx}{\sqrt{1+x^4}}$	2	$\int_0^{1/2} \frac{dx}{\sqrt[3]{1-x^2}}$	3	$\int_0^1 \frac{dx}{1-x+x^2}$
4	$\int_1^2 \frac{\sqrt{x^2-1}}{x^4} dx$	5	$\int_0^{1.5} \frac{x^4+1}{x^6+1} dx$	6	$\int_0^{\ln 2} \sqrt{e^x-1} dx$
7	$\int_0^1 x \ln(1+x) dx$	8	$\int_0^1 \frac{\ln(1+x)}{1+x^2} dx$	9	$\int_0^1 \frac{\ln(1+x^2)}{1+x^2} dx$
10	$\int_2^3 \frac{dx}{1+\sqrt{\ln x}}$	11	$\int_0^{\pi/3} \ln \cos x dx$	12	$\int_{10}^{100} \frac{\ln(1+x)}{x} dx$
13	$\int_0^1 \sqrt{x} \sin x dx$	14	$\int_0^1 \sqrt{x} \cos x dx$	15	$\int_0^{1/2} \cos \frac{x^2}{4} dx$
16	$\int_0^{\pi/2} \frac{dx}{x+\sqrt{\cos x}}$	17	$\int_0^{\pi} \frac{dx}{1+\sin^3 x}$	18	$\int_0^{\pi/2} \frac{dx}{1+\cos^2 x}$
19	$\int_0^1 \frac{\sin x}{1+x^2} dx$	20	$\int_0^{\pi} \frac{x \sin x}{1+\cos^2 x}$	21	$\int_0^{\pi/2} \left(\sqrt{1-\frac{1}{4} \sin^2 x} \right)^{-1} dx$
22	$\int_0^{\pi/2} \sqrt{1-\frac{1}{2} \sin^2 x} dx$	23	$\int_0^{2/3} \frac{\arcsin x}{x} dx$	24	$\int_0^{1/2} \frac{(\operatorname{arctg} x)^2}{x} dx$

4. Вычислить интегралы методом Гаусса. Оценить предельную погрешность.

Вариант	Интеграл	Вариант	Интеграл
1	$\int_{\pi/2}^{\pi} \sqrt{x} e^{-x^2} dx$	2	$\int_0^{\pi/3} \cos(2 \sin x) dx$
3	$\int_0^1 \cos(x+x^3) dx$	4	$\int_1^2 \frac{\ln x}{x+1} dx$

5	$\int_0^{\pi} e^{\sin^2 x} \sin 3x \, dx$	6	$\int_1^{20} \frac{\ln(1+x)}{x} \, dx$
7	$\int_0^1 \frac{e^{-x}}{\sqrt{1+x}} \, dx$	8	$\int_0^1 \sin(x^4 + 2x^3 + x^2) \, dx$
9	$\int_0^1 \frac{\cos x}{1+x^2} \, dx$	10	$\int_{\pi/4}^{\pi/2} \ln(\sin x) \, dx$
11	$\int_{\pi/2}^{\pi} x^4 e^{-x^2} \, dx$	12	$\int_0^{\pi} \cos(x - \sin x) \, dx$
13	$\int_{\pi/4}^{3\pi/4} e^{-\frac{x}{\sin x}} \, dx$	14	$\int_0^{2\pi/3} \ln(1 + \cos x) \, dx$
15	$\int_1^5 \frac{\sin x}{\sqrt{x}} \, dx$	16	$\int_0^3 e^{\cos x} \cos 2x \, dx$
17	$\int_{0,001}^3 \frac{x^2}{e^x - 1} \, dx$	18	$\int_{\pi/4}^{3\pi/4} e^{-\cos x} \cos(\sin x) \, dx$
19	$\int_{\pi/4}^{\pi/2} \sin(2 \cos x) \, dx$	20	$\int_0^1 \frac{x e^{-x}}{1+x^2} \, dx$
21	$\int_0^{\pi} \frac{\cos x}{\sqrt{1+x^2}} \, dx$	22	$\int_1^2 e^{-\left(x+\frac{1}{x}\right)} \, dx$
12	$\int_{0,1}^1 \cos x \ln x \, dx$	24	$\int_0^1 x^{3/2} e^{-x} \, dx$

5. РЕШЕНИЕ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ

5.1. Основные понятия

Обыкновенным дифференциальным уравнением порядка r называется уравнение

$$F(x, y(x), y'(x), \dots, y^{(r)}(x)) = 0, \quad (5.1)$$

которое связывает независимую переменную x , искомую функцию $y = y(x)$ и ее производные $y'(x)$, ..., $y^{(r)}(x)$.

Решение (интегрирование) дифференциального уравнения (3.1) заключается в отыскании функций (решений) $y(x)$, которые удовлетворяют этому уравнению для всех значений x в определенном конечном или бесконечном интервале (a, b) .

Общее решение обыкновенного дифференциального уравнения порядка r имеет вид

$$y = y(x, C_1, \dots, C_r),$$

где $C_1, \dots, C_r$ – произвольные постоянные, частный выбор которых дает частное решение.

В задаче Коши (начальной задаче) требуется найти частное решение, удовлетворяющее начальным условиям

$$y(x_0) = y_0, \quad y'(x_0) = y'_0, \quad \dots, \quad y^{(r-1)}(x_0) = y_0^{(r-1)},$$

по которым определяются r постоянных $C_1, \dots, C_r$.

Задача Коши для нормальной системы дифференциальных уравнений

$$y'_1 = f_1(x, y_1, \dots, y_n),$$

$$\dots\dots\dots$$

$$y'_n = f_n(x, y_1, \dots, y_n)$$

заключается в отыскании функций $y_1(x)$, ..., $y_n(x)$, удовлетворяющих этой системе и начальным условиям

$$y_1(x) = y_{10}, \quad \dots, \quad y_n(x) = y_{n0}.$$

Двухточечная граничная (краевая) задача для уравнения (5.1) ставится следующим образом: найти функцию $y(x)$, которая на отрезке $[a, b]$ удовлетворяет (5.1), а на концах отрезка – граничным условиям

$$\begin{aligned}\varphi_i(y(a), y'(a), \dots, y^{(r-1)}(a)) &= 0, \quad i=1, 2, \dots, L, \\ \psi_j(y(b), y'(b), \dots, y^{(r-1)}(b)) &= 0, \quad j=L+1, L+2, \dots, r.\end{aligned}$$

В важном частном случае линейной граничной задачи при $r=2$ дифференциальное уравнение и условия записываются в виде

$$\begin{aligned}y''(x) + p(x)y'(x) + q(x)y(x) &= f(x), \quad a \leq x \leq b; \\ \alpha_0 y'(a) + \beta_0 y(a) &= \gamma_0; \\ \alpha_1 y'(b) + \beta_1 y(b) &= \gamma_1 \quad (\alpha_0^2 + \beta_0^2 \neq 0, \alpha_1^2 + \beta_1^2 \neq 0).\end{aligned}\tag{5.2}$$

Методы решения обыкновенных дифференциальных уравнений можно разделить на графические, аналитические, приближенные и численные. Графические методы используют геометрические построения. В частности, метод изоклин, применяемый для решения уравнений 1-го порядка вида $y' = f(x, y)$, позволяет построить интегральные кривые по полю направлений, определенному изоклинами.

С помощью аналитических методов для некоторых классов дифференциальных уравнений удастся получить решение в виде конечных формул и выражений. С некоторыми этими методами студенты знакомятся в курсе дифференциальных уравнений, изучая решение уравнений первого порядка – с разделяющимися переменными, однородные, линейные, в полных дифференциалах и др., а также ряд уравнений высших порядков (например, линейные с постоянными коэффициентами). Решение, записанное в виде бесконечной суммы (функционального ряда), также принято считать аналитическим.

Многие приближенные методы используют подходы, позволяющие находить приближенные решения путем специального выбора классов функций. Например, широко известен

метод степенных рядов, который аппроксимирует решение отрезком ряда Тейлора. К приближенным методам также относят различные методы линеаризации, разложение решения по некоторому малому параметру, содержащемуся в задаче, асимптотические методы, метод последовательных приближений и т.д.

Наиболее значимыми являются численные методы решения дифференциальных уравнений, которые в настоящее время служат основным инструментом при исследовании научно-технических задач. Особенно эффективны эти методы в сочетании с использованием современных компьютеров. Суть большинства численных методов (так называемых конечно-разностных методов) состоит в следующем: область непрерывного изменения аргумента x (например, отрезок) заменяется дискретным множеством точек $\{x_i\}$, называемых узлами. Эти узлы составляют разностную сетку. Искомая функция непрерывного аргумента $y(x)$ приближенно заменяется узловыми значениями $\{y_i\}$ – функцией дискретного аргумента на заданной сетке. Эта функция называется сеточной. Исходное дифференциальное уравнение заменяется разностным уравнением относительно сеточной функции. При этом для входящих в уравнение производных используются соответствующие конечно-разностные соотношения. Такая замена дифференциального уравнения разностным называется разностной аппроксимацией. Таким образом, решение дифференциального уравнения сводится к получению числовой таблицы приближенных значений $\{y_i\}$ на сетке $\{x_i\} \in [x_0, b]$.

5.2. Численное решение задачи Коши

Многие дифференциальные уравнения не имеют аналитического решения. Кроме того, часто аналитическое решение и не нужно, но требуется получить результат в виде графических построений, либо в табулированном виде. В таких случаях решение дифференциальных уравнений ищется в численном виде, для чего Maple-функция **dsolve** используется с параметром **numeric** или **type=numeric**.

Пример 1. Решить задачу Коши

$$y' + y = \sin xy, \quad y(0) = 0,5.$$

Решение.

В Maple 14 численное решение уравнения осуществляется так:

```
> p:=dsolve({D(y)(x)+y(x)=sin(x*y(x)), y(0)=0.5}, y(x),  
type=numeric, range=0..4); # в range указывается диапа-  
зон изменения x
```

```
p := proc(x_rkf45) ... end proc
```

```
> plots[odeplot](p); # построение графика решения
```

Следует отметить, что в более ранних версиях, в частности, Maple 5.4, вызов этих функций отличается тем, что диапазон изменения указывается в **plots**, а не в **dsolve**:

```
> p:=dsolve({D(y)(x)+y(x)=sin(x*y(x)), y(0)=0.5}, y(x),  
type=numeric);
```

```
p := proc(x_rkf45) ... end proc
```

```
> plots[odeplot](p, 0..4); # построение графика решения
```

Помимо построения графика можно вычислить значение функции в любой заданной точке.

```
> for t from 0 by 0.4 to 4 do print(p(t)); od; # табули-  
рование функции
```

```
[x = 0., y(x) = 0.500000000000000]  
[x = 0.4, y(x) = 0.363017603454018012]  
[x = 0.8, y(x) = 0.308828294005778160]  
[x = 1.2, y(x) = 0.306867805078362366]  
[x = 1.6, y(x) = 0.352983256331245887]  
[x = 2.0, y(x) = 0.456367192785067156]  
[x = 2.4, y(x) = 0.608172116288803854]  
[x = 2.8, y(x) = 0.725942775091025338]  
[x = 3.2, y(x) = 0.743994997189059371]  
[x = 3.6, y(x) = 0.703144650928495874]  
[x = 4.0, y(x) = 0.649493152675693964]
```

Обратите внимание на особый вывод значений функции.

Пример 2. Решить задачу Коши для системы

$$y' = z, \quad z' = 0.1z - \sin y, \quad y(0)=0, \quad z(0)=1.$$

Решение.

```
> sys:=diff(y(x),x)=z(x),diff(z(x),x)=0.1*z(x)-sin(y(x));
```

$$sys := \frac{d}{dx} y(x) = z(x), \frac{d}{dx} z(x) = 0.1 z(x) - \sin(y(x))$$

```
> fcns:={y(x), z(x)};
```

$$fcns := \{y(x), z(x)\}$$

```
> F:=dsolve({sys,y(0)=0,z(0)=1},fcns,numeric); # чис-  
ленное решение
```

```
F:=proc(x_rkf45) ... end proc
```

```
> F(2); # вывод значений функций при x=2
```

$$[x=2., y(x)=1.12663272094226685, z(x)=-0.210795050787527832]$$

```
> plots[odeplot](F, [[x,y(x)], [x,z(x)]], 0..2.5,  
color=black); # графики функций
```

Ниже приводятся формулы итерационных процессов наиболее известных одношаговых методов решения задачи Коши.

Метод Эйлера:

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

Метод Эйлера с пересчетом:

$$y_{i+1} = y_i + \frac{h}{2}[f(x_i, y_i) + f(x_{i+1}, y_i + hf(x_i, y_i))], \quad i=0, 1, 2, \dots$$

Усовершенствованный метод Эйлера:

$$\tilde{y} = y_i + (h/2)f(x_i, y_i),$$

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

Метод Рунге-Кутты 4-го порядка:

$$y_{i+1} = y_i + \frac{h}{6}(k_0 + 2k_1 + 2k_2 + k_3), \quad i=0, 1, 2, \dots$$

где

$$k_0 = f(x_i, y_i), \quad k_1 = f(x_i + h/2, y_i + (h/2)k_0), \\ k_2 = f(x_i + h/2, y_i + (h/2)k_1), \quad k_3 = f(x_i + h, y_i + hk_2).$$

Пример 3. Решить задачу Коши $y' = x + y + \sin xy$, $y(0) = 0$.

Решение.

Составим программу, в которую заложены сразу все четыре приведенных выше метода. Выбор метода решения задачи определяется значением переменной **_Meth** (1 соответствует методу Эйлера, 2 – методу Эйлера с пересчетом, 3 – усовершенствованному методу Эйлера и 4 – метод Рунге–Кутты).

> **restart;**

> **f:=(x,y)->x+y+sin(x*y);** # задана функция $f(x, y)$ из правой части уравнения

$$f := (x, y) \rightarrow x + y + 1. \sin(xy)$$

> **u:=array(0..100,[]);** # выделение памяти под массив

> **Y1:=vector(101); X:=vector(101);** # выделение памяти под массивы

$$u := \text{array}(0 \dots 100, [\])$$

$$Y1 := \text{array}(1 \dots 101, [\])$$

$$X := \text{array}(1 \dots 101, [\])$$

Присвоение начальных значений: **a** – начальная точка отрезка (берется из начального условия), **b** – конечная точка (выбирается произвольно), **h** – шаг, **n** – число отсчетов, **y0** – начальное значение.

> **a:=0.; b:=2.; h:=0.1; n:=(b-a)/h;**

> **y0:=0;** # число **y0** берется из начального условия $y(a)=y_0$

> **u[0]:=y0;** # инициализация нулевого элемента массива

> **x:=a; k:='k';**

> **_Meth:=4;** # значение этой переменной определяет один из четырех реализованных методов

> **if _Meth=1 then** # если метод Эйлера ...

for i from 0 to n-1 do

u[i+1]:=evalf((u[i]+h*f(x,u[i])));

```

x:=x+h;
od:
fi;
> if _Meth =2 then # если метод Эйлера с пересчетом...
for i from 0 to n-1 do
ut:=u[i]+h*f(x,u[i]);
u[i+1]:=evalf(u[i]+h/2*(f(x,u[i])+f(x+h,ut)));
x:=x+h;
od:
fi;
> if _Meth =3 then # если усоверш. метод Эйлера ...
for i from 0 to n-1 do
ut:=u[i]+h/2*f(x,u[i]);
u[i+1]:=evalf(u[i]+h*f(x+h/2,ut));
x:=x+h;
od:
fi;
> if _Meth =4 then # если метод Рунге–Кутты ...
for i from 0 to n-1 do
k0:=f(x,u[i]); k1:=f(x+h/2,u[i]+h*k0/2);
k2:=f(x+h/2,u[i]+h*k1/2); k3:=f(x+h,u[i]+h*k2);
u[i+1]:=evalf(u[i]+h/6*(k0+2*k1+2*k2+k3));
x:=x+h;
od:
fi;
> Y1:=evalf([seq(u[k],k=0..n)]); # формирование век-
тора узловых значений
Y1:= [0., 0.005183979167, 0.02162386603, 0.05104378161, 0.09580415509,
0.1590857034, 0.2451322918, 0.3595432837, 0.5095314988, 0.7038353178,
0.9514154636, 1.257209511, 1.613830532, 1.995140009, 2.367069445, 2.715459640,
3.064688190, 3.481232674, 4.057674745, 4.767235270, 5.428150597]
> X:=evalf([seq(a+k*h,k=0..n)]); # формирование век-
тора узлов
X:= [0., 0.1000000000, 0.2000000000, 0.3000000000, 0.4000000000, 0.5000000000,
0.6000000000, 0.7000000000, 0.8000000000, 0.9000000000, 1.000000000,
1.100000000, 1.200000000, 1.300000000, 1.400000000, 1.500000000, 1.600000000,
1.700000000, 1.800000000, 1.900000000, 2.000000000]

```

Нанесём на координатной плоскости точки, изображающие найденное решение:

```
> g1:=plot([ [X[m], Y1[m]] $m=1..n+1], style=POINT,
symbol=CIRCLE, color=black) :
```

Для сравнения получим численное решение данной задачи встроенными средствами Maple:

```
> x:='x':y:='y':
p:=dsolve({diff(y(x), x)=f(x, y(x)), y(a)=y0}, y(x),
type=numeric);
> g2:=plots[odeplot](p, a..b): # график этого решения
> plots[display](g1, g2): # совмещение двух полученных
решений
```

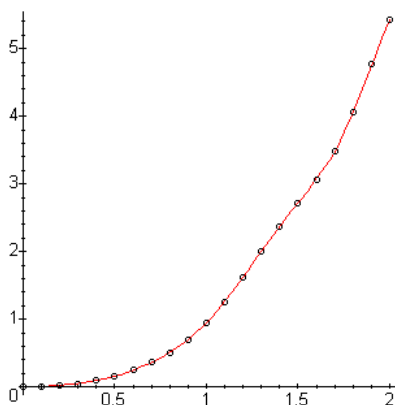


Рис. 5.1

5.3. Решение краевой задачи методом стрельбы

Рассмотрим краевую задачу для уравнения второго порядка, разрешенного относительно второй производной:

$$y'' = f(x, y, y'). \quad (5.3)$$

Будем искать решение $y = y(x)$ этого уравнения на отрезке $[a, b]$. Граничные условия на концах рассматриваемого отрезка примем в простом виде

$$y(a) = y_0, \quad y(b) = y_1, \quad (5.4)$$

что получается из (5.2) при $\alpha_0 = \alpha_1 = 0$ и соответствует граничным условиям 1-го рода.

Сущность метода стрельбы заключается в сведении решения краевой задачи (5.3), (5.4) к решению последовательности задач Коши для того же уравнения (3.3) с начальными условиями

$$y(a) = y_0, \quad y'(a) = \operatorname{tg} \varphi. \quad (5.5)$$

Здесь y_0 – точка на оси ординат, в которой помещается начало искомой интегральной кривой; φ – угол наклона касательной к интегральной кривой в этой точке. Считая решение задачи Коши $y = y(x, \varphi)$ зависящим от параметра φ , будем искать такую интегральную кривую $y = y(x, \varphi^*)$, которая выходит из точки (a, y_0) и попадает в точку (b, y_1) . Таким образом, если $\varphi = \varphi^*$, то решение $y(x, \varphi)$ задачи Коши совпадает с решением $y(x)$ краевой задачи. При $x = b$, учитывая второе граничное условие (3.4), получаем $y(b, \varphi) = y_1$, или

$$y(b, \varphi) - y_1 = 0. \quad (5.6)$$

Следовательно, для нахождения параметра φ получим уравнение вида $F(\varphi) = 0$, где $F(\varphi) = y(b, \varphi) - y_1$. Это уравнение отличается от привычной записи тем, что функцию $F(\varphi)$ нельзя представить в виде некоторого аналитического выражения, поскольку она является решением задачи Коши (5.3), (5.4). Тем не менее, для решения уравнения (5.6) может быть использован любой из рассмотренных ранее методов решения нелинейных уравнений. В частности, одним из самых надежных является метод Ньютона. Его применение состоит в следующем. Пусть φ_0 – начальное приближение к φ^* . Построим итерационный процесс для нахождения последующих приближений φ_k с помощью формулы метода Ньютона:

$$\varphi_k = \varphi_{k-1} - \frac{F(\varphi_{k-1})}{F'(\varphi_{k-1})}.$$

С учетом того, что $F'(\varphi) = \partial y(x, \varphi) / \partial \varphi$, имеем

$$\varphi_k = \varphi_{k-1} + \frac{y_1 - y(b, \varphi_{k-1})}{\partial y(b, \varphi_{k-1}) / \partial \varphi}. \quad (5.7)$$

Производную в знаменателе этого выражения можно найти численно:

$$\frac{\partial y(x, \varphi)}{\partial \varphi} \approx \frac{y(x, \varphi + \Delta\varphi) - y(x, \varphi)}{\Delta\varphi}, \quad (5.8)$$

где $\Delta\varphi$ – произвольное малое возмущение φ .

Для вычисления правой части (5.8) нужно решить задачу Коши при $\varphi = \varphi_{k-1} + \Delta\varphi$, в результате чего найдем значение $y(x, \varphi_{k-1} + \Delta\varphi)$. Затем по формуле (5.7) находим следующее приближение φ_k параметра φ^* и т.д. Этот итерационный процесс продолжается до тех пор, пока два последовательных приближения φ_{k-1} и φ_k не станут отличаться меньше, чем на заданное малое число ε .

Описанный алгоритм называется методом стрельбы вполне оправданно, поскольку в нем как бы проводится «пристрелка» по углу наклона интегральной кривой в начальной точке с тем, чтобы попасть в точку (b, y_1) . Следует отметить, что этот алгоритм хорошо работает в том случае, если решение $y(x, \varphi)$ не слишком чувствительно к изменениям φ ; в противном случае можно столкнуться с неустойчивостью.

Замечание. Вместо угла φ итерационный процесс можно строить для тангенса φ . Вычислительная процедура при этом почти не меняется.

Пример 4. Решить краевую задачу

$$y'' - y = x, \quad y(0) = 0, \quad y(1) = 0.$$

Решение. Имеем краевую задачу 1-го рода. Программа на языке Maple может быть реализована следующим образом.

```
> restart; # очистка оперативной памяти
> eqn:=(D@@2)(y)(x)-y(x)=x; # задание дифференциального
уравнения
> tgphi:=0: eps:=0.001: # инициализация переменных
> n:=0; Y1:=0; YB:=0; a:=0; b:=1;
```

Поясним смысл введенных переменных: **tgphi** – $\operatorname{tg} \varphi$ – тангенс угла наклона кривой в левой граничной точке $x=0$, именно эта величина подлежит уточнению с помощью итераций; **eps** – малое число – используется в условии прекращения итераций; а также для приращения величины $\operatorname{tg} \varphi$; **n** – число итераций, **y1** – точное значение y_1 из граничного условия в правой точке: $y(1)=y_1$; **YB** – приближение y_1 ; **a**, **b** – левая и правая граничные точки.

> **Digits:=15;** # задание числа значащих цифр в вычислениях (по умолчанию 10)

Далее идет цикл, на каждом шаге которого решаются две близкие задачи Коши, отличающиеся только граничным условием в точке $x=1$, и вычисляется поправка к **tgphi**. Как только значение текущей задачи Коши при $x=1$ (переменная **YB**) приблизится к y_1 , цикл останавливается.

```
> while (abs(YB-Y1)>eps) or (n=0) do
> p0:=dsolve({eqn,y(a)=0.,D(y)(a)=tgphi},y(x),
type=numeric,output=listprocedure); # решение начальной задачи с условиями  $y(0)=0$ ,  $y'(0)=\operatorname{tg} \varphi$ 
> f:=subs(p0,y(x)); YB:=f(b); # YB – значение найденного решения при  $x=1$ 
> p1:=dsolve({eqn,y(a)=0.,D(y)(a)=tgphi+eps},y(x),type=numeric,output=listprocedure); # решение возмущенной начальной задачи с условиями  $y(0)=0$ ,  $y'(0)=\operatorname{tg} \varphi + \varepsilon$ 
> f1:=subs(p1,y(x)); # YB – значение возмущенного решения при  $x=1$ 
> n:=n+1; # счетчик числа итераций
> tgphi:=tgphi+(Y1-YB)/(f1(b)-YB)*eps; # уточнение  $\operatorname{tg} \varphi$  по методу Ньютона
> od;
> tgphi; YB; # вывод  $\operatorname{tg} \varphi$ , равного  $y'(0)$ , и YB – значения  $y(x)$  в правой граничной точке  $x=1$ 
> g1:=plots[odeplot](p0,a..b,colour=black);
> plots[display](g1);
```

Интересно сравнить полученный результат с точным решением, тем более, что данная краевая задача имеет аналити-

ческое решение, которое без труда найдем с помощью встроенной функции dsolve:

```
> p:=dsolve({eqn,y(0)=0.,y(1)=0},y(x));
> u:=unapply(subs(p,y(x)),x);
```

Протабулируем полученное методом стрельбы решение вместе с точным решением с шагом 0.1:

```
> for t from a to b by 0.1 do
> printf(`x=%g y=%g u=%g\n`,t,f(t),evalf(u(t)));
> od;
> g2:=plot(u(x),x=a..b): # график точного решения
> plots[display](g1,g2); # совмещение графиков прибли-
женного и точного решений
```

Решение граничной задачи второго рода

$$y'' = f(x, y, y'), \quad y'(a) = M_0, \quad y'(b) = M_1$$

по методу стрельбы сводится к последовательному решению задач Коши

$$y'' = f(x, y, y'), \quad y'(a) = M_0, \quad y(a) = y_0,$$

причем «пристрелка» проводится по параметру y_0 , значение которого целенаправленно подбирается так, чтобы найденное решение $\tilde{y}$ приблизительно удовлетворяло условию в точке $x = b$:

$$\tilde{y}'(b) \approx M_1.$$

Пример 5. Решить краевую задачу

$$y'' - y = x, \quad y'(0) = -1, \quad y'(1) = 1.$$

Решение в системе компьютерной математики Maple:

```
> restart;
> Digits:=15; eps:=0.001; a:=0; b:=1; M0:=-1.; M1:=1;
# eps – малое число – используется в условии завершения цикла, а
также для приращения величины  $y_0$ ; a, b – левая и правая гранич-
ные точки; M0 и M1 – заданные граничные значения  $y'$  на концах от-
резка  $[a, b]$ .
> n:=0; y0:=0; # n – число итераций, y0 – приближение вели-
чины  $y_0$ 
```

```

> eqn:=(D@@2) (y) (x)-y(x)=x;
> while (abs(dy1-M1)>eps) or (n=0) do
> p0:=dsolve({eqn,y(a)=y0,D(y)(a)=M0},y(x),
type=numeric,output=listprocedure);
> q:=subs(p0,diff(y(x),x)); dy1:=q(b);
> p1:=dsolve({eqn,y(a)=y0+eps,D(y)(a)=M0},y(x),
type=numeric,output=listprocedure);
> q1:=subs(p1,diff(y(x),x));
> n:=n+1;
> y0:=y0-(dy1-M1)/(q1(b)-dy1)*eps;
> od;
> y0; dy1;    # dy1 – приближение величины M1
> g1:=plots[odeplot](p0,a..b,colour=black):
> plots[display](g1);

```

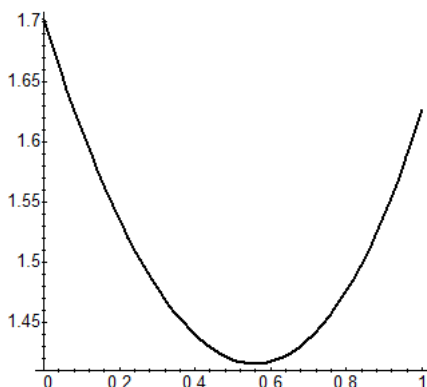


Рис. 5.2

```

> for t from a to b by 0.1 do printf(`x=%g
y=%g \n`,t,subs((p0[2]),y(x))(t)); od;

```

Ниже приведена программа решения краевой задачи для уравнения 2-го порядка с общими граничными условиями 3-го рода

$$A_0 y'(a) + B_0 y(a) = C_0,$$

$$A_1 y'(b) + B_1 y(b) = C_1.$$


```

> restart;
> Digits:=15; eps:=0.001; A0:=-1; B0:=0; C0:=2;
A1:=0.; B1:=1; C1:=1; a:=0; b:=1;
> eqn:=(D@@2)(y)(x)-y(x)=x; # дифф. уравнение
> tga:=0; y0:=0; n:=0; dy1:=0;g:=0;

> if (A0<>0) then
while (abs(g-C1)>eps) or (n=0) do
p0:=dsolve({eqn,y(a)=y0,D(y)(a)=(C0-B0*y0)/A0},
y(x),type=numeric);
g:=A1*subs(p0(b)[3],diff(y(x),x))+B1*subs(p0(b)[2],
y(x));
p1:=dsolve({eqn,y(a)=y0+eps,
D(y)(a)=(C0-B0*(y0+eps))/A0}, y(x),type=numeric);
g1:=A1*subs(p1(b)[3],diff(y(x),x))+B1*subs(p1(b)[2],y(x));
n:=n+1;
y0:=y0-(g-C1)/(g1-g)*eps;
od;

else

> while (abs(g-C1)>eps) or (n=0) do
p0:=dsolve({eqn,y(a)=(C0-A0*tga)/B0,D(y)(a)=tga},
y(x),type=numeric);
g:=A1*subs(p0(b)[3],diff(y(x),x))+B1*subs(p0(b)[2],
y(x));
p1:=dsolve({eqn,y(a)=(C0-A0*(tga+eps))/B0,
D(y)(a)=tga+eps},y(x),type=numeric);
g1:=A1*subs(p1(b)[3],diff(y(x),x))+B1*subs(p1(b)[2],
y(x));
n:=n+1;
tga:=tga-(g-C1)/(g1-g)*eps;
od;
fi;

> for t from a to b by 0.1 do
printf(`t=%g    y=%g\n`,t,subs(p0[2](t),y(x)));
od;

> g2:=plots[odeplot](p0,a..b,colour=black):
> plots[display](g2);

```

Упражнения

1. Решить задачу Коши для дифференциального уравнения $y' = f(x, y)$ на отрезке $[a, b]$ при начальном заданном условии $y(a) = y_0$ и шаге интегрирования h :

- 1) методом Эйлера и усовершенствованным (модифицированным) методом Эйлера с шагом h и $h/2$,
- 2) методом Рунге-Кутты с шагом h и $2h$,
- 3) методом Адамса.

№	Функция $f(x, y)$	Интервал $[a, b]$	y_0	Шаг h
1	$f(x, y) = 3x^2 + 0,1xy$	$[0; 1]$	$y(0) = 0,2$	0,1
2	$f(x, y) = 0,185(x^2 + \cos(0,7x)) + 1,843y$	$[0,2; 1,2]$	$y(0,2) = 0,25$	0,1
3	$f(x, y) = x + \cos\left(\frac{y}{3}\right)$	$[1,6; 2,6]$	$y(1,6) = 4,6$	0,1
4	$f(x, y) = x + \sin\left(\frac{y+1}{\sqrt{13}}\right)$	$[0,2; 1,2]$	$y(0,2) = 1,1$	0,1
5	$f(x, y) = x + \cos\left(\frac{y}{e}\right)$	$[1,4; 2,4]$	$y(1,4) = 2,5$	0,1
6	$f(x, y) = x + \cos\left(\frac{y}{\pi}\right)$	$[1,7; 2,7]$	$y(1,7) = 5,3$	0,1
7	$f(x, y) = -3y^2 + \sqrt{4x^2 + 1}$	$[2,6; 4,6]$	$y(2,6) = 3,5$	0,2
8	$f(x, y) = 2 - \sin^2(x + y)$	$[2; 3]$	$y(2) = 2,3$	0,1
9	$f(x, y) = 1,6x + 0,5y^2$	$[0; 1]$	$y(0) = 0,3$	0,1
10	$f(x, y) = x - \cos\left(\frac{y}{\sqrt{5}}\right)$	$[1,8; 2,8]$	$y(1,8) = 2,6$	0,1
11	$f(x, y) = x + \cos\left(\frac{y}{\sqrt{11}}\right)$	$[2,1; 3,1]$	$y(2,1) = 2,5$	0,1

№	Функция $f(x, y)$	Интервал [a, b]	y_0	Шаг h
12	$f(x, y) = e^{2x} + 0,25y^2$	[0; 0,5]	$y(0) = 2,6$	0,05
13	$f(x, y) = x + \cos\left(\frac{y}{\sqrt{2}}\right)$	[-2; -1]	$y(-2) = 3$	0,1
14	$f(x, y) = 0,133 \cdot (x^2 + \sin(2x)) + 0,872y$	[0,2; 1,2]	$y(0,2) = 0,25$	0,1
15	$f(x, y) = \sin(x + y) + 1,5$	[1,5; 2,5]	$y(1,5) = 4,5$	0,1
16	$f(x, y) = x + \cos\left(\frac{y}{1,25}\right)$	[0,4; 1,4]	$y(0,4) = 0,8$	0,1
17	$f(x, y) = 2,5x + \cos(y + 0,6)$	[1; 3]	$y(1) = 1,5$	0,2
18	$f(x, y) = \cos(1,5y + x)^2 + 1,4$	[1; 2]	$y(1) = 1,5$	0,1
19	$f(x, y) = \frac{1}{1 + x^3 y} + 2y$	[1,5; 2]	$y(1,5) = 2,1$	0,05
20	$f(x, y) = \cos(y) + 3x$	[0; 2]	$y(0) = 1,3$	0,1
21	$f(x, y) = \cos(1,5x - y^2) - 1,3$	[-1; 1]	$y(-1) = 0,2$	0,2
22	$f(x, y) = x + \sin\left(\frac{y}{3}\right)$	[1,6; 2,6]	$y(1,6) = 4,6$	0,1
23	$f(x, y) = e^{-(y-1)} + 2x$	[0; 0,5]	$y(0) = 0,3$	0,05
24	$f(x, y) = 1 + 2y \cdot \sin(x) - y^2$	[1; 2]	$y(1) = 0$	0,1
25	$f(x, y) = \frac{\cos(y)}{x + 2} - 0,3y^2$	[0; 1]	$y(0) = 0$	0,1
26	$f(x, y) = 0,166 \cdot (x^2 + \sin(1,1 \cdot x)) + 0,883 \cdot y$	[0,2; 1,2]	$y(0,2) = 0,25$	0,1

№	Функция $f(x, y)$	Интервал [a, b]	y_0	Шаг h
27	$f(x, y) = x + \cos\left(\frac{y+x}{\pi}\right)$	[1,7; 2,7]	$y(1,7) = 5,6$	0,1
28	$f(x, y) = x + \cos\left(\frac{y}{e}\right)$	[1,4; 2,4]	$y(1,4) = 2,5$	0,1
29	$f(x, y) = x + \sin\left(\frac{y}{\sqrt{10}}\right)$	[0,6; 1,6]	$y(0,6) = 0,8$	0,1
30	$f(x, y) = y + \cos\left(\frac{x}{\pi}\right)$	[1; 2]	$y(1) = 5,9$	0,1
31	$f(x, y) = 1 + 0,8y \cdot \sin(x) - 2y^2$	[0; 1]	$y(0) = 0$	0,1
32	$f(x, y) = x + \sin\left(\frac{y}{2,5}\right)$	[0,5; 1,5]	$y(0,5) = 1,8$	0,1
33	$f(x, y) = x + \cos\left(\frac{y}{\sqrt{1,3}}\right)$	[1,2; 2,2]	$y(1,2) = 1,8$	0,1
34	$f(x, y) = 1 + 2,2 \cdot \sin(x) + 1,5y^2$	[0; 1]	$y(0) = 0$	0,1
35	$f(x, y) = \frac{\cos(y)}{x+2} - 0,3y^2$	[0; 1]	$y(0) = 0$	0,1
36	$f(x, y) = \frac{\cos(y)}{1,5+x} - 1,2y^2$	[0; 1]	$y(0) = 0$	0,1
37	$f(x, y) = \cos(x-y) + \frac{1,25y}{1,5+x}$	[0; 1]	$y(0) = 0$	0,1
38	$f(x, y) = 0,2x^2 + y^2$	[0; 1]	$y(0) = 0,8$	0,1
39	$f(x, y) = x^2 + y$	[0; 1]	$y(0) = 0,4$	0,1
40	$f(x, y) = xy + 0,1y^2$	[0; 1]	$y(0) = 0,5$	0,1

2. Методом пристрелки найти решение граничных задач в точках $x_k = 0, 1k$.

Вариант 1. $y''(x) + 2y'(x) - \frac{4}{x}y(x) = 1$,
 $0.5 \leq x \leq 1$, $y'(0.5) = 1.5$, $y(1) + y'(1) = 4$.

Вариант 2. $y''(x) - \frac{6x}{3x^2 - 0.5}y'(x) - \frac{1}{x}y(x) = 0.5 - x^2$,
 $0.5 \leq x \leq 1$, $y'(0.5) = 0.25$, $2y(1) + y'(1) = 3.5$.

Вариант 3. $y''(x) + \frac{1}{x}y'(x) - \frac{1}{x^2}y(x) = -\frac{2}{x^3}$,
 $0.5 \leq x \leq 1$, $y(0.5) = -2\ln 2$, $y(1) = 0$.

Вариант 4. $y''(x) + \frac{1}{x}y'(x) = \frac{1}{x}$, $0.5 \leq x \leq 1$, $y'(0.5) = 3$, $y(1) = 1$.

Вариант 5. $y''(x) + 2xy'(x) - y(x) = 2(x^2 + 1)\cos x$,
 $0 \leq x \leq 0.5$, $y(0) = 0$, $y(0.5) = 0.5\sin 0.5$.

Вариант 7. $x^2 \ln x \cdot y''(x) - xy'(x) + y(x) = 0$,
 $y(1) = 0$, $y(e) = e - 2$.

Вариант 8. $y''(x) - \ln x \cdot y'(x) - 2y(x) = 1$, $y(0.5) + y'(0.5) = 1$,
 $y(1.5) - y'(1.5) = 1$.

Вариант 9. $y''(x) + y'(x) - \frac{2}{\cos^2 x}y(x) = \frac{2}{\cos^2 x}$, $y(0) = 0$,
 $y'(0.5) = 1/\cos^2 0.5$.

Вариант 10. $y''(x) - \frac{2}{x}y'(x) - \frac{4}{x^2 + 2}y(x) = 8$, $y'(0.5) = 0.5$,
 $y(1) + y'(1) = 1$.

6. ПРИБЛИЖЕНИЕ (АППРОКСИМАЦИЯ) ФУНКЦИЙ

6.1. Введение

Пусть величина y является функцией аргумента x . Это означает, что любому значению x из области определения поставлено в соответствие число y . Вместе с тем на практике часто неизвестна явная связь между y и x , т.е. невозможно записать эту связь в виде некоторой зависимости $y=f(x)$. Иногда даже известная зависимость $y=f(x)$ оказывается настолько громоздкой (например, содержит трудно вычисляемые выражения, сложные интегралы и т.п.), что ее использование в практических расчетах требует слишком много времени.

В этой связи и возникает задача о приближении, или *аппроксимации*, функции, состоящая в том, чтобы данную функцию $f(x)$ приближенно заменить (аппроксимировать) некоторой функцией $g(x)$ так, чтобы отклонение (в некотором смысле) $g(x)$ от $f(x)$ в заданной области было наименьшим. Функция $g(x)$ при этом называется аппроксимирующей. Что касается самого понятия «наименьшее отклонение», то оно уточняется при рассмотрении конкретных способов аппроксимации.

Аппроксимация, при которой приближение строится на дискретном множестве точек, называется *точечной* (дискретной). Это наиболее распространенный и практически важный случай, когда точный вид связи между величинами x и y неизвестен, но эта связь задается в виде таблицы $\{x_i, y_i\}$. Это означает, что дискретному множеству значений аргумента $\{x_i\}$ поставлено в соответствие множество значений функции $\{y_i\}$ ($i = 1, 2, \dots, n$). Эти значения – либо результаты расчетов, либо экспериментальные данные. На практике могут понадобиться значения величины и в других точках, отличных от x_i . Однако зачастую получить эти значения можно лишь путем очень сложных расчетов или проведением дорогостоящих экспериментов. Вот здесь и приходит на помощь точечная аппроксимация. К ней относятся интерполирование, среднеквадратическое приближение и др.

При построении приближения на непрерывном множестве точек (например, на отрезке) аппроксимация называется *непрерывной*, или интегральной. Такая аппроксимация возникает, когда точная связь известна, но ей по каким-то причинам пользоваться нецелесообразно. К непрерывной аппроксимации относится, например, приближение на основе метода взвешенных невязок.

Отметим, что аппроксимация функций – это традиционно один из основных разделов численных методов. Разработанные давно, методы аппроксимации при вычислениях вручную могли использоваться лишь для решения не слишком трудоемких задач. С появлением компьютеров начался период их широкого и эффективного внедрения в практику. В настоящее время они имеют не только самостоятельное применение, но и являются весьма важной частью многих других методов вычислительной математики (например, методов решения дифференциальных краевых задач).

6.2. Интерполирование

Линейную комбинацию

$$g(x) = c_1\varphi_1(x) + c_2\varphi_2(x) + \dots + c_n\varphi_n(x) = \sum_{k=1}^n c_k\varphi_k(x) \quad (6.1)$$

с действительными коэффициентами c_k называют обобщенным многочленом по системе функций φ_k ($k = 1, \dots, n$).

Задача интерполирования функции $f(x)$ состоит в том, чтобы найти значение $f(x)$ при $x \neq x_m$, $m = 1, 2, \dots, n$, если известны узлы интерполирования $x_1, x_2, \dots, x_n$ и значения функции $f(x)$ в этих узлах. Решается эта задача следующим образом: выбирается система функций $\{\varphi_k\}$, строится обобщенный многочлен (6.1), а коэффициенты c_k задаются таким образом, чтобы в узлах интерполирования значения обобщенного многочлена совпадали со значениями данной функции $f(x)$:

$$g(x_m) = \sum_{k=1}^n c_k\varphi_k(x_m) = f(x_m), \quad m = 1, 2, \dots, n. \quad (6.2)$$

$$L_{n-1}(x) = \sum_{k=1}^n f(x_k) l_k(x) = \left(\sum_{k=1}^n \frac{A_k}{x - x_k} f(x_k) \right) \left(\prod_{k=1}^n (x - x_k) \right), \quad (6.4)$$

где

$$l_k(x) = \frac{(x - x_1) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_n)}{(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)} = \prod_{\substack{i=1, \\ i \neq k}}^n \frac{x - x_i}{x_k - x_i};$$

$$A_k = \frac{1}{(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)} = \prod_{\substack{i=1, \\ i \neq k}}^n \frac{1}{x_k - x_i}.$$

$L_m(x)$ носит название многочлена Лагранжа.

Рассмотрим на примере различные способы построения интерполяционного полинома в системе Maple.

Пример. Построить многочлен наименьшей степени, принимающий в данных точках следующие значения:

x	0.5	2	3	4	5
$f(x)$	0.3	1	2	2	1

1-й способ: на основе прямого решения системы (6.3).

Зададим два массива **X** и **Y** – соответственно абсциссы и ординаты отдельных точек функции:

> **X:=[0.5, 2, 3, 4, 5]; Y:=[0.3, 1, 2, 2, 1];**

$X := [0.5, 2, 3, 4, 5]$

$Y := [0.3, 1, 2, 2, 1]$

> **n:=5;** # число узлов

$n := 5$

Определим диапазон $[a, b]$ изменения x , который потребуется в дальнейшем для построения графика:

> **a:=min(X[i]\$i=1..n); b:=max(X[i]\$i=1..n);**

$a := 0.5$

$b := 5$

Может оказаться полезным слегка расширить этот диапазон, взяв его в виде $[a - \delta, b + \delta]$, где δ достаточно мало, например, положить

> **delta:=(b-a)/n/2.;**

$\delta := 0.4500000000$

и тем самым

```
> a:=a-delta: b:=b-delta:
```

Поскольку число узлов 5, искомым интерполяционный полином будет степени на единицу меньше, т.е. 4. Введем его в общем виде с пока неопределенными коэффициентами:

```
> g:=t->c0+c1*t+c2*t^2+c3*t^3+c4*t^4;  
g := t → c0 + c1 t + c2 t2 + c3 t3 + c4 t4
```

Составим систему уравнений (6.3) и присвоим её переменной **eqns**:

```
> eqns:={g(X[1])=Y[1], g(X[2])=Y[2], g(X[3])=Y[3],  
g(X[4])=Y[4], g(X[5])=Y[5]};  
eqns:={c0+0.5c1+0.25c2+0.125c3+0.0625c4=0.3,  
c0+2c1+4c2+8c3+16c4=1, c0+3c1+9c2+27c3+81c4=2,  
c0+4c1+16c2+64c3+256c4=2, c0+5c1+25c2+125c3+625c4=1}
```

Заметим, что то же самое можно было сделать короче с помощью команды

```
> eqns:={ (g(X[i])=Y[i]) $i=1..n};
```

Решим эту систему:

```
> r:=solve(eqns, {c0, c1, c2, c3, c4});  
r := {c4 = 0.045291005, c3 = -0.63407407, c2 = 2.7156614,  
c1 = -3.4748148, c0 = 1.4349206}
```

Подставим эти значения коэффициентов в интерполяционный полином **g**:

```
> subs(r, g(t));  
r1 := 1.434921 - 3.474815 t + 2.715661 t2 - 0.6340741 t3 + 0.04529101 t4
```

Это и есть искомым интерполяционный многочлен, записанный в виде простого выражения. Чтобы его использовать в дальнейших вычислениях как Maple-функцию, целесообразно воспользоваться строкой

```
> fu:=unapply(subs(r, g(t)), t);
```

Вычислим эту функцию в ряде точек:

```
> {[1, fu(1)], [2, fu(2)], [3.5, fu(3.5)], [4.82, fu(4.82)]};  
{{[3.5, 2.1504764], [4.82, 1.219491], [1, 0.086984135], [2, 1.0000005]}}
```

Построим график (интервал, на котором он строится, должен содержать все узлы):

```
> g1:=plot(fu(x),x=a..b):# вывод изображения будет позже
```

Для того чтобы наглядно убедиться в прохождении интерполирующей функции через заданные точки функции (те, что из таблицы), полезно на график нанести эти точки. Это можно сделать так:

```
> g2:=plot([X[i],Y[i]]$i=1..n),x=a..b,style=POINT,
symbol=CIRCLE,color=black);
g2:=PLOT(CURVES(
[[0.5000000000000000,0.3000000000000000],[2.,1.],[3.,2.],[4.,2.],[5.,1.]]),
SYMBOL(CIRCLE),COLOUR(RGB,0,0,0),AXESLABELS(x, ),
STYLE(POINT),VIEW(0..5.,DEFAULT))
```

(Результат сохраняется в переменной **g2**, но графически пока не отображается).

Для графического вывода построений (как по отдельности, так и совместно) используется функция **display** из пакета графических построений **plots**. В частности, следующая команда совмещает два последних объекта **g1** и **g2**:

```
> plots[display](g1,g2);
```

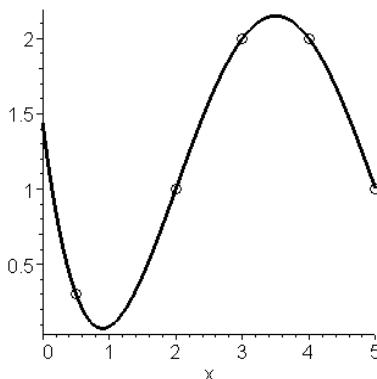


Рис. 6.1

2-й способ: на основе формулы Лагранжа (4.4).

```
> X:=[0.5,2,3,4,5]: Y:=[0.3,1,2,2,1]: n:=5:
> a:=min(X[i]$i=1..n): b:=max(X[i]$i=1..n):
> r1:=sum(Y[k]/product(piecewise(i=k,1,(X[k]-X[i])),
i=1..n)/(x-X[k]),k=1..n)*product((x-X[k]),k=1..n);
```

$$r1 := \left(\frac{0.005079365080}{x-0.5} - \frac{0.1111111111}{x-2.} + \frac{0.4000000000}{x-3.} - \frac{0.2857142857}{x-4.} + \frac{0.03703703704}{x-5.} \right) (x-0.5)(x-2)(x-3)(x-4)(x-5)$$

Смысл использования функции **piecewise**, стоящей внутри первого произведения, – отсеять сомножитель $(x_k - x_i)$ при $i = k$.

> simplify(r1) ;

$$-3.474815 x + 0.04529100 x^4 - 0.6340740 x^3 + 2.715661 x^2 + 1.434921$$

Команды, отвечающие за построение графика, – почти такие же, как в 1-м способе.

График интерполирующей функции сохраним в **g1**:

> g1:=plot(r1,x=a..b) :

Нанесём также заданные точки функции, заданные координатами в массивах **X** и **Y**:

> g2:=plot([X[i],Y[i]]\$i=1..n),x=a..b,style=POINT, symbol=CIRCLE,color=black) :

Совместим два последних изображения:

> plots[display](g1,g2) ;

Картинка не отличается от приведенной на рис. 6.1.

6.3. Локальная интерполяция

В алгебраическом интерполировании при росте числа узлов увеличивается, как правило, степень интерполяционного многочлена. Кроме того, когда интерполирование выполняется для функций, не являющихся достаточно гладкими, интерполирование высокого порядка нецелесообразно. В этих случаях лучший результат может дать локальная (кусочная) интерполяция.

В отличие от глобальной интерполяции, когда интерполирующая функция $g(x)$ строится сразу для всего рассматриваемого интервала изменения x , при локальной интерполяции такая функция ищется отдельно для разных частей этого интервала, а затем как бы сшивается из соответствующих отдельных кусков. При локальной интерполяции должно сохраняться основное условие интерполирования: в узлах x_i значе-

ния интерполирующей функции должны быть равны значениям данной функции $y_i = f(x_i)$.

Простейшим и часто используемым видом локальной интерполяции является линейная (*кусочно-линейная*) интерполяция. Она состоит в том, что заданные точки (x_k, y_k) ($k=1, 2, \dots, n$) соединяются прямолинейными отрезками, и функция $f(x)$ приближается ломаной с вершинами в этих точках.

Уравнения каждого отрезка ломаной в общем случае разные. Поскольку имеется $n-1$ интервалов (x_k, x_{k+1}) , то для каждого из них в качестве уравнения интерполяционного многочлена используется уравнение прямой, проходящей через две заданные точки. В частности для k -го интервала можно написать уравнение прямой, проходящей через две точки (x_k, y_k) и (x_{k+1}, y_{k+1}) в виде

$$y = \frac{y_{k+1} - y_k}{x_{k+1} - x_k} (x - x_k) + y_k, \quad x_k \leq x \leq x_{k+1}. \quad (6.5)$$

Следовательно, при использовании линейной интерполяции сначала нужно определить интервал, в который попадает значение аргумента, а затем подставить его в формулу (6.5) и найти приближенное значение в данной точке.

Приведем реализацию этого алгоритма в среде Maple для таблично заданной функции из предыдущего примера.

```
> restart;
```

```
> x:=[0.5,2,3,4,5];y:=[0.3,1,2,2,1];n:=5: # исходные данные
```

Программируем формулу (6.5), причем величину y из этой формулы определяем как функцию (с именем h) не только аргумента x , но и номера интервала k :

```
> h:=(x,k)->(Y[k+1]-Y[k])*(x-X[k])/(X[k+1]-X[k])+Y[k];
```

Интерполирующую функцию на всем числовом промежутке задаем кусочным образом:

```
> w:=x->piecewise(x<X[2],h(x,1),x<X[3],h(x,2),
x<X[4],h(x,3),h(x,4));
```

```
  w:=x->piecewise(x<X2,h(x,1),x<X3,h(x,2),x<X4,h(x,3),h(x,4))
```

```
> simplify(w(x));
```

$$\left\{ \begin{array}{ll} 0.46667x + 0.066660 & x < 2. \\ x - 1. & x < 3. \\ 2. & x < 4. \\ -1.x + 6. & 4. \leq x \end{array} \right.$$

Итак, искомая интерполирующая функция есть $\mathbf{w}(\mathbf{x})$.
Далее – графический вывод результата интерполирования:

```
> g3:=plot(w(x), x=X[1]..X[n]):
> g2:=plot([X[i], Y[i]]$i=1..n], x= X[1]-0.5..X[n],
style=POINT, symbol=CIRCLE, color=black);
> plots[display](g3, g2);
```

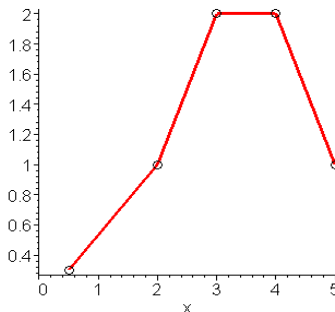


Рис. 6.2

Рассмотрим теперь случай *квадратичной* локальной интерполяции. В качестве интерполяционной функции на отрезке $[x_k, x_{k+2}]$ принимается квадратичный трехчлен. Такую интерполяцию называют также параболической. Если отвлечься от того, что отрезок $[x_k, x_{k+2}]$ является подмножеством отрезка $[x_1, x_n]$, и рассмотреть его отдельно, то квадратичную интерполяцию можно трактовать как глобальную с тремя узлами. Интерполирующую функцию на этом отрезке можно выразить явно как многочлен Лагранжа $L_2(x)$:

$$y = \frac{(x - x_{k+1})(x - x_{k+2})}{(x_k - x_{k+1})(x_k - x_{k+2})} y_k + \frac{(x - x_k)(x - x_{k+2})}{(x_{k+1} - x_k)(x_{k+1} - x_{k+2})} y_{k+1} + \frac{(x - x_k)(x - x_{k+1})}{(x_{k+2} - x_k)(x_{k+2} - x_{k+1})} y_{k+2}, \quad x_k \leq x \leq x_{k+2}.$$

Ниже приводится фрагмент программы на языке Maple, реализующий кусочно-квадратичную интерполяцию (подразумевается, что переменные **X**, **Y**, **n**, **g2**, введенные ранее, остались в оперативной памяти).

```
> h:=(x,k)->Y[k]*(x-X[k+1])*(x-X[k+2])/(X[k]-X[k+1])/(X[k]-X[k+2])+Y[k+1]*(x-X[k])*(x-X[k+2])/(X[k+1]-X[k])/(X[k+1]-X[k+2])+Y[k+2]*(x-X[k])*(x-X[k+1])/(X[k+2]-X[k])/(X[k+2]-X[k+1]);
> w:=x->piecewise(x<X[3],h(x,1),x<=X[5],h(x,3));
    w := x -> piecewise(x < X3, h(x, 1), x ≤ X5, h(x, 3))
> simplify(w(x));
    {
      0.21333 x2 - 0.066655 x + 0.28000      x < 3.
      -0.50000 x2 + 3.5000 x - 4.          x ≤ 5.
      0.                                     5. < x
    }
> g3:=plot(w(x),x=X[1]..X[5]):
> plots[display](g3,g2);
```

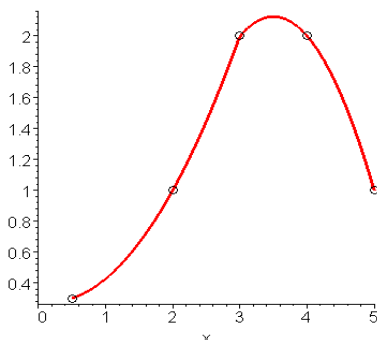


Рис. 6.3

Если присмотреться, из последнего графика можно увидеть главный недостаток данного способа кусочной интерполяции: в граничных точках отдельных участков, где осуществляется интерполирование, производная аппроксимирующей функции терпит разрыв. От этого недостатка свободны другие виды локальной интерполяции, в частности, сплайн-интерполяция и кусочная интерполяция многочленами Эрмита.

6.4. Интерполирование сплайнами

Пусть функция $f(x)$ определена на отрезке $[a, b]$ и известны её значения в системе узлов $a = x_1 < x_2 < \dots < x_n = b$. Функция $S_m(x)$ называется интерполяционным сплайном порядка m для функции $f(x)$, если выполнены следующие условия:

1) на каждом из отрезков $[x_k, x_{k+1}]$ ($k = 1, \dots, n-1$) $S_m(x)$ является полиномом степени m ;

2) на всем отрезке $[a, b]$ $S_m(x)$ имеет непрерывные производные до порядка $m-1$ включительно;

3) $S_m(x_k) = f(x_k)$, $k = 1, \dots, n$.

Если $m \geq 2$, то для единственности $S_m(x)$ следует задать дополнительно еще $m-1$ условий, которые обычно задаются на концах отрезка, либо произвольно, либо из дополнительной информации о поведении $f(x)$.

При $m = 1$ получаем известный метод ломаных – кусочно-линейную интерполяцию. Сплайны демонстрируют ряд замечательных свойств, особенно в плане равномерной сходимости и гладкости. Этим объясняется их широкое применение в различных задачах численного анализа. Чаще всего используются квадратичный сплайн $S_2(x)$ и кубический сплайн $S_3(x)$.

Рассмотрим построение кубического сплайна $S_3(x)$. Пусть на отрезке $[a, b]$ заданы значения функции $f(x)$ в узлах x_i ($i = 1, \dots, n$), $x_1 = a$, $x_n = b$. По определению, $S_3(x)$ на i -м отрезке $[x_i, x_{i+1}]$ является кубическим многочленом. Обозначим его $p_i(x)$. Будем $p_i(x)$ искать по формуле

$$p_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \quad i = 1, \dots, n-1.$$

Таким образом, для построения кубического сплайна нужно найти $4 \times (n-1)$ чисел – коэффициентов $\{a_i\}$, $\{b_i\}$, $\{c_i\}$, $\{d_i\}$. По определению $S_3(x)$ должны выполняться условия

$$p_i(x_i) = f(x_i), \quad i = 1, \dots, n-1, \quad (6.6a)$$

$$p_i(x_{i+1}) = f(x_{i+1}), \quad i = 1, \dots, n-1, \quad (6.6b)$$

$$p'_i(x_{i+1}) = p'_{i+1}(x_{i+1}), \quad i = 1, \dots, n-2, \quad (6.6c)$$

$$p''_i(x_{i+1}) = p''_{i+1}(x_{i+1}), \quad i = 1, \dots, n-2. \quad (6.6d)$$

Дополнительные два условия можно взять в виде

$$p''_1(x_1) = 0, \quad p''_{n-1}(x_n) = 0, \quad (6.7)$$

что соответствует нулевой кривизне линии в граничных точках отрезка $[a, b]$. Получаемая с такими условиями функция $S_3(x)$ называется свободным кубическим сплайном.

Итак, имеем замкнутую систему линейных алгебраических уравнений (6.6) – (6.7) относительно $\{a_i\}$, $\{b_i\}$, $\{c_i\}$, $\{d_i\}$, решение которой и даст требуемые числовые коэффициенты.

Осуществим сплайн-интерполяцию в системе Maple для функции, заданной таблично, непосредственно программируя приведенные выше формулы.

> **restart;**

> **X:=[0.5,2,3,4,5];Y:=[0.3,1,2,2,1];n:=5;**

Определим кубический многочлен $p_k(x)$ как функцию двух переменных **x** и **k** (номер частичного отрезка):

> **p:=(x,k)->a[k]+b[k]*(x-X[k])+c[k]*(x-X[k])^2+d[k]*(x-X[k])^3;**

$$p := (x, k) \rightarrow a_k + b_k (x - X_k) + c_k (x - X_k)^2 + d_k (x - X_k)^3$$

Запишем в отдельные переменные группы уравнений – последовательно соотношения (6.6a)-(6.6d):

> **eqns1:=(p(X[i],i)=Y[i])\$i=1..n-1;**

$$eqns1 := a_1 = 0.3, a_2 = 1, a_3 = 2, a_4 = 2$$

> **eqns2:=(p(X[i+1],i)=Y[i+1])\$i=1..n-1;**

$$eqns2 := a_1 + 1.5 b_1 + 2.25 c_1 + 3.375 d_1 = 1, a_2 + b_2 + c_2 + d_2 = 2, a_3 + b_3 + c_3 + d_3 = 2,$$

$$a_4 + b_4 + c_4 + d_4 = 1$$

> **eqns3:=(D[1](p)(X[i+1],i)=D[1](p)(X[i+1],i+1))\$i=1..n-2;**

$$eqns3 := b_1 + 3.0 c_1 + 6.75 d_1 = b_2, b_2 + 2 c_2 + 3 d_2 = b_3, b_3 + 2 c_3 + 3 d_3 = b_4$$

> **eqns4:=(D[1]@2)(p)(X[i+1],i)=(D[1]@2)(p)(X[i+1],i+1))\$i=1..n-2;**

$$eqns4 := 2 c_1 + 9.0 d_1 = 2 c_2, 2 c_2 + 6 d_2 = 2 c_3, 2 c_3 + 6 d_3 = 2 c_4$$

Последнюю группу составят дополнительные условия (6.7):

```
> eqns5:=(D[1]@@2)(p)(X[1],1)=0,(D[1]@@2)(p)(X[n],n-1)=0;
      eqns5:=2 c1=0, 2 c4+6 d4=0
```

Сведем воедино все указанные группы уравнений, а также неизвестные коэффициенты:

```
> eqns:={eqns1,eqns2,eqns3,eqns4,eqns5};
> var:={a[i]$i=1..n-1,b[i]$i=1..n-1,c[i]$i=1..n-1,
d[i]$i=1..n-1};
      var:={a2,a3,a4,a1,b1,c1,d1,b2,c2,d2,b3,c3,d3,b4,c4,d4}
```

Решим полученную систему:

```
> rez:=solve(eqns,var);
rez:={c1=0., a2=1., a3=2., a4=2., a1=0.3000000000, d4=0.1896713615
      d1=0.1032863850, d2=-0.3962441315, d3=0.05164319249, b4=-0.6206572770
      b1=0.2342723005, b3=0.6723004695, c3=-0.7239436620, b2=0.9314553991,
      c2=0.4647887324, c4=-0.5690140845}
```

Сплайн построим как кусочную функцию, отождествляя с каждым частичным интервалом соответствующий полином $p_k(x)$:

```
> w:=x->piecewise(x<X[2],subs(rez,p(x,1)),x<X[3],
subs(rez,p(x,2)),x<X[4],subs(rez,p(x,3)),subs(rez,p(x,4)));
w:=x->piecewise(x<X2,subs(rez,p(x,1)),x<X3,subs(rez,p(x,2)),x<X4,
subs(rez,p(x,3)),subs(rez,p(x,4)))
```

Здесь функция **subs** обеспечивает постановку найденных значений коэффициентов в $p_k(x)$.

Выведем результат в более наглядном виде:

```
> simplify(w(x)): evalf(",5);
      {
      0.16995 + 0.31174 x + 0.10329 x3 - 0.15493 x2      x < 2.
      4.1662 - 5.6826 x + 2.8423 x2 - 0.39624 x3      x < 3.
      -7.9268 + 6.4103 x - 1.1887 x2 + 0.051643 x3    x < 4.
      -16.761 + 13.036 x - 2.8451 x2 + 0.18967 x3     4. ≤ x
```

И далее – привычные графические построения

```
> g4:=plot(w(x), x=X[1]..X[n], color=black):
```

```
> g2:=plot([ [X[i],Y[i]] $i=1..n],style=POINT,
symbol=CIRCLE):
> plots[display](g2,g4);
```

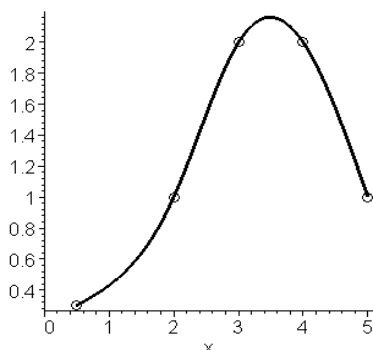


Рис. 6.4

6.5. Интерполяция Эрмита

При построении интерполяционного многочлена Эрмита требуется, чтобы в узлах совпадали с табличными данными не только его значения, но и значения его производных до некоторого порядка. Пусть даны узлы $x_1, x_2, \dots, x_n$ и соответствующие системы чисел

$$\begin{aligned} & y_1, y'_1, \dots, y_1^{(\alpha_1-1)}, \\ & y_2, y'_2, \dots, y_2^{(\alpha_2-1)}, \\ & \dots \dots \dots \\ & y_n, y'_n, \dots, y_n^{(\alpha_n-1)}. \end{aligned}$$

В теории численных методов доказывается, что может существовать только один многочлен $H_{m-1}(x)$ степени не выше $m-1$, где $m = \alpha_1 + \alpha_2 + \dots + \alpha_n$, удовлетворяющий условиям

$$H_{m-1}^{(r)}(x_k) = y_k^{(r)}, \quad r=0, 1, \dots, \alpha_k-1, \quad k=1, \dots, n.$$

Многочлен $H_{m-1}(x)$ называют многочленом Эрмита, а узел x_k — узлом интерполирования кратности α_k . Многочлен Эрмита интерполирует функцию $f(x)$, если $y_k^{(r)} = f^{(r)}(x_k)$ ($r=0, 1, \dots, \alpha_k-1, \quad k=1, \dots, n$).

В общем случае выражения для многочленов Эрмита очень громоздки, и пользоваться ими на практике трудно, разве что за исключением простейших случаев. Однако возможности современных систем компьютерной математики позволяют без особых затрат программно организовать вычисление таких многочленов. Продемонстрируем некоторые такие приемы в системе Maple.

Пример. Построить многочлен наименьшей степени, принимающий в данных точках сам и для своей производной следующие значения:

x	0.5	2	3	4	5
$f(x)$	0.3	1	2.2	2	1
$f'(x)$	0.2	0.9	1	-1.1	-0.3

> restart;

Вводим исходные данные – узлы интерполяции (массив **X**), значения в них функции (массив **Y**) и ее производной (массив **DY**):

> X:=[0.5,2,3,4,5]; Y:=[0.3,1,2.2,2,1];

DY:=[0.2,0.9,1,-1.1,-0.3];

> n:=5; m:=2*n; # n – число узлов интерполяции

> xmin:=min(X[i]\$i=1..n): xmax:=max(X[i]\$i=1..n):

Определяем аппроксимирующую функцию – алгебраический многочлен минимально возможной степени $2n-1$ (почему?), но с пока неопределенными коэффициентами **c[k]**:

> h:=x->sum(c[k]*x^(k-1),k=1..m);

$$h := x \rightarrow \sum_{k=1}^m c_k x^{(k-1)}$$

Формируем уравнения сначала из условия равенства значения полинома в узлах значениям функции $f(x)$:

> eqns1:=(h(X[i])=Y[i])\$i=1..n;

а затем то же для производных –

> eqns2:=(D(h)(X[i])=DY[i])\$i=1..n;

Решаем полученную систему относительно коэффициентов многочлена:

```
> rez:=solve({eqns1,eqns2},{c[i]$i=1..m});
{c3 = -69.446, c10 = 0.0052060, c5 = -66.153, c4 = 89.168, c2 = 27.901, c1 = -4.1172,
c9 = -0.13217, c8 = 1.4204, c6 = 30.001, c7 = -8.4136}
```

Подставив эти числа в $h(x)$, результат сохраняем в виде новой функции **hermite**:

```
> hermite:=unapply(subs(rez,h(x)),x);
hermite := x → -4.11724 + 27.9011 x - 69.4456 x2 + 89.1680 x3 - 66.1528 x4
+ 30.0007 x5 - 8.41358 x6 + 1.42042 x7 - 0.132166 x8 + 0.00520603 x9
```

Это и есть искомый многочлен Эрмита. Выведем его график:

```
> g6:=plot(hermite(x),x=xmin..xmax,color=black);
> g2:=plot([X[i],Y[i]]$i=1..n),style=POINT,symbol=CIRCLE);
> plots[display](g2,g6);
```

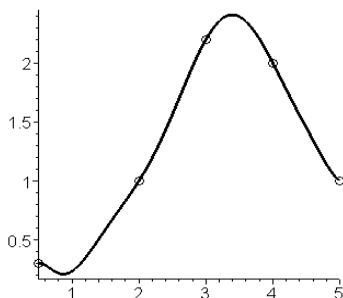


Рис. 6.5

Нетрудно видеть, что приведенный алгоритм эрмитовой интерполяции легко модифицируется на случай, если производная аппроксимируемой функции известна не во всех узлах, а лишь в некоторых, а также, если в них заданы производные более высокого порядка. Но на практике такого рода (глобальная) интерполяция Эрмита используется редко из-за слишком высокой степени полинома уже при небольшом числе узлов. Зачастую лучший результат обеспечивает локальная интерполяция, т. е. когда производится интерполирование на отдельных отрезках изменения x полиномами малой степени.

Рассмотрим локальную интерполяцию Эрмита по заданным значениям функции и её производной. На каждом отрезке

$[x_k, x_{k+1}]$ имеем по два значения функции и производной на концах отрезка, тем самым по ним можно построить полином 3-й степени

$$h(x) = a + bx + cx^2 + dx^3, \quad x_k \leq x \leq x_{k+1},$$

который должен удовлетворять условиям

$$h(x_k) = a + bx_k + cx_k^2 + dx_k^3 = f(x_k),$$

$$h(x_{k+1}) = a + bx_{k+1} + cx_{k+1}^2 + dx_{k+1}^3 = f(x_{k+1}),$$

$$h'(x_k) = b + 2cx_k + 3dx_k^2 = f'(x_k),$$

$$h'(x_{k+1}) = b + 2cx_{k+1} + 3dx_{k+1}^2 = f'(x_{k+1}).$$

Эта система уравнений замкнутая, её можно разрешить относительно четырех коэффициентов a, b, c, d и в итоге получить многочлен Эрмита на k -м отрезке. Прделав эту процедуру $n-1$ раз, найдем полную интерполирующую функцию на всём отрезке $[x_1, x_n]$.

Вот как это можно сделать в Maple.

```
> restart;
> X:=[0.5,2,3,4,5]: Y:=[0.3,1,2.2,2,1]: # табличные
данные
> DY:=[0.2,0.9,1,-1.1,-0.3]: n:=5:
> p:=array(1..n-1); # выделение памяти под массив
> for k from 1 to n-1 do # цикл по частичным отрезкам
    h:=x->a+b*x+c*x^2+d*x^3; # кубический полином
    r:=solve({h(X[k])=Y[k],h(X[k+1])=Y[k+1],
        D(h)(X[k])=DY[k],D(h)(X[k+1])=DY[k+1]},
        {a,b,c,d}); # решение системы 4 уравнений
    p[k]:=subs(r,h(x)); # запись локального многочлена
    Эрмита в массив
    od: # конец цикла (в более поздних версиях Maple вместо od
используется end do)
> w:=unapply(piecewise(x<X[2],p[1],x<X[3],p[2],
x<X[4],p[3],p[4]),x); # создание аппроксимирующей функ-
ции путем сшивки ее локальных частей
> evalf(w(x),5);# просмотр этой аппроксимирующей функции
```

$$\left\{ \begin{array}{ll} 0.20741 + 0.18889x - 0.044444x^2 + 0.074074x^3 & x < 2. \\ 6.4000 - 8.3000x + 3.8000x^2 - 0.50000x^3 & x < 3. \\ -22.400 + 18.100x - 4.2000x^2 + 0.30000x^3 & x < 4. \\ -40. + 31.700x - 7.7000x^2 + 0.60000x^3 & \text{otherwise} \end{array} \right.$$

```
> g7:=plot(w(x), x=X[1]-0.1..X[n]+0.1): # график
этой функции на отрезке [0.4, 5.1]
```

```
> g2:=plot([X[i], Y[i]]$i=1..n), style=POINT,
symbol=CIRCLE):
```

```
> plots[display](g2, g7);
```

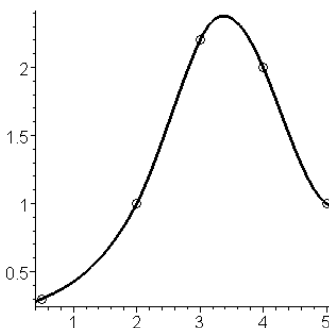


Рис. 6.6.

```
> [D(w)(0.5), D(w)(3), D(w)(4)]; # проверка значений про-
изводной в ряде узлов
[0.2000000000, 1.00000000, -1.10000000]
```

6.6. Среднеквадратическое приближение

При интерполировании основным условием является прохождение графика интерполирующей функции через данные значения функции в узлах интерполяции. Однако в ряде случаев выполнение этого условия затруднительно или даже нецелесообразно. Например, как отмечалось ранее, при большом числе узлов, когда используется глобальная интерполяция, получается высокая степень многочлена, что часто оборачивается увеличением погрешности аппроксимации вплоть до полной несостоятельности приближения. Кроме того, таблич-

ные данные могли быть получены путем измерений и содержать ошибки. Построение аппроксимирующей функции с условием обязательного её графика через эти экспериментальные точки означало бы тщательное повторение допущенных при измерениях ошибок. Выход из этого положения может быть найден выбором такой функции, график которой проходит близко от данных точек. Понятие «близко» уточняется для конкретных видов аппроксимации.

Одним из таких видов является среднеквадратическое приближение, для которого мерой отклонения аппроксимирующей функции $g(x)$ от заданной функции $f(x)$ на множестве точек (x_k, y_k) ($k = 1, 2, \dots, n$) является величина, равная сумме квадратов разностей между значениями в данных точках:

$$S = \sum_{k=1}^n [g(x_k) - y_k]^2.$$

Аппроксимирующую функцию нужно подобрать так, чтобы величина S была наименьшей. В этом состоит метод наименьших квадратов.

Пусть вид зависимости $g(x)$ однозначно определяется выбором параметров $\alpha_1, \alpha_2, \dots, \alpha_m$. Тогда величину S можно рассматривать как функцию многих независимых переменных, в роли которых выступают эти самые параметры:

$$S(\alpha_1, \alpha_2, \dots, \alpha_m) = \sum_{k=1}^n [g(x_k, \alpha_1, \alpha_2, \dots, \alpha_m) - y_k]^2. \quad (6.8)$$

Условия минимума функции $S = S(\alpha_1, \alpha_2, \dots, \alpha_m)$ требуют, чтобы её частные производные по всем переменным равнялись нулю:

$$\frac{\partial S}{\partial \alpha_1} = 0, \quad \frac{\partial S}{\partial \alpha_2} = 0, \quad \dots, \quad \frac{\partial S}{\partial \alpha_m} = 0.$$

Записанные соотношения – система уравнений для определения параметров $\alpha_1, \alpha_2, \dots, \alpha_m$.

Наиболее часто при применении метода наименьших квадратов функция $g(x)$ является линейной по неизвестным параметрам $\alpha_1, \alpha_2, \dots, \alpha_m$:

$$g(x, \alpha_1, \alpha_2, \dots, \alpha_m) = \alpha_1 \varphi_1(x) + \alpha_2 \varphi_2(x) + \dots + \alpha_m \varphi_m(x) = \sum_{i=1}^m \alpha_i \varphi_i(x).$$

В этом случае система алгебраических уравнений тоже является линейной, что, конечно же, важно при реализации вычислительного процесса. Часто в качестве такой аппроксимирующей функции используется полином степени $m-1$:

$$g(x, \alpha_1, \alpha_2, \dots, \alpha_m) = \alpha_1 + \alpha_2 x + \dots + \alpha_m x^{m-1} = \sum_{i=1}^m \alpha_i x^{i-1},$$

причем $m < n$; случай $m = n$ соответствует глобальной интерполяции. На практике стараются подобрать аппроксимирующую функцию как можно более простого вида, например многочлен степени 1, 2, 3, 4.

Пример. Аппроксимировать по методу наименьших квадратов функцию, заданную в виде таблицы, линейной и квадратичной функцией.

x	0.5	2	3	4	5
$f(x)$	0.6	1	1.4	1.6	1.55

Решение.

> **restart;**

Задаём абсциссы и ординаты известных точек, а также их количество:

> **X:=[0.5,2,3,4,5]: Y:=[0.6,1,1.4,1.6,1.55]: n:=5:**

Построим приближение квадратичной функцией: $f(x) \approx a + bx + cx^2$. Для этого вводим аппроксимирующую функцию в виде многочлена 2-й степени, имеющего три параметра a , b и c :

> **p:=a+b*x+c*x^2 # (1)**

$$p := cx^2 + bx + a$$

Ставится задача определить эти параметры. Составляем величину S по формуле (6.8):

> **S:=sum((subs(x=X['k'],p)-Y['k'])^2,'k'=1..n);**

$$S := (a + 0.5b + 0.25c - 0.6)^2 + (a + 2b + 4c - 1)^2 + (a + 3b + 9c - 1.4)^2 + (a + 4b + 16c - 1.6)^2 + (a + 5b + 25c - 1.55)^2$$

Решаем систему $\partial S/\partial a = 0$, $\partial S/\partial b = 0$, $\partial S/\partial c = 0$ относительно параметров a , b и c :

```
> r:=solve({diff(S,a)=0,diff(S,b)=0,
            diff(S,c)=0},{a,b,c}); # (2)
r := {a = 0.3399584847, b = 0.4712636222, c = -0.04392838609}
      r := {a = 0.5608606557, b = 0.2307377049}
```

Окончательно формируем аппроксимирующую функцию, подставляя в многочлен полученные значения параметров:

```
> w:=unapply(subs(r,p),x);
w := x → -0.04392838609 x2 + 0.4712636222 x + 0.3399584847
> xmin:=min(X[i]$i=1..n): xmax:=max(X[i]$i=1..n):
> d:=(xmax-xmin)/n/3.;
> g8:=plot(w(x),x=xmin-d..xmin+d,
           color=black): # построение графика и заданных точек
> g2:=plot([X[i],Y[i]]$i=1..n],style=POINT,
           symbol=CIRCLE):
> plots[display](g2,g8);
```

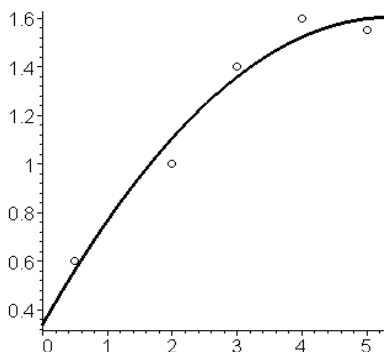


Рис. 6.7

Для аппроксимации в виде линейной функции $y = a + bx$ в приведенном блоке команд достаточно заменить строку (1) на

```
> p:=a+b*x;
```

Соответственно команда под номером (2) хотя и должна получить вид

```
> r:=solve({diff(S,a)=0,diff(S,b)=0},{a,b});
```

тем не менее, в силу избыточности может оставаться в старом варианте – результат окажется верным.

В итоге после запуска всего модифицированного блока получим искомую функцию:

$$w := x \rightarrow 0.5608606557 + 0.2307377049x$$

и её график

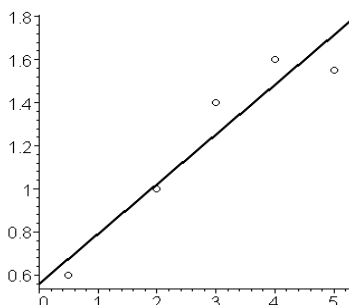


Рис. 6.8.

Можно определить реальное числовое значение величины S (суммы квадратов отклонений) для полученных аппроксимаций с помощью команды

```
> S:=sum((w(X['k'])-Y['k'])^2,'k'=1..n);
```

В первом случае это значение равно $S_1 = 0.02275$, во втором – $S_2 = 0.06847$. Если S считать величиной, характеризующей погрешность аппроксимации, то можно сделать вывод, что данная зависимость лучше аппроксимируется параболой, нежели прямой. Но в общем случае к таким утверждениям следует подходить с осторожностью. Действительно, для глобальной интерполяции мы бы вообще получили величину S , равную нулю, но как отмечалось выше, такая аппроксимация не всегда правильно отражает поведение функции. Для адекватной оценки необходимо учитывать априорные сведения о функциональной зависимости, характер эмпирических данных, их вероятностное распределение и т.д.

Рассмотрим некоторые способы среднеквадратического приближения для нелинейной относительно параметров аппроксимирующей функции. Пусть $g(x) = ae^{bx}$. Действуя по обычной схеме, получим

```
> p:=a*exp(b*x) :
> S:=sum((subs(x=X['k'],p)-Y['k'])^2,'k'=1..n);
S:=(a e(0.5 b) - 0.6)2 + (a e(2 b) - 1)2 + (a e(3 b) - 1.4)2 + (a e(4 b) - 1.6)2
+ (a e(5 b) - 1.55)2
> r:=fsolve({diff(S,a)=0,diff(S,b)=0},
{a=0..3,b=0..3}):
> w:=unapply(subs(r,p),x);
w:=x → 0.7146300309 e(0.1759098190x)
```

Обратите внимание, здесь из-за нелинейности системы приходится алгоритмы численного решения уравнений, для чего надо задавать интервалы изменения переменных, включающие решение. Для определения этих интервалов в случае двух параметров аппроксимации может помочь графическая визуализация (см. раздел 1 «Решение нелинейных уравнений»). Для трёх и более параметров нужны дополнительные соображения о локализации решения. Однако, несмотря на все усилия, случается, что метод не сходится и не дает никакого результата. Это связано с тем, что сходимость решения систем нелинейных уравнений зависит слишком от многих факторов, которые не всегда удается учесть с помощью универсального алгоритма. Так что среднеквадратическое приближение нелинейной функцией может столкнуться с серьезными трудностями вычислительного характера.

Поэтому на практике по возможности пытаются перейти к линейной аппроксимирующей зависимости. В нашем случае это достигается логарифмированием: $\ln g(x) = \ln a + bx$. Если ввести обозначения $A \equiv \ln a$, $\hat{g}(x) \equiv \ln g(x)$, получим новую функцию $\hat{g}(x) = A + bx$, аппроксимирующую табличные данные $(x_k, \ln y_k)$ ($k = 1, \dots, n$). Очевидно, эта функция уже линейна

по своим параметрам A и b . Применяв метод наименьших квадратов, найдем оптимальные значения этих параметров A^* и b^* , которые относятся к функции $\hat{g}(x)$, а исходную функцию $g(x)$ определим по формуле $g(x) = e^{A^*} \cdot e^{b^*x}$. Реализация на языке Maple (без вывода промежуточных результатов):

```
> X:=[0.5,2,3,4,5]:Y:=[0.6,1,1.4,1.6,1.55]:n:=5;
> Y1:=[ln(Y[i])$i=1..n];
> p:=A+b*x;
> S:=sum( (subs(x=X['k'],p)-Y1['k'])^2, 'k'=1..n) ;
> r:=solve({diff(S,A)=0,diff(S,b)=0},{A,b});
           r:={b=0.2210629565,A=-0.4943015392}
> w1:=unapply(exp(subs(r,p)),x);
           w1:=x -> e(-0.4943015392+0.2210629565x)
> xn:=min(X[i]$i=1..n)-0.5: xx:=max(X[i]$i=1..n)+0.3:
> g8:=plot(w1(x),x=xn..xx,color=black,thickness=3):
> g2:=plot([X[i],Y[i]]$i=1..n],style=POINT,
symbol=BOX,color=black):
> plots[display](g2,g8);
```

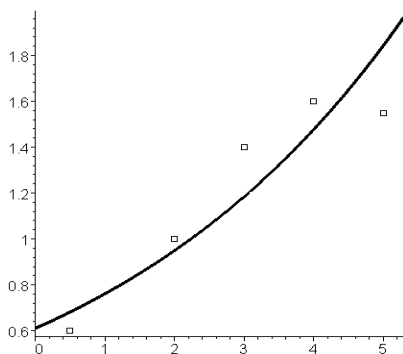


Рис. 6.9

Как видно из итоговых аналитических зависимостей, две последние аппроксимации с одной и той же функциональной зависимостью $g(x) = ae^{bx}$, но полученные разными способами, все же отличаются. Подумайте, почему?

6.7. Аппроксимация с помощью взвешенных невязок

Существует довольно общий подход к аппроксимации функций, называемый методом взвешенных невязок. В своем обычном варианте он, прежде всего, предназначен для получения непрерывных аппроксимаций. Тем не менее, и многие рассмотренные выше методы точечной аппроксимации являются просто частными случаями этого общего подхода. Суть его в следующем.

Пусть требуется построить аппроксимацию функции $f(x)$ на отрезке $[a, b]$. Аппроксимирующая функция $g(x)$ берется в виде линейной комбинации

$$f(x) \approx g(x) = \sum_{k=1}^M g_k N_k(x), \quad (6.9)$$

где $\{g_k\}$ – некоторые параметры, вычисляемые таким образом, чтобы получить хорошее приближение, $\{N_k, k=1, 2, 3, \dots\}$ – полная система линейно независимых *базисных* функций. Условие полноты этой системы означает, что она способна приближать $g(x)$ к $f(x)$ с любой наперед заданной точностью при $M \rightarrow \infty$.

Вводится функция R – погрешность аппроксимации или *невязка*, определяемая в каждой точке промежутка $[a, b]$ как разность f и g :

$$R(x) = f(x) - g(x) = f(x) - \sum_{k=1}^M g_k N_k(x), \quad a \leq x \leq b. \quad (6.10)$$

Понятно, что для хорошего приближения невязка должна как можно меньше отличаться от нуля, и задача формулируется следующим образом: подобрать коэффициенты из (6.9) так, чтобы $R(x) \approx 0$ для всех $x \in [a, b]$. Чтобы уменьшить эту невязку (по модулю) на всем указанном промежутке, потребуем равенства нулю соответствующего числа интегралов от невязки, взятых с различными весами, т.е.

$$\int_a^b W_i (f - g) dx = \int_a^b W_i R dx = 0, \quad i=1, 2, \dots, M, \quad (6.11)$$

где $\{W_i, i=1, 2, 3, \dots\}$ – множество линейно независимых *весовых* функций. Тогда общее требование сходимости $R \rightarrow 0$ при $M \rightarrow \infty$ можно записать в форме условия выполнения равенств (6.11) для всех i при $M \rightarrow \infty$. В функциональном анализе показывается, что это действительно будет верно лишь при условии $R \rightarrow 0$ во всех точках отрезка $[a, b]$.

С учетом выражения (6.10) система уравнений метода взвешенных невязок сводится к системе линейных алгебраических уравнений для неизвестных коэффициентов

$$\sum_{k=1}^M g_k \int_a^b W_i N_k dx = \int_a^b W_i f dx; \quad i=1, 2, \dots, M, \quad (6.12)$$

или в общем виде –

$$Sg = f, \quad (6.12^*)$$

где

$$g^T = (g_1, g_2, \dots, g_M); \quad S_{ik} = \int_a^b W_i N_k dx; \quad f_i = \int_a^b W_i f dx; \quad 1 \leq i, k \leq M.$$

Таким образом, если известна аппроксимируемая функция f и выбраны подходящие системы базисных и весовых функций, то, решая систему (6.12), можно получить коэффициенты $\{g_k\}$ в аппроксимации (6.9).

На практике могут быть использованы различные системы весовых функций $\{W_i, i=1, 2, \dots\}$, ведущие к разным частным методам аппроксимации посредством взвешенных невязок. Ниже приводятся некоторые из наиболее употребительных таких систем.

1) Поточечная коллокация. Весовые функции заданы формулой

$$W_i = \delta(x - x_i),$$

где $\delta(x - x_i)$ – дельта-функция Дирака, по определению обладающая свойствами:

$$\delta(x - x_i) = 0 \quad \text{при } x \neq x_i; \quad \delta(x - x_i) = \infty \quad \text{при } x = x_i;$$

$$\int_{x < x_i}^{x > x_i} G(x) \delta(x - x_i) dx = G(x_i).$$

Такой выбор весовых функций эквивалентен тому, что невязка полагается равной нулю (т.е. $f=g$) в ряде фиксированных точек $\{x_i\}$. А последнее требование означает, что в данном случае реализуется интерполяция.

2) Коллокация по отрезкам. Пусть на отрезке $[a, b]$ введена система точек $\{x_i\}$: $x_1=a < x_2 < x_3 < \dots < x_M < x_{M+1}=b$. Если весовые функции определены по правилу

$$W_i = \begin{cases} 1, & \text{если } x_i \leq x \leq x_{i+1}, \\ 0, & \text{если } x < x_i \text{ или } x > x_{i+1}, \end{cases}$$

то уравнения метода взвешенных невязок равносильны требованию равенства нулю интегралов от невязки по каждому из M промежутков $[x_i, x_{i+1}]$. Другими словами, площади под кривыми $y=f(x)$ и $y=g(x)$ на каждом таком промежутке должны быть равны. Матрица S и вектор f в (6.12*) имеют элементы

$$S_{ik} = \int_{x_i}^{x_{i+1}} N_k(x) dx; \quad f_i = \int_{x_i}^{x_{i+1}} f(x) dx.$$

3) Метод Галеркина. В этом наиболее популярном варианте метода взвешенных невязок в качестве весовых функций используются сами базисные функции, т.е. $W_i=N_i$. Матричные элементы системы метода взвешенных невязок и элементы её вектора правых частей имеют вид

$$S_{ik} = \int_a^b N_i(x) N_k(x) dx; \quad f_i = \int_a^b f(x) N_i(x) dx.$$

Отметим, что здесь матрица S симметричная и положительно определенная, что дает этому методу дополнительные вычислительные преимущества.

Особая простота уравнений метода Галеркина возникает, когда функции $\{N_k, k=1, 2, \dots\}$ на $[a, b]$ образуют ортогональную систему, т.е. выполняется соотношение

$$\int_a^b N_i N_k dx = \delta_{ik} d, \quad d \neq 0,$$

(δ_{ik} – символ Кронекера; $\delta_{ik}=0$ при $i \neq k$ и $\delta_{ik}=1$ при $i=k$). Например, рассматривая аппроксимацию на отрезке $[0, l]$ базисными функциями $\{N_k = \sin \frac{k\pi x}{l}, k=1, 2, \dots\}$, получим для коэффициентов системы:

$$S_{ik} = \int_0^l \sin \frac{i\pi x}{l} \sin \frac{k\pi x}{l} dx = \begin{cases} l/2, & \text{если } i = k, \\ 0, & \text{если } i \neq k; \end{cases}$$

$$f_i = \int_0^l f(x) \sin \frac{i\pi x}{l} dx.$$

Следовательно, матрица **S** системы (6.12*) имеет диагональную форму, что сразу позволяет записать решение

$$g_k = \frac{2}{l} \int_0^l f(x) \sin \frac{k\pi x}{l} dx, \quad k=1, 2, \dots, M.$$

Полученное выражение тождественно формуле для коэффициентов разложения функций по синусам кратных дуг из теории рядов Фурье. Вообще говоря, известные из математики ортогональные разложения можно рассматривать как аппроксимацию по Галёркину с помощью взвешенных невязок при $M \rightarrow \infty$.

4) Метод моментов. Один из очевидных способов выбора весовых функций состоит в использовании системы $\{W_i = x^{i-1}, i=1, 2, 3, \dots\}$. Тогда метод взвешенных невязок приводит в требование, чтобы площадь под кривой невязки R и её различные моменты относительно начала координат на отрезке $[a, b]$ были равны нулю, т.е.

$$\int_a^b R dx = 0, \quad \int_a^b x R dx = 0, \quad \int_a^b x^2 R dx = 0, \quad \dots, \quad \int_a^b x^{M-1} R dx = 0.$$

Коэффициенты системы уравнений метода выражаются через интегралы

$$S_{ik} = \int_a^b x^{i-1} N_k(x) dx; \quad f_i = \int_a^b x^{i-1} f(x) dx.$$

Что касается термина “момент”, то помимо известных величин из механики (например, момент силы), можно также проследить аналогию с теорией вероятностей, где вводится понятие начального момента k -го порядка случайной величины X :

$$\alpha_k = \int_{-\infty}^{\infty} x^k f_X(x) dx.$$

Замечания. 1) В ряде случаев целесообразно вместо (6.9) брать аппроксимацию вида

$$f(x) \approx g(x) = \psi(x) + \sum_{k=1}^M g_k N_k(x), \quad (6.13)$$

где $\psi(x)$ – любая гладкая функция, принимающая на концах отрезка $[a, b]$ те же значения, что и $f(x)$, т.е. $\psi(a)=f(a)$, $\psi(b)=f(b)$; при этом базисные функции $\{N_k\}$ должны равняться нулю при $x=a$ и $x=b$. Тогда в системе уравнений метода взвешенных невязок (6.12) изменится вычисление только элементов вектора правых частей, а именно

$$f_i = \int_a^b W_i(f - \psi) dx, \quad i = 1, 2, \dots, M.$$

В качестве ψ обычно берут линейную функцию

$$\psi(x) = [f(b) - f(a)](x - a)(b - a)^{-1} + f(a). \quad (6.14)$$

Данный способ аппроксимации позволяет лучше учесть поведение функции вблизи границ отрезка. К примеру, предположим, что на $[a, b]$ производится разложение с помощью усеченных синус-рядов Фурье

$$f(x) \approx g(x) = \sum_{k=1}^M g_k \sin \frac{\pi k(x - a)}{b - a}.$$

Заметив, что все базисные функции, участвующие в этом разложении, обращаются в ноль при $x=a$ и $x=b$, получаем $g(a)=g(b)=0$. При этом сама аппроксимируемая функция $f(x)$ в этих точках может быть и не равной нулю. Тогда около значений a и b будем иметь заметную ошибку аппроксимации. Однако, выбрав аппроксимацию в виде

$$f(x) \approx g(x) = \frac{f(b) - f(a)}{b - a}(x - a) + f(a) + \sum_{k=1}^M g_k \sin \frac{\pi k(x - a)}{b - a},$$

обеспечим и $g(a) = f(a)$, и $g(b) = f(b)$, и, тем самым, лучшее соответствие вблизи граничных точек.

2) Метод наименьших квадратов, если его применить к случаю интегральной аппроксимации, приводит к тем же уравнениям, что и метод Галёркина.

В заключение рассмотрим примеры решения задач на основе метода взвешенных невязок в системе Maple.

Пример. Аппроксимировать функцию $f(x) = \int_0^x e^{-2 \sin t} dt$

на промежутке $[0, \pi]$.

Решение. Выбор весовых функций осуществим по Галеркину, число параметров возьмем $M=4$. Составляем программу:

> **restart;**

> **f:=x->int(exp(-2*sin(t)), t=0..x);**

$$f := x \rightarrow \int_0^x e^{(-2 \sin(t))} dt$$

> **N:=[1,x,x^2,x^3,x^4,x^5];** # базисные функции

> **W:=N;** # весовые функции как в методе Галеркина

> **m:=4;** # число параметров аппроксимации

> **g:=x->sum(c[k]*N[k], k=1..m);** # аппроксимация в виде суммы (4.9)

$$g := x \rightarrow \sum_{k=1}^m c_k N_k$$

> **a:=0: b:=Pi: diap:=x=a..b;** # диапазон изменения переменной x

> **for i from 1 to m do**

eqns[i]:=(int(g(x)*W[i],diap)=evalf(Int(f(x)*W[i],diap)));

od; # формирование системы уравнений метода взвешенных невязок

$$\begin{aligned} eqns_1 &:= c_1 \pi + \frac{1}{2} c_2 \pi^2 + \frac{1}{3} c_3 \pi^3 + \frac{1}{4} c_4 \pi^4 = 1.688450194 \\ eqns_2 &:= \frac{1}{5} c_4 \pi^5 + \frac{1}{4} c_3 \pi^4 + \frac{1}{3} c_2 \pi^3 + \frac{1}{2} c_1 \pi^2 = 3.273543417 \\ eqns_3 &:= \frac{1}{6} c_4 \pi^6 + \frac{1}{5} c_3 \pi^5 + \frac{1}{4} c_2 \pi^4 + \frac{1}{3} c_1 \pi^3 = 7.506750706 \\ eqns_4 &:= \frac{1}{7} c_4 \pi^7 + \frac{1}{6} c_3 \pi^6 + \frac{1}{5} c_2 \pi^5 + \frac{1}{4} c_1 \pi^4 = 18.70167800 \end{aligned}$$

```
> r:=solve({eqns[j]$j=1..m},{c[j]$j=1..m}); # решение системы
{c4=0.087577824, c3=-0.41270077, c2=0.75908203, c1=0.023952682}
> plot([subs(r,g(x)),f(x)],x=a..b); # графики аппроксимирующей и исходной функций
```

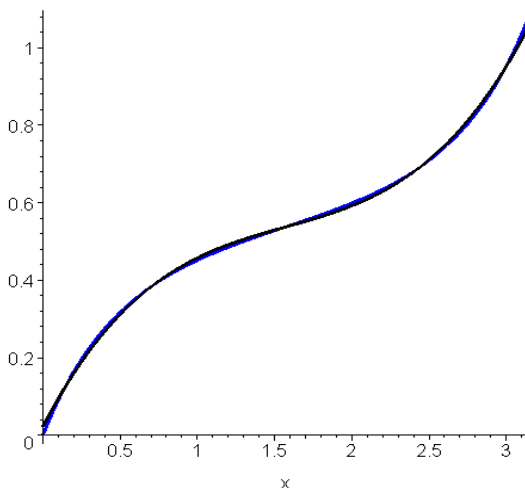


Рис. 6.10

Оценить точность аппроксимации можно с помощью нормы невязки, определяемой по формуле

$$\delta = \left(\int_a^b [g(x) - f(x)]^2 dx \right)^{1/2}.$$

Чем меньше эта величина, тем лучше аппроксимация. Её вычисление в Maple оформляем в виде строки:

```
> delta:=evalf(int((subs(r,g(x))-f(x))^2,diap))^0.5;
      δ := 0.01199774484
```

Если в представленном тексте программы поменять число параметров m на 5, то величина δ не изменится, но с шестью параметрами она резко уменьшается: $\delta=0.000724$. Это свидетельствует о сходимости метода.

Может возникнуть вопрос: зачем нужен метод взвешенных невязок, если можно функцию $f(x)$ разложить в степенной ряд, и его частичную сумму использовать в качестве аппроксимирующей функции? Для сравнения выведем первые четыре члена разложения нашей функции в ряд в окрестности точки $x = \pi/2$ (середина отрезка).

```
> h:=convert(series(1.*f(x),x=Pi/2,4),polynom);
      h := 0.537450389+0.1353352832x - 0.06766764160π + 0.04511176106⎛x -  $\frac{\pi}{2}$ ⎞3
```

```
> plot([h,f(x)],diap,thickness=3);
```

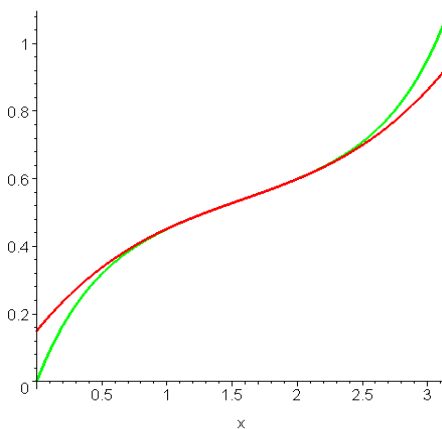


Рис. 6.11

Хорошо видно, что приближение тем хуже, чем дальше от точки разложения $x = \pi/2$. Отсюда вывод: аппроксимация методом взвешенных невязок вносит более равномерную ошибку на заданном промежутке и, как правило, дает лучший результат, нежели с помощью разложения в степенной ряд (при одинаковой степени аппроксимирующего полинома).

С помощью приведенной программы можно провести все варианты метода взвешенных невязок. Для этого достаточно заменить систему весовых функций (массив **W**) и, возможно, базисные функции (массив **N**). Например, для поточечной коллокации следует ввести

```
> a:=0;b:=Pi; # начало и конец промежутка изменения x
> W:=[Dirac(x-('k'-1)*(b-a)/('m'-1))$'k'=1..m];
# дельта-функции Дирака
```

И ещё, для того чтобы Maple корректно вычислял интегралы с дельта-функциями Дирака, необходимо расширить пределы интегрирования, например, задать

```
> diap:=x=a-1..b+1;
```

Коллокацию по отрезкам получим, если задать

```
> a:=0;b:=Pi;
> W:=[piecewise(x>a+('k'-1)*(b-a)/m and
x<a+'k'*(b-a)/m,1,0)$'k'=1..m]; # единичные запаздывающие импульсы
```

Пример. Получить аппроксимацию решения задачи Коши для дифференциального уравнения $(x^2 y')' + (x^2 - 1)y = 0$, $y(0.5) = 0$, $y'(0.5) = 2.5$ на отрезке $[0.5, 5]$.

Решение. Попробуем решить это уравнение аналитически:

```
> restart;
> p:=dsolve({diff(x^2*diff(y(x),x),x)+(x^2-1)*y(x)=0,y(0.5)=0.,D(y)(0.5)=2.5},y(x));
```

$$p := y(x) = -\frac{5}{4}$$

$$\frac{\sqrt{2} \operatorname{BesselY}\left(\frac{\sqrt{5}}{2}, \frac{1}{2}\right) \operatorname{BesselJ}\left(\frac{\sqrt{5}}{2}, x\right)}{\left(\operatorname{BesselY}\left(\frac{\sqrt{5}}{2}, \frac{1}{2}\right) \operatorname{BesselJ}\left(\frac{\sqrt{5}}{2} + 1, \frac{1}{2}\right) - \operatorname{BesselY}\left(\frac{\sqrt{5}}{2} + 1, \frac{1}{2}\right) \operatorname{BesselJ}\left(\frac{\sqrt{5}}{2}, \frac{1}{2}\right)\right) \sqrt{x}} + \frac{5}{4}$$

$$\frac{\sqrt{2} \operatorname{BesselJ}\left(\frac{\sqrt{5}}{2}, \frac{1}{2}\right) \operatorname{BesselY}\left(\frac{\sqrt{5}}{2}, x\right)}{\left(\operatorname{BesselY}\left(\frac{\sqrt{5}}{2}, \frac{1}{2}\right) \operatorname{BesselJ}\left(\frac{\sqrt{5}}{2} + 1, \frac{1}{2}\right) - \operatorname{BesselY}\left(\frac{\sqrt{5}}{2} + 1, \frac{1}{2}\right) \operatorname{BesselJ}\left(\frac{\sqrt{5}}{2}, \frac{1}{2}\right)\right) \sqrt{x}}$$

Maple справился с этой задачей, но решение выдал в виде громоздкого выражения через специальные функции (Бесселя). Работать с таким выражением крайне неудобно. Вот здесь и приходит на помощь аппроксимация. Сначала введем функцию для хранения точного решения

```
> u:=unapply(subs(p,y(x)),x);
```

Дальнейшие строки программы уже встречались и почти не требуют комментария.

```
> a:=0.5: b:=5: m:=4: # т.е. взято четыре параметра
```

```
> N:=[1,x,x^2,x^3,x^4,x^5,x^6]: W:=N: # весовые функции совпадают с базисными
```

```
> g:=x->sum(c[k]*N[k],k=1..m): # стандартная аппроксимация
```

```
> for i from 1 to m do
```

```
eqns[i]:=(int(g(x)*W[i],x=a..b)=
```

```
evalf(Int(u(x)*W[i],x=a..b)));od;
```

```
> r:=solve({eqns[j]$j=1..m},{c[j]$j=1..m});
```

```
> subs(r,g(x));
```

$$-0.662168 + 1.92305x - 0.712318x^2 + 0.0688428x^3$$

Этот кубический полином и есть искомая функция, аппроксимирующая решение заданного дифференциального уравнения с учетом начальных условий. Несомненно, она гораздо проще для восприятия и выполнения действий, чем исходная функция. Оценим точность приближения сначала визуально с помощью команды

```
> plot([subs(r,g(x)),u(x)],x=a..b,thickness=2);
```

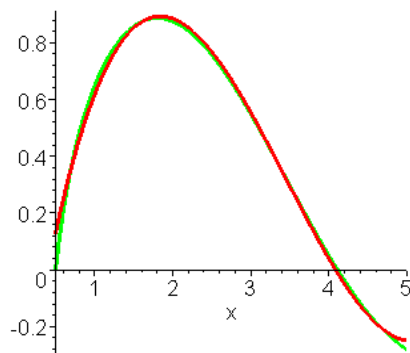


Рис. 6.12

Графики почти совпадают. Погрешность аппроксимации:

```
> delta:=evalf(Int((subs(r,g(x))-
u(x))^2,x=a..b,method=_CCquad)^0.5;
```

$$\delta := 0.03904897769$$

Если метод сходится, то теоретически, увеличивая число параметров M (в программе это переменная m), должны получать более точное приближение, а значит, уменьшающуюся величину δ . Составим таблицу.

M	Аппроксимирующая функция $g(x)$	Погрешность δ
4	$-0.6622+1.923x-0.7123x^2+0.0688x^3$	0.03905
5	$-0.8572+2.359x-1.014x^2+0.1496x^3-0.0073x^4$	0.02450
6	$-1.064+2.955x-1.600x^2+0.4037x^3-0.0573x^4+0.0036x^5$	0.01765
7	$-1.392+4.112x-3.084x^2+1.311x^3-0.342x^4+0.048x^5-0.0027x^6$	0.00979

Погрешность аппроксимации действительно убывает, метод сходится.

Скорость сходимости и точность метода взвешенных невязок могут существенно зависеть от ряда факторов, в частности, от гладкости самой функций и её производных, от того или иного выбора весовых и базисных функций, от учета граничных условий (см. замечание).

Взяв в рассматриваемом примере аппроксимацию, удовлетворяющую граничным условиям на концах промежутка $[a, b]$,

```
> g:=(u(b)-u(a))*(x-a)/(b-a)+u(a)+
sum(c[k]*N[k]*(x-a)*(x-b),k=1..m);
```

убедитесь, что при одном и том же числе M результат становится точнее.

Разобранный пример касался случая, когда Maple был способен найти аналитическое решение дифференциального уравнения. Но что делать, когда такого решения нет, и результат выдается в численном виде? Оказывается, часто удаётся построить аппроксимацию методом взвешенных невязок и в этом случае.

Пример. Аппроксимировать решение задачи Коши $y' + y = \sin(xy)$, $y(0) = 0.5$ при $0 \leq x \leq 4$.

Решение.

```
> restart;
```

Решение ищем в численном варианте (с параметром `type=numeric`):

```
> p:=dsolve({D(y)(x)+y(x)=sin(x*y(x)), y(0)=0.5},  
y(x), type=numeric);
```

```
p := proc(x_rkf45) ... end proc
```

Решение хранится в переменной **p**, но его формат весьма необычный. Это видно из примера вывода значения функции в некоторой точке:

```
> p(2.3);
```

```
[x = 2.3, y(x) = 0.568875854698128180]
```

В таком представлении эта функция не может быть использована в дальнейших вычислениях (в частности под знаком интеграла). Поэтому её необходимо переопределить, например, с помощью процедуры

```
> u:=proc(t) global i; local w;  
w:=subs(p(t)[2], y(x)); RETURN(w) end;  
u:=proc(t) local w; global i; w:=subs(p(t)[2], y(x)); RETURN(w) end proc
```

Чтобы проверить, снова вычислим:

```
> u(2.3)
```

```
0.5688758547
```

Цель достигнута. Теперь реализация самого метода:

```
> m:=5: a:=0: b:=4.:
```

Выбираем системы базисных и весовых функций

```
> N:=x->[sin(Pi*'k'*(x-a)/(b-a))$'k'=1..m];
```

```
> W:=x->[1, x, x^2, x^3, x^4, x^5, x^6];
```

Обратите внимание, здесь в отличие от предыдущих примеров массивы **M** и **N** введены как вектор-функции. Строим аппроксимацию с совпадением граничных значений (см. (6.13)–(6.14)):

```
> g:=x->(u(b)-u(a))*(x-a)/(b-a)+u(a)+  
sum(c[k]*N(x)[k], k=1..m);
```

```

> for i from 1 to m do
    eqns[i]:=int(g(x)*W(x)[i],x=a..b)=
    evalf(Int(W(t)[i]*'u'(t),a..b));
od; # система уравнений метода
> r:=solve({eqns[j]$j=1..m},{c[j]$j=1..m}); # решение системы
> evalf[6](subs(r,g(x))); # получена аппроксимирующая функция
0.0373732x+0.500000-0.0949366sin(0.785398x)-0.186125sin(1.57080x)+
0.00722236sin(2.35619x)-0.00207420sin(3.14159x)-0.0207444sin(3.92699x)
> gr1:=plots[odeplot](p,a..b): # графики функций
> gr2:=plot([subs(r,g(x))],x=a..b,color=blue,
thickness=2):
> plots[display](gr1,gr2);

```

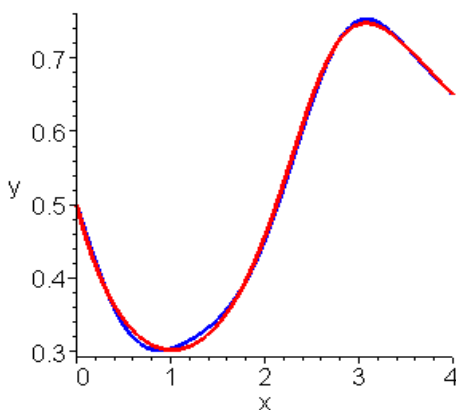


Рис. 6.13.

Соответствие имеет место. Для количественной оценки погрешности вычисляем:

```

> delta:=evalf(Int((subs(r,g(t))-'u'(t))^2,
a..b,method=_Gquad))^0.5;
delta:=0.009961375

```

Проверку сходимости проведите самостоятельно.

Упражнения

1. По имеющимся данным провести глобальное интерполирование многочленом минимальной степени; локальное интерполирование полиномами 2-й степени, а также линейным и кубическим сплайном. Найти значение функции в точке x , используя полученные результаты. Построить графическую иллюстрацию интерполирования.

Варианты 1, 11, 21, 31.

x	0,43	0,48	0,55	0,62	0,70	0,75
$f(x)$	1,6359	1,7323	1,8768	2,0304	2,2284	2,3597

В точке $x = 0,702$; $x = 0,512$; $x = 0,645$; $x = 0,608$.

Варианты 2, 12, 22, 32.

x	0,02	0,08	0,12	0,17	0,23	0,30
$f(x)$	1,0231	1,0959	1,1472	1,2148	1,3012	1,4097

В точке $x = 0,203$; $x = 0,102$; $x = 0,114$; $x = 0,285$.

Варианты 3, 13, 23, 33.

x	0,35	0,41	0,47	0,51	0,56	0,64
$f(x)$	2,7395	2,3008	1,9686	1,7877	1,5950	1,3431

В точке $x = 0,482$; $x = 0,436$; $x = 0,552$; $x = 0,526$.

Варианты 4, 14, 24, 34.

x	0,41	0,46	0,52	0,60	0,65	0,72
$f(x)$	2,5741	2,3251	2,0933	1,8620	1,7492	1,6209

В точке $x = 0,616$; $x = 0,487$; $x = 0,665$; $x = 0,537$.

Варианты 5, 15, 25, 35.

x	0,68	0,73	0,80	0,88	0,93	0,99
$f(x)$	0,8086	0,8949	1,0296	1,2096	1,3408	1,5236

В точке $x = 0,774$; $x = 0,896$; $x = 0,715$; $x = 0,955$.

Варианты 6, 16, 26, 36.

x	0,11	0,15	0,21	0,29	0,35	0,40
$f(x)$	9,0542	6,6165	4,6917	3,3510	2,7395	2,3652

В точке $x = 0,275$; $x = 0,314$; $x = 0,235$; $x = 0,332$.

Варианты 7, 17, 27, 37.

x	0,05	0,10	0,17	0,25	0,30	0,36
$f(x)$	0,0500	0,1003	0,1716	0,2553	0,3093	0,3764

В точке $x = 0,263$; $x = 0,143$; $x = 0,325$; $x = 0,085$.

Варианты 8, 18, 28, 38.

x	0,51	0,55	0,61	0,69	0,75	0,80	0,84
$f(x)$	9,245	6,561	4,719	3,015	2,597	2,253	2,004

В точке $x = 0,749$; $x = 0,585$; $x = 0,67$; $x = 0,53$.

Варианты 9, 19, 29, 39

x	0,115	0,159	0,218	0,294	0,353	0,408
$f(x)$	5,2542	4,2150	3,8227	2,4421	1,6385	0,2563

В точке $x = 0,256$; $x = 0,3$; $x = 0,188$; $x = 0,36$.

Варианты 10, 20, 30, 40.

x	2,0	2,3	2,5	3,0	3,5	3,8	4,0
$f(x)$	5,848	6,127	6,300	6,694	7,047	7,243	7,368

В точке $x = 2,22$; $x = 2,78$; $x = 3,34$; $x = 3,75$.

2. Набор экспериментальных данных значений x_i и y_i имеет вид таблицы:

1	x_i	1,20	1,57	1,94	2,31	2,68	3,05	3,42	3,79
	y_i	2,59	2,06	1,58	1,25	0,91	0,66	0,38	0,21
2	x_i	1,73	2,56	3,39	4,22	5,05	5,89	6,70	7,53
	y_i	0,63	1,11	1,42	1,94	2,30	2,89	3,29	3,87
3	x_i	-4,38	-3,84	-3,23	-2,76	-2,22	-1,67	-1,13	-0,60
	y_i	2,25	2,83	3,44	4,31	5,29	6,55	8,01	10,04
4	x_i	1,00	1,64	2,28	2,91	3,56	4,19	4,84	5,48
	y_i	0,28	0,19	0,15	0,11	0,09	0,08	0,07	0,06
5	x_i	5,84	3,82	6,19	9,22	7,87	6,29	4,43	8,91
	y_i	79,31	57,43	60,66	92,55	90,12	71,30	70,50	91,25
6	x_i	2,91	2,94	6,35	6,58	3,80	6,43	0,57	5,96
	y_i	82,16	61,02	44,56	82,52	99,17	70,24	63,23	66,48

7	x_i	5,46	2,73	6,49	4,26	2,39	6,46	0,86	2,05
	y_i	65,72	58,05	60,05	55,79	50,83	47,69	44,49	59,74
8	x_i	1,28	1,76	2,24	2,72	3,20	3,68	4,16	4,64
	y_i	2,10	2,62	3,21	3,96	4,98	6,06	7,47	9,25
9	x_i	-4,84	-4,30	-3,76	-3,22	-2,68	-2,14	-1,60	-1,06
	y_i	-0,09	-0,11	-0,13	-0,16	-0,19	-0,26	-0,39	-0,81
10	x_i	3,54	4,29	4,78	3,99	1,13	6,29	1,89	3,27
	y_i	22,81	28,42	24,95	26,96	8,78	33,55	15,77	22,89
11	x_i	4,08	4,42	2,52	-0,08	2,14	3,36	7,35	5,00
	y_i	18,31	21,85	16,93	8,23	10,90	17,18	36,45	24,11
12	x_i	1,16	1,88	2,60	3,32	4,04	4,76	5,48	6,20
	y_i	0,18	0,26	0,32	0,36	0,40	0,43	0,95	0,85
13	x_i	1,00	1,71	2,42	3,13	3,84	4,55	5,26	5,97
	y_i	12,49	4,76	2,55	1,60	1,11	0,82	0,63	0,50
14	x_i	-0,64	-0,36	-0,08	0,20	0,48	0,76	1,04	1,32
	y_i	29,51	18,86	12,05	7,70	4,92	3,14	2,01	1,28
15	x_i	-2,45	-1,94	-1,43	-0,92	-0,41	0,10	0,61	1,12
	y_i	0,87	1,19	1,68	2,23	3,04	4,15	5,66	7,72
16	x_i	1,54	1,91	2,28	2,65	3,02	3,39	3,76	4,13
	y_i	-2,52	-3,08	-3,54	-3,93	-4,27	-4,57	-4,84	-5,09
17	x_i	1,20	2,00	2,80	3,60	4,40	5,20	6,00	6,80
	y_i	-10,85	-6,15	-4,14	-3,02	-2,30	-1,81	-1,45	-1,17
18	x_i	-1,04	-0,67	-0,30	0,07	0,44	0,81	1,18	1,55
	y_i	10,80	8,08	5,97	4,44	3,31	2,46	1,83	1,36
19	x_i	0,41	0,97	1,53	2,09	2,65	3,21	3,77	4,33
	y_i	0,45	1,17	1,56	1,82	2,02	2,18	2,31	2,44
20	x_i	1	2	5	8	9	12	14	16
	y_i	6	7,45	8,24	12,46	13,09	14,56	25,89	29,91
21	x_i	0	2	4	5	8	10	12	15
	y_i	29,8	22,9	17,1	15,16	10,7	10,2	10,1	15,2

22	x_i	1,65	1,39	1,19	1,02	0,82	0,75	0,66	0,89
	y_i	1,034	1,232	1,432	1,752	2,056	2,37	2,76	3,198
23	x_i	-0,22	-3,05	-1,76	-1,25	-0,45	-0,80	-0,26	-3,07
	y_i	58,46	36,05	31,17	16,17	11,16	69,23	58,08	43,13
24	x_i	6,35	6,58	3,80	6,43	0,57	5,96	3,40	4,55
	y_i	44,56	82,52	99,17	70,2	63,23	66,48	48,35	40,24
25	x_i	2,95	2,60	2,69	3,01	2,44	2,51	3,37	2,98
	y_i	113,8	119,6	106,2	120	107,4	114,8	115,5	117,4
26	x_i	4,23	1,40	4,07	2,93	3,44	1,09	1,82	2,43
	y_i	65,72	58,05	60,05	55,79	50,83	47,69	44,49	59,74
27	x_i	4,07	2,93	3,44	1,09	1,82	2,43	3,85	0,97
	y_i	60,05	55,79	50,83	47,69	44,49	59,74	56,81	45,82
28	x_i	8,90	9,22	8,74	8,98	8,77	9,31	8,81	9,14
	y_i	105,2	85,02	52,76	56,86	72,19	61,09	70,44	51,67
29	x_i	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8
	y_i	0,095	0,182	0,262	0,337	0,406	0,470	0,588	0,642
30	x_i	2,51	3,74	8,70	5,36	1,89	3,01	3,59	2,64
	y_i	55,65	67,68	105,2	85,02	52,76	56,86	72,19	61,09

Найти методом наименьших квадратов аппроксимирующие зависимости в виде линейных комбинаций функций следующих систем:

а) $1, x, x^2, x^3, x^4, \dots$

(т.е. в виде $g(x) = \alpha_1 + \alpha_2 x$, $g(x) = \alpha_1 + \alpha_2 x + \alpha_3 x^2$, $g(x) = \alpha_1 + \alpha_2 x + \alpha_3 x^2 + \alpha_4 x^3$ и т.д.);

б) $1, \cos \frac{2\pi x}{l}, \sin \frac{2\pi x}{l}, \cos \frac{4\pi x}{l}, \sin \frac{4\pi x}{l}, \dots$ (здесь $l = b - a$;
 $b = \max_i x_i$; $a = \min_i x_i$);

в) $1, e^x, e^{2x}, e^{3x}, e^{4x}, \dots$;

г) $1, \operatorname{sh} x, \operatorname{ch} x, \operatorname{sh} 2x, \operatorname{ch} 2x, \dots$;

д) $1, x, \sin \frac{\pi(x-a)}{l}, \sin \frac{2\pi(x-a)}{l}, \sin \frac{3\pi(x-a)}{l}, \dots$ (здесь $a = \min_i x_i$);

е) многочлены Чебышева $\{T_n(x) = \cos(n \arccos x), n=0, 1, 2, \dots\}$;
 $T_0(x)=1, T_1(x)=x, T_2(x)=2x^2-1, T_3(x)=4x^3-3x, T_4(x)=1+8x^4-8x^2$.

ж) $1, (x-a)^2, (x-a)^4, (x-a)^6, (x-a)^8, \dots$ (здесь $a = \min_i x_i$);

з) $1, x-\delta, \frac{1}{x-\delta}, (x-\delta)^2, \frac{1}{(x-\delta)^2}, \dots$ ($\delta < \min_i x_i$).

Число слагаемых в линейных комбинациях использовать от 2 до 5 в случае а) и от 3 до 5 в остальных случаях.

Сравнить качество полученных приближений путем сравнения их отклонений. Построить графики получившихся зависимостей и табличных значений аргументов и функции. Сделать выводы о характере аппроксимирующей функции.

3. а) Используя значения функции $y=f(x)$ (и, возможно, её производных) в ряде точек на заданном отрезке, построить аппроксимирующую функцию $p(x)$ в виде

1) интерполяционного многочлена типа Лагранжа;

2) интерполяционного многочлена Эрмита;

3) интерполяционного сплайна 3-го порядка с дополнительными условиями на концах отрезка $p'(a) = N_1, p'(b) = N_2$ или $p''(a) = M_1, p''(b) = M_2$, где N_i и M_i – соответствующие значения первой и второй производной функции $f(x)$ на концах отрезка;

4) наилучшего среднеквадратического линейного (параболического, кубического) приближения.

На основе полученной аппроксимации найти значение аргумента x_1 при заданном значении функции y_1 . Сравнить с точным значением. Вычислить $p'(x_1)$ и $\int_a^b p(x)dx$ и их точные аналоги.

б) Аппроксимировать функцию $y=f(x)$ на $[a, b]$ с помощью метода взвешенных невязок. Показать сходимость аппроксимаций.

№	$y=f(x)$	$[a, b]$	y_1
1	$y = \sin x$	$[1; 2]$	0,9725
2	$y = \ln x$	$[2; 3]$	0,9465
3	$y = \cos x$	$[5; 6]$	0,7662
4	$y = \operatorname{tg} x$	$[-1; 0]$	-0,6667
6	$y = \sqrt[3]{x}$	$[0; 1]$	0,9160
5	$y = x - \frac{1}{x}$	$[2; 3]$	2,6543
7	$y = \operatorname{ch} 2x$	$[0; 1]$	1,1575
8	$y = \cos^2 x$	$[0; 1]$	0,9431
9	$y = \frac{\operatorname{sh} 2x}{x}$	$[0,1; 1,1]$	2,9035
10	$y = \frac{\sin 4x}{x}$	$[1; 2]$	-0,5907
11	$y = \frac{\operatorname{th} 4x}{x}$	$[0;1]$	2,6555
12	$y = \frac{1}{1+4x^2}$	$[-1; 0]$	0,7754
13	$y = \operatorname{tg} x - \frac{1}{x}$	$[-1, 0]$	1,5522
14	$y = e^{-3,4x^2}$	$[0; 1]$	0,4617
15	$y = \operatorname{tg} x - \frac{1}{x^2}$	$[2; 3]$	-0,9876
16	$y = \frac{1}{3x+5}$	$[-2; -1]$	-9,8526
17	$y = \frac{x^3}{\sin x + 1,3}$	$[0,1; 1,1]$	0,1853

№	$y=f(x)$	$[a, b]$	y_1
18	$y = \sin x + \cos x$	$[0; 1]$	1,3871
19	$y = \lg x$	$[0,1; 1,1]$	-0,3333
20	$y = \frac{x^2 - 2}{\ln x + 5}$	$[0,1; 1,1]$	-0,2999
21	$5\sin^2 x$	$[0; 1]$	2,7852
22	$y = \frac{\lg x}{5x}$	$[0,5; 1,5]$	0,0028
23	$y = \frac{5}{3x + 5}$	$[0; 1]$	0,7543
24	$y = \lg x + 1,4$	$[0,5; 1,5]$	1,4545
25	$y = \sin x - \cos x$	$[0; 1]$	-0,0995
26	$y = \frac{-0,5}{3x + 0,2}$	$[0; 1]$	-0,2876
27	$y = \frac{5}{\ln x}$	$[1,1; 2,1]$	11,122
28	$y = \sqrt{x}$	$[0; 1]$	0,8787
29	$y = \operatorname{sh} x$	$[2; 3]$	6,3131
30	$y = x - \frac{1}{x^2}$	$[2; 3]$	2,9512

4*. Для функции $y=f(x)$, заданной таблицей

x	0,5	1	1,5	2	2,5	3,0
y	10,5	1,6	0,55	0,26	0,15	0,092

подберите подходящий вид аппроксимирующей зависимости из следующих:

а) $y = ax^b$; б) $y = ae^{bx}$; в) $y = a + \frac{b}{x}$; г) $y = \frac{1}{a + bx}$;

д) $y = a + b \ln x$,

находя методом наименьших квадратов их параметры и сравнивая между собой среднеквадратические погрешности.

5*. Методом наименьших квадратов аппроксимировать функцию, заданную в виде таблицы

x	0,5	1	1,5	2,0	2,5	3,0	4,0	5,0
y	0,223	0,517	0,546	0,752	0,820	0,814	1,012	1,192

зависимостью $y = \frac{x}{ax + b}$.

6*. Найти оптимальные параметры a и b зависимости $y = \frac{1}{\sqrt{2\pi b}} e^{-\frac{(x-a)^2}{2b}}$, если известно, что ей приближенно подчиняются экспериментальные точки (x_i, y_i) :

x_i	0,4	0,8	1,2	1,6	2,0	2,4	2,8	3,2	3,6
y_i	0,043	0,135	0,300	0,471	0,534	0,482	0,316	0,141	0,039

7*. Получить среднеквадратическое приближение $y = g(x)$ таблично заданной функции

x	0	0,5	1,0	1,5	2,0	2,5	3,5	5,0
y	6,57	4,15	2,29	0,442	-0,19	-0,81	-1,72	-2,05

для следующих трехпараметрических семейств функций

а) $g(x) = ae^{bx} + c$; б) $g(x) = ax^b + c$; в) $g(x) = ax^2 + bx + c$.

8*. Построить аппроксимации по методу взвешенных невязок для функций, аналитически выражающихся через интегралы, на заданном промежутке изменения аргумента. Исследовать сходимость для различных наборов весовых и базисных функций.

а) $f(x) = -\int_0^x \frac{\ln(1-t)}{t} dt, 0 \leq x \leq 1$; б) $f(x) = -\int_0^x \ln \cos t dt, 0 \leq x \leq \pi/2$;

в) $f(x) = \int_0^x e^{-t^2} dt, 0 \leq x \leq 3$; г) $f(x) = \int_0^x \frac{dt}{\ln t}, 0 \leq x \leq 0,8$;

д) $f(x) = \int_0^x \ln \frac{1+t}{1-t} \frac{dt}{t}, 0 \leq x \leq 1$; е) $f(x) = \frac{1}{\sqrt{2\pi}} \int_0^x \frac{\sin t}{\sqrt{t}} dt, 0 \leq x \leq 6$;

$$\text{ж) } f(x) = \int_0^x \cos^2 t dt, \quad 0 \leq x \leq 4; \quad \text{з) } f(x) = \int_0^x \frac{\sin t}{t} dt, \quad 0 \leq x \leq 6;$$

$$\text{и) } \int_x^\infty \frac{\cos t}{t} dt, \quad 1 \leq x \leq 6; \quad \text{к) } \int_0^x t^2 \sqrt{8-t^2} \sqrt{4-t^2} dt, \quad 0 \leq x \leq 2.$$

9*. Решить задачу локального интерполирования для функции $I(x) = \frac{2}{\pi} \int_0^x e^{-t^2} dt$ на отрезке $[0, 3]$ с шагом h . Рассмотреть случаи кусочно-линейной и кусочно-квадратичной интерполяции. Вычислить максимальную погрешность для $h=0,25; 0,1; 0,05; 0,01; 0,005$.

10*. Для функции из задачи 6 на том же отрезке провести интерполяцию Эрмита:

а) глобальную по 3–5 точкам;

б) локальную, задавая в узлах значения функции и её первой производной;

в) локальную, задавая в узлах значения функции, её первой и второй производных;

г) локальную, задавая на каждом отрезке $[x_k, x_{k+1}]$ значения $I(x_k)$, $I(x_{k+1})$, $I'(x_k)$, $I'(x_{k+1})$ и $I((x_k + x_{k+1})/2)$.

Определить максимальную погрешность интерполяции как для самой функции, так и для её первой производной. $h=1,5; 1; 0,5; 0,1$.

11*. Функция $f(x)$ задана таблицей своих значений. Построить сплайн третьего порядка и вычислить значение функции в указанных точках; $M_i \equiv S_3''(x_i)$.

а)	x_i	0,1	0,15	0,19	0,25	0,28	0,30
	$f(x_i)$	1,1052	1,1618	1,2092	1,2840	1,3231	1,3499

$$2M_1 + M_2 = 3,3722; \quad 0,5M_5 + 2M_6 = 3,3614, \quad x=0,20.$$

б)	x_i	0,2	0,24	0,27	0,30	0,32	0,38
	$f(x_i)$	1,2214	1,2712	1,3100	1,3499	1,3771	1,4623

$$2M_1 + 0,1M_2 = 2,5699; \quad 0,3M_5 + 2M_6 = 3,3378, \quad x=0,20.$$

в)	x_i	0,1	0,13	0,17	0,20	0,25	0,28
	$f(x_i)$	0,0998	0,1296	0,1692	0,1987	0,2474	0,2764

$$2M_1 + 0,5M_2 = -0,2644; \quad 0,4M_5 + 2M_6 = -0,6580, \quad x = 0,15.$$

12*. Построить кубический сплайн $S_3(x)$ на сетке x_i ($i=1, 2, \dots, n$), $x_i - x_{i-1} = h_i$, $x_1 = a$, $x_n = b$ с дополнительными условиями

а) $S'_3(a) = N_1$, $S'_3(b) = N_2$; б) $S'_3(a) = S'_3(b)$, $S''_3(a) = S''_3(b)$;

в) $S''_3(a) = M_1$, $S''_3(b) = M_2$.

13*. Функция $f(x)$ представлена своими вторыми производными $M_i = f''(x_i)$ в узлах равномерной сетки x_i ($i=1, 2, \dots, n$), $x_i - x_{i-1} = h$, $x_1 = a$, $x_n = b$. Построить кубический сплайн $S_3(x)$, аппроксимирующий $f(x)$, с учетом дополнительных условий

а) $S_3(a) = F_1$, $S_3(b) = F_2$; б) $S_3(a) = F_1$, $S'_3(a) = N_1$;

в) $A_1 S'_3(a) + B_1 S_3(a) = C_1$, $A_2 S'_3(b) + B_2 S_3(b) = C_2$;

$$(A_1^2 + B_1^2 \neq 0, \quad A_2^2 + B_2^2 \neq 0, \quad B_1^2 + B_2^2 \neq 0).$$

14*. Построить кубический сплайн, приближающий решение дифференциального уравнения $y'' - y = -3x$, $y(0) = 0$, $y(2) = 0$ при $0 \leq x \leq 2$. (Непосредственно решение $y(x)$ не находить!)

15*. Известно, что линейно независимыми решениями дифференциального уравнения $x^2 y'' - xy' + (x^2 - n^2)y = 0$ являются функции Бесселя n -го порядка $J_n(x)$ и $Y_n(x)$ — соответственно первого и второго рода. Используя метод взвешенных невязок, получить аппроксимацию решения краевой задачи для этого уравнения с параметром n и граничными условиями

а) $n=0$; $y'(1)=0$, $y(4)=-1$; б) $n=0$; $y'(1)=-1$, $y'(4)=1$;

в) $n=0$; $y(1)=0$, $y(8)=1$; г) $n^2=0,5$; $y(1)=-1$, $y'(4)=-1$;

д) $n^2=0,5$; $y(1)+y'(1)=0$, $y'(7)=5$;

е) $n=1$; $y(1)-y'(1)=-2$, $y(7)+4y'(7)=0$;

ж) $n=1$; $y'(1)=-2$, $y'(5)=-1$; з) $n=1$; $y(1)=2$, $y(10)=0$.

16*. На основе метода взвешенных невязок аппроксимировать решение дифференциальных уравнений на заданном отрезке

а) $y'(x) = \frac{2xy^3}{1-x^2y^2}$; $y(2)=1$, $2 \leq x \leq 3$;

б) $y' = y + (1+x)y^2$; $y(0)=1$, $0 \leq x \leq 0,5$;

в) $y' = y^2 + \frac{y}{x} + \frac{1}{x^2}$; $y(1)=-3$, $1 \leq x \leq 4,5$;

г) $y' = e^{-x}(y^2 + 1,8) + 3$; $y(0)=-2$, $0 \leq x \leq 1,5$;

д) $y' = \sin(2y^2) + x + z$, $z' = x + y - 2z^2 + 1$; $y(0)=1$, $z(0)=0,5$, $0 \leq x \leq 2$.

17*. Функция двух переменных $u=f(x,y)$ задана своими значениями u_k в ряде точек – узлов (x_k, y_k) , $k=1, \dots, L$ области Ω . Построить для неё интерполяционный полином $g(x,y)$ наименьшей степени, если:

Ω – треугольник, а его узлы расположены а) в его вершинах, $L=3$; б) в его вершинах и на серединах сторон, $L=6$; в) в его вершинах, по два на сторонах и один в центре, $L=10$;

Ω – четырехугольник, узлы которого есть г) его вершины, $L=4$; д) его вершины и середины сторон, $L=8$.

Для стандартных трех- и четырехугольника результат представить в общем виде $g(x,y) = \sum_{k=1}^L u_k N_k(x,y)$ и определить так называемые функции формы $N_k(x,y)$. Стандартным треугольником называется фигура, ограниченная линиями $x=0$, $y=0$, $1-x-y=0$, а стандартным четырехугольником – квадрат $[-1, 1] \times [-1, 1]$.

18*. Аппроксимировать функцию $u(x,y)$ в области Ω :

а) $u = \frac{y}{\sqrt{x^2 + y^2}}$; б) $u(x,y) = \int_1^{2xy} e^{-t^2} dt$; в) $u(x,y) = \int_1^{4x+y} \frac{\sin t}{t} dt$.

(в качестве Ω взять любой треугольник или прямоугольник, не выходящий за пределы области определения функции).

19*. Аппроксимировать на отрезке функции $y=f(x)$, заданные неявно или параметрически:

- а) $2y \ln y = x^2 - x$, $x \in [0, 3]$; б) $xy - \sqrt[3]{xy^2 + 6} = 0$; $x \in [1/5, 2]$;
 в) $\cos xy = x - y^2 + 4$; $x \in [0, 3/2]$, $y > 0$;
 г) $x^3 y + y^4 - x - 7 = 0$, $x \in [0, 5]$, $y > 0$.
 д) $x(t) = (1+t) \ln(1+t)$, $y(t) = 1/(1-t+t^2)$, $0 \leq t \leq 2$;
 е) $x(t) = \ln(1+t^2)$, $y(t) = t + \operatorname{arctg} t$, $0 \leq t \leq 6$;
 ж) $x(t) = t \ln t$, $y(t) = 4 \ln t / t$, $1 \leq t \leq 7$.

В заданиях **17–20** в качестве функции взять $y = f(x)$, $a \leq x \leq b$ из таблицы упр. 3, б (стр. 175-176), сделав замену $x = (a+b)/2 + (b-a)u/2$.

20*. Считается, что точность интерполирования можно заметно повысить за счет специального выбора узлов. Проверить этот факт для заданной на отрезке $[-1; 1]$ функции $f(u)$, взяв за узлы корни полиномов Чебышева. Нули полинома Чебышева $T_n(u)$ определяются соотношением

$$u_k = \cos(\pi(2k+1)/2n), \quad k=0, 1, \dots, n-1.$$

21*. Осуществить дискретное среднеквадратическое приближение функции $f(u)$, $u \in [-1; 1]$ на множестве точек $\{u_k = \cos(\pi k/n), k=0, 1, \dots, n\}$ линейной комбинацией многочленов Чебышева

$$g(u) = \alpha_0 T_0(u) + \alpha_1 T_1(u) + \alpha_2 T_2(u) + \dots + \alpha_n T_n(u).$$

22*. Исследовать интегральную аппроксимацию по методу Галеркина на отрезке $[-1; 1]$ в виде разложения по ортогональной системе функций Чебышева $\{T_n(u)\}$ с весом $1/\sqrt{1-x^2}$.

23*. Исследовать свойства матрицы системы уравнений при дискретном среднеквадратическом приближении функции $f(u)$, $u \in [-1; 1]$ для зависимостей $g(u) = \sum_{k=0}^n \alpha_k u^k$ и

$$g(u) = \sum_{k=0}^n \alpha_k T_k(u), \text{ где } \{T_n(u)\} - \text{система полиномов Чебышева.}$$

7. МЕТОД КОНЕЧНЫХ РАЗНОСТЕЙ

Предположим, что решается одномерная линейная краевая задача для уравнения 2-го порядка, т.е. требуется определить функцию $y(x)$, удовлетворяющую заданному дифференциальному уравнению

$$y''(x) + p(x)y'(x) + q(x)y(x) = f(x) \quad (7.1)$$

на отрезке $a \leq x \leq b$ вместе с краевыми условиями при $x=a$ и $x=b$

$$\begin{aligned} \alpha_0 y'(a) + \beta_0 y(a) &= \gamma_0, \\ \alpha_1 y'(b) + \beta_1 y(b) &= \gamma_1 \end{aligned} \quad (7.2)$$

($\alpha_i^2 + \beta_i^2 \neq 0$, $i = 0, 1$). Для решения этой задачи методом конечных разностей (МКР) прежде всего производится дискретизация независимой переменной x , т.е. строится множество (или сетка) $n+1$ точек (узлов) x_k ($k=0, 1, 2, \dots, n$) на отрезке $a \leq x \leq b$, причем $x_0=a$, $x_n=b$. Шаг сетки $h_k \equiv x_{k+1} - x_k$ из-за удобства программирования чаще всего берется постоянным ($h_1=h_2=\dots=h_n \equiv h$), но может быть и переменным (особенно когда заранее известен характер поведения решения).

Следующий этап состоит в замене в дифференциальном уравнении членов, содержащих дифференцирование, разностными отношениями. На основе формулы Тейлора можно получить выражения для производных функции

$$\left(\frac{dy}{dx} \right)_k = \frac{y_{k+1} - y_{k-1}}{2h} + O(h^2), \quad \left(\frac{d^2 y}{dx^2} \right)_k = \frac{y_{k+1} - 2y_k + y_{k-1}}{h^2} + O(h^2),$$

где $h = x_{k+1} - x_k = x_k - x_{k-1}$, $y_k \equiv y(x_k)$. Подставляя эти выражения в уравнение (7.1) для всех (может быть, за исключением граничных) узлов $\{x_k\}$, а также в уравнения граничных условий, получим систему линейных алгебраических уравнений относительно узловых значений $\{y_k\}$.

Для оценки точности получаемого решения необходимо провести расчеты для разных значений шага (не менее 3) и убедиться в том, что полученные значения функции в одних и тех

же узлах близки между собой и разность их уменьшается, что говорит о стремлении решения к некоторому пределу при $h \rightarrow 0$.

Следует отметить, что при определенных наборах коэффициентов уравнения (5.1) $p(x)$, $q(x)$, $f(x)$ и граничных условий α_i , β_i , γ_i решение рассматриваемой краевой задачи может не существовать или быть не единственным. Кроме того, даже если оно и существует и единственно, то МКР может быть неустойчивым. Не касаясь деталей этого вопроса, ниже мы будем предполагать безусловный характер устойчивости как задачи, так и метода.

МКР может быть с успехом распространен на нелинейные дифференциальные уравнения, на системы уравнений, уравнения в частных производных, а также адаптирован к решению задач Коши для различных дифференциальных уравнений. В любом случае решение дифференциального уравнения конечно-разностным методом сводится к решению системы алгебраических уравнений относительно значений искомой функции на заданном множестве точек. Эта система является линейной или нелинейной в зависимости от того, линейно или нелинейно дифференциальное уравнение. Следует обратить внимание, что МКР непосредственно дает значения функции только на дискретном множестве узловых точек, а решение во всех остальных точках области задачи можно получить с помощью какого-нибудь интерполяционного метода. Из-за этого в точность решения будет вноситься дополнительная погрешность аппроксимации.

Предложенный ниже пример позволяет проследить процесс построения разностной схемы и выявить ключевые моменты реализации МКР.

Пример. Методом конечных разностей решить краевую задачу

$$\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - \frac{2}{x}; \quad \varphi(1) = 0, \quad \varphi'(2) = 1.$$

Решение. В соответствии с МКР в исходном уравнении

$$x^2 \frac{d^2 \varphi}{dx^2} + 2x \frac{d\varphi}{dx} - 2\varphi = -1 - 2/x$$

вместо первой и второй производной записываются аппроксимации:

$$[\varphi''(x)]_i \approx \frac{\varphi_{i+1} - 2\varphi_i + \varphi_{i-1}}{h^2}, \quad [\varphi'(x)]_i \approx \frac{\varphi_{i+1} - \varphi_{i-1}}{2h}.$$

Разобьем отрезок $[1, 2]$ на четыре равные части. Получим 5 узлов $x_i = 1 + h i$; $i = 0, 1, \dots, 4$; $h = 0,25$. Запишем последовательно уравнения для всех узлов, начиная с x_1 (для узла $x_0 = 1$, которому соответствует индекс $i = 0$, не пишем, поскольку для него значение известно из граничного условия: $\varphi_0 = \varphi(1) = 0$).

Для $i = 1$ ($x = 1,25$) – первое уравнение:

$$1,25^2 \frac{\varphi_2 - 2\varphi_1 + \varphi_0}{0,25^2} + 2 \cdot 1,25^2 \frac{\varphi_2 - \varphi_0}{2 \cdot 0,25} - 2\varphi_1 = -1 - \frac{2}{1,25};$$

для $i = 2$ ($x = 1,5$) – второе уравнение:

$$1,5^2 \frac{\varphi_3 - 2\varphi_2 + \varphi_1}{0,25^2} + 2 \cdot 1,5 \frac{\varphi_3 - \varphi_1}{2 \cdot 0,25} - 2\varphi_2 = -1 - \frac{2}{1,5};$$

для $i = 3$ ($x = 1,75$) – третье уравнение:

$$1,75^2 \frac{\varphi_4 - 2\varphi_3 + \varphi_2}{0,25^2} + 2 \cdot 1,75 \frac{\varphi_4 - \varphi_2}{2 \cdot 0,25} - 2\varphi_3 = -1 - \frac{2}{1,75};$$

для $i = 4$ ($x = 2$) – четвертое уравнение

$$2,0^2 \frac{\varphi_5 - 2\varphi_4 + \varphi_3}{0,25^2} + 2 \cdot 2,0 \frac{\varphi_5 - \varphi_3}{2 \cdot 0,25} - 2\varphi_4 = -1 - \frac{2}{2,0}.$$

Здесь φ_5 – значение неизвестной функции в фиктивном узле $x_5 = 2,25$.

Имеем четыре уравнения при пяти неизвестных ($\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5$). Недостающее пятое уравнение получим, если аппроксимируем второе граничное условие, соответствующее узлу с номером $i = 4$:

$$\frac{\varphi_5 - \varphi_3}{2 \cdot 0,25} = 1.$$

Итак, решение полученной системы даст приближенные значения искомой функции в точках $x=1,25$; $x=1,5$; $x=1,75$; $x=2$. Найти значения в других точках отрезка $[1, 2]$ можно с помощью методов интерполяции.

Порядок решения в системе Maple

> **restart;**

1. Введем переменную **deq** для хранения дифференциального уравнения:

> **deq:=x^2*diff(y(x),x\$2)+2*x*diff(y(x),x)-2*y(x)=-1-2/x;**

$$deq := x^2 \left(\frac{d^2}{dx^2} y(x) \right) + 2x \left(\frac{d}{dx} y(x) \right) - 2y(x) = -1 - \frac{2}{x}$$

(**y(x)** здесь обозначает $\varphi(x)$).

2. Определим конечно-разностные уравнения.

Первое уравнение (переменная **eq1**) получим, осуществляя подстановку в **deq** вместо $\varphi''(x)$ и $\varphi'(x)$ аппроксимации $(\varphi_2 - 2\varphi_1 + \varphi_0)/0,25^2$ и $(\varphi_2 - \varphi_0)/(2 \cdot 0,25)$ соответственно, вместо **y(x)** узловую переменную **y1** (т.е. φ_1) и, наконец, вместо **x** значение 1,25:

> **eq1:=subs(diff(y(x),x\$2)=(y2-2*y1+y0)/0.25^2,**
diff(y(x),x)=(y2-y0)/(2*0.25),
y(x)=y1,x=1.25,deq);
 $eq1 = 30.00000000y2 - 52.00000000y1 + 20.00000000y0 = -2.600000000$

Аналогично зададим следующие три уравнения для точек $x=1,5$; $x=1,75$; $x=2$ – переменные **eq2**, **eq3** и **eq4** соответственно:

> **eq2:=subs(diff(y(x),x\$2)=(y3-2*y2+y1)/0.25^2,**
diff(y(x),x)=(y3-y2)/(2*0.25),y(x)=y2,x=1.5,deq);
 $eq2 = 42.00000000y3 - 74.00000000y2 + 30.00000000y1 = -2.333333333$
 > **eq3:=subs(diff(y(x),x\$2)=(y4-2*y3+y2)/0.25^2,**
diff(y(x),x)=(y4-y3)/(2*0.25),y(x)=y3,x=1.75,deq);

```
eq3 := 56.00000000 y4 - 100.0000000 y3 + 42.00000000 y2 = -2.142857143
> eq4 := subs(diff(y(x), x$2) = (y5 - 2*y4 + y3) / 0.25^2,
diff(y(x), x) = (y5 - y3) / (2*0.25), y(x) = y4, x = 2.0, deq);
eq4 := 72.00000000 y5 - 130.0000000 y4 + 56.00000000 y3 = -2.000000000
```

Пятое уравнение – это аппроксимация граничного условия $\phi'(2)=1$:

```
> eq5 := (y5 - y3) / (2*0.25) = 1;
eq5 := 2.000000000 y5 - 2.000000000 y3 = 1
```

3. Решим эту систему с помощью функции **solve**. При этом следует положить $\phi_0=0$ (или $y_0=0$), что следует из первого граничного условия:

```
> rez := solve({y0=0, eq1, eq2, eq3, eq4, eq5}, {y0, y1, y2, y3, y4, y5});
```

Получим результат:

```
rez := {y0 = 0., y1 = 0.7529029084, y4 = 1.821710593, y2 = 1.218365041,
y5 = 2.053299821, y3 = 1.553299821}
```

($y_0, y_1, \dots$ здесь обозначают $\phi_0, \phi_1, \dots$). Это и есть конечно-разностное решение, которое непосредственно дает значения искомой функции лишь в отдельных точках – узлах сетки $\{x_k\}$.

4. Сравним с точным решением:

```
> st := dsolve({deq, y(1)=0, D(y)(2)=1}, y(x));
> u := unapply(subs(st, y(x)), x);
```

$$u := x \rightarrow -\frac{11}{5} \frac{1}{x^2} + \frac{7}{10} x + \frac{1}{2} \frac{2+x}{x}$$

Выведем найденные числа $\{y_i\}$ и значения функции $u(x)$ в тех узлах, для которых найдено конечно-разностное решение:

```
> X := [1, 1.25, 1.5, 1.75, 2]:
Y := subs(rez, [y0, y1, y2, y3, y4]);
Y := [0, 0.7529029084, 1.218365041, 1.553299821, 1.821710593]
> n := 5: # число узлов сетки
> for j from 1 to n do printf(`x=%4.2g      Y=%7.5g
u=%7.5g\n`, X[j], Y[j], evalf(u(X[j]))); od;
```

x=1.00	Y=0.00000	u=0.00000
x=1.25	Y= .75290	u= .76700
x=1.50	Y=1.21836	u=1.23888
x=1.75	Y=1.55329	u=1.57806
x=2.00	Y=1.82171	u=1.85000

Совпадение есть. Погрешность $\sim 1\%$.

5. Выполним построения графика точного решения и полученных приближенных узловых значений.

```
> i:='i';
> g3:=plot([X[i],Y[i]]$i=1..n),x=1..2,style=POINT,
symbol=CIRCLE,color=black);
> plots[display](g3);
```

(Графическое построение не приводится).

6. Найдем приближенное решение в виде непрерывной функции с помощью интерполяционного метода.

Сначала проведем глобальную интерполяцию:

```
> Digits:=6: f:=interp(X,Y,a);
f:=-0.990296 a^4 + 7.12002 a^3 - 12.2729 - 19.6536 a^2 + 25.7968 a
> fu:=proc(x) global a; a:=x; RETURN(f) end;
fu:=proc(x) global a; a:=x; RETURN(f) end proc
```

Построим графики интерполяционной функции $fu(x)$ и точного решения вместе с узлами интерполяции

```
> g1:=plot(fu(x),x=1..2,color=blue):g2:=plot(u(x),
x=1..2,color=red):
> plots[display](g1,g2,g3);
```

На практике часто (особенно при большом числе узлов n) вместо глобальной интерполяции предпочтительнее использовать локальную интерполяцию (линейную, квадратичную или сплайн-интерполяцию). Ниже приведен способ построения кусочно-линейной интерполяции.

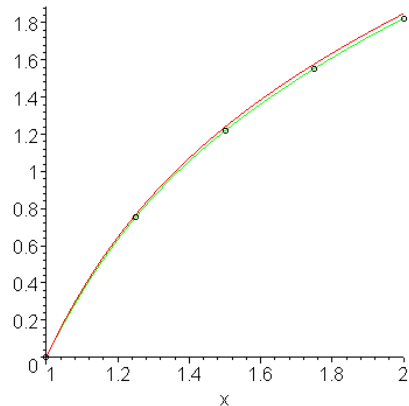


Рис. 7.1

```

> readlib (spline) ;
      proc (XX, YY, z, d) ... end proc
> f1:=spline (X,Y,x,linear) ;
      
$$f1 := \begin{cases} -3.01161 + 3.01161 x & x < 1.25 \\ -1.57444 + 1.86187 x & x < 1.5 \\ -0.791200 + 1.33972 x & x < 1.75 \\ -0.325560 + 1.07364 x & \text{otherwise} \end{cases}$$

> g4:=plot (f1,x=1..2,color=black,thickness=2) :
> g2:=plot (u (x) ,x=1..2,color=red) :
> g5:=plot ([ [X[i],Y[i]]$i=1..n] ,x=1..2.1,
style=POINT,symbol=CIRCLE) :
> plots[display] (g2,g4,g5) ;

```

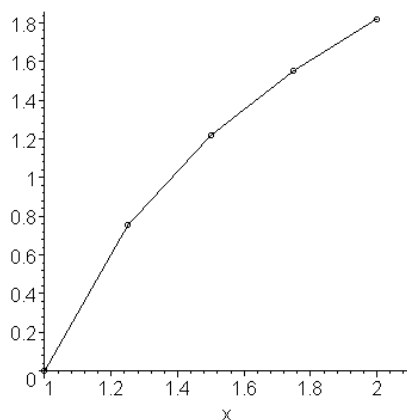


Рис. 7.2

7. Проверка сходимости метода конечных разностей.

Сходимость означает, что с уменьшением расстояния h между ближайшими узлами (а это приводит к увеличению числа узлов) точность МКР должна возрастать, т.е. должна стремиться к нулю разность $|y_i - y(x_i)|$ для любого узла с номером i при $h \rightarrow 0$. Таким образом, для проверки сходимости нужно провести серию расчетов по рассмотренной схеме МКР для разного числа узлов n .

Сделать вывод о сходимости можно визуально, отмечая число совпадающих знаков у точного и приближенного решений. Но лучше для этого использовать числовую характеристику, например

$$\delta_1 = \left\{ \frac{1}{n} \sum_{i=0}^n (y_i - y(x_i))^2 \right\}^{1/2} \quad \text{или} \quad \delta_2 = \max_i |y_i - y(x_i)|$$

Следует, однако, заметить, что тот способ реализации МКР в системе Maple, что приведен выше, достаточно нагляден, но с ростом числа узлов становится крайне громоздким. Действительно, если это число равно n , то придется вручную вводить столько же уравнений. Попробуйте оценить, на сколько возрастет текст программы при $n = 10$, $n = 20$, $n = 100$.

Поэтому приведем другую, более универсальную, Maple-программу, дающую решение данной краевой задачи методом конечных разностей при любом n (т.е. n выступает как параметр):

```
> restart;
> Digits:=15;
> a:=1; b:=2; n:=10; h:=(b-a)/n;
```

Граничные условия зададим в общем виде (10), устанавливая для коэффициентов соответствующие значения (**A0** соответствует α_0 , **B0** соответствует β_0 , **C0** соответствует γ_0 и т.д.):

```
> A0:=0; B0:=4; C0:=0; A1:=1.; B1:=0; C1:=1;
> diffeqn:=x^2*diff(y(x),x$2)+2*x*diff(y(x),x)-
2*y(x)=-1-2/x;
> Eqns_bas:={seq(subs(diff(y(x),x$2)=
(g[k+1]-2*g[k]+g[k-1])/h^2,diff(y(x),x)=(g[k+1]-
g[k-1])/(2.*h),y(x)=g[k],
x=a+h*k,diffeqn),k=0..n)};
> eq1:={A0*(g[1]-g[-1])/2./h+B0*g[0]=C0}; # гранич-
ное условие в точке x=a
> eq2:={A1*(g[n+1]-g[n-1])/2./h+B1*g[n]=C1}; # гра-
ничное условие в точке x=b
> rez:=solve(Eqns_bas union eq1 union eq2,
{g[i]$i=-1..n+1}):
```

```

> Y:=evalf([seq(subs(rez,g[k]),k=0..n)]):
> X:=evalf[5]([seq(a+k*h,k=0..n)]):
> gr1:=plot([X[i],Y[i]]$i=1..n+1,x=a..b,
style=POINT,symbol=CIRCLE,color=black):
> readlib(spline); # – получение непрерывного решения с
помощью сплайн-интерполяции
> Digits:=10: f1:=spline(X,Y,x,linear):
> gr2:=plot(f1,x=a..b,color=blue):
> x:='x': y:='y':
> p:=dsolve({diffeqn,A0*D(y)(a)+B0*y(a)=C0,
A1*D(y)(b)+B1*y(b)=C1},y(x));
> u:=unapply(subs(p,y(x)),x); # u(x) – точное решение
> gr3:=plot(u(x),x=a..b,color=red):
> plots[display]([gr1,gr2,gr3])

```

Вычисление величин δ_1 и δ_2 – норм ошибки решения:

```

> delta1:=(1/(n+1)*sum( (Y['i]-u(X[i]))^2,
'i'=1..n+1))^(0.5);
> delta2:=max( seq(abs(Y[i]-u(X[i])),i=1..n+1) );

```

Следующие строки используются для табулирования:

```

> np:=trunc(0.1/h); if(np=0) then np:=1; fi;
> for j from 1 by np to n+1 do printf(`x=%4.2g
Y=%8.5g    u=%8.5g\n`,
X[j],Y[j],evalf(u(X[j]))); od;

```

Результаты вычисления величин δ_1 и δ_2 в зависимости от n приведены в таблице.

	$n=4$	$n=8$	$n=16$	$n=50$	$n=100$
δ_1	0,04509	0,01527	0,005263	$9,3449 \cdot 10^{-4}$	$3,2883 \cdot 10^{-4}$
δ_2	0,02829	0,007121	0,001784	$1,1828 \cdot 10^{-4}$	$4,5699 \cdot 10^{-5}$

Видно, что сходимостью данный метод конечных разностей обладает.

Упражнения

1. Решить методом конечных разностей одну из следующих краевых задач. Использовать не менее 5 узлов. Сравнить с точным решением. Построить графики. Провести аппроксимацию полученного решения по методу кусочно-линейной или кусочно-квадратичной интерполяции. Показать сходимость, увеличивая каждый раз число узлов в 2 раза.

Вариант 1. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1 - \ln x^2$, $\varphi(1) = 0$, $\varphi(2) = 0$.

Вариант 2. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1 - \ln x^2$, $\varphi'(1) = 0$, $\varphi'(2) = 0$.

Вариант 3. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1 - \ln x^2$, $\varphi'(1) = 0$, $\varphi'(4) - \varphi(4) = -2$.

Вариант 4. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1 - 2 \ln x$,
 $\varphi(1) + \varphi'(1) = 0$, $\varphi(2) - \varphi'(2) = 0$.

Вариант 5. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1 - 2 \ln x$,
 $\varphi(1) - \varphi(2) = 2$, $\varphi'(2) + \varphi(1) = 1$.

Вариант 6. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = 0$, $\varphi'(1) = -1$, $\varphi'(2) = 1$.

Вариант 7. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = 1 - \frac{1}{x}$, $\varphi(1) = 0$, $\varphi'(2) = 1$.

Вариант 8. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = 2 - \ln x$, $\varphi(1) - 2\varphi'(1) = 0$, $\varphi'(2) = 1$.

Вариант 9. $\frac{d}{dx}\left((x+1)^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1$, $\varphi(0) = 0$, $\varphi'(1) = 1$.

Вариант 10. $\frac{d}{dx}\left((x+1)^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1 - \ln(x+1)$, $\varphi(0) = 0$, $\varphi'(2) = 1$.

2. Решить краевую задачу методом конечных разностей. Сравнить с точным решением. Показать сходимость. Построить график непрерывного решения на основе сплайн-интерполяции.

Вариант 1. $\frac{d}{dx} \left((x^2 + x + 1) \frac{d\varphi}{dx} \right) = x^2 - 1$; а) $\varphi(0) = 0$, $\frac{d\varphi}{dx}(1) = \frac{1}{3}$;

б) $\varphi(0) = 0$, $\varphi(1) = \frac{1}{3}$; в) $\varphi(0) + \frac{d\varphi}{dx}(0) = 0$, $\varphi(1) - \frac{d\varphi}{dx}(1) = -1$.

Вариант 2. $\frac{d}{dx} \left(e^x \frac{d\varphi}{dx} \right) = -2x$; а) $\frac{d\varphi}{dx}(0) = 0$, $\varphi(4) = 1$;

б) $\varphi(0) = 0$, $\varphi(4) = 1$; в) $2\varphi(0) - \frac{d\varphi}{dx}(0) = 0$, $\frac{d\varphi}{dx}(4) + \varphi(4) = -1$.

Вариант 3. $\frac{d}{dx} \left(\frac{1}{1+x^2} \frac{d\varphi}{dx} \right) = -2x$; а) $\varphi(0) = 0$, $\frac{d\varphi}{dx}(1) = -1$;

б) $\varphi(0) = 0$, $\varphi(1) = -1$; в) $\frac{d\varphi}{dx}(0) = -1$, $\frac{d\varphi}{dx}(1) + 4\varphi(1) = 2$.

Вариант 4. $\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) + 2\varphi = 2 - x$; а) $\varphi(1) = 1$, $\frac{d\varphi}{dx}(2) = -1$;

б) $\varphi(1) = 1$, $\varphi(2) = -1$; в) $\varphi(1) - \frac{d\varphi}{dx}(1) = 5$, $\frac{d\varphi}{dx}(2) + 2\varphi(2) = 0$.

Вариант 5. $\frac{d}{dx} \left(e^{-x} \frac{d\varphi}{dx} \right) + e^x \varphi = 0$; а) $\varphi(0) = 1$, $\frac{d\varphi}{dx}(1) = \frac{1}{2}$;

б) $\varphi(0) = 1$, $\varphi(1) = \frac{1}{2}$; в) $\varphi(0) - 5 \frac{d\varphi}{dx}(0) = -2$, $\frac{d\varphi}{dx}(1) + \varphi(1) = 2$.

Вариант 6. $\frac{d}{dx} \left((x+2) \frac{d\varphi}{dx} \right) - \frac{1}{x+2} \varphi = x^2 - \frac{1}{2}$, а) $\varphi(0) = -1$, $\frac{d\varphi}{dx}(2) = \frac{1}{2}$;

б) $\varphi(0) = 1$, $\varphi(2) = \frac{1}{2}$; в) $\frac{d\varphi}{dx}(0) - 2\varphi(0) = 1$, $\frac{d\varphi}{dx}(2) = \frac{1}{2}$.

Вариант 7. $\frac{d}{dx}\left(e^{-x} \frac{d\varphi}{dx}\right) - e^x \varphi = \frac{e^{2x}}{2}$; а) $\varphi(0) = 1$, $\frac{d\varphi}{dx}(1) = 3$;

б) $\varphi(0) = 1$, $\varphi(1) = 3$; в) $2\varphi(0) - 3\frac{d\varphi}{dx}(0) = 8$, $\frac{d\varphi}{dx}(1) + 2\varphi(1) = 2$.

Вариант 8. $\frac{d^2\varphi}{dx^2} = -\varphi + 1 - x + x^2$; а) $\frac{d\varphi}{dx}(1) = -1$, $\varphi(3) = 1$;

б) $\varphi(1) = -1$, $\varphi(3) = 1$; в) $\varphi(1) - \frac{d\varphi}{dx}(1) = -1$, $\frac{d\varphi}{dx}(2,5) + 2\varphi(2,5) = 1$.

Вариант 9. $\frac{d}{dx}\left(x^4 \frac{d\varphi}{dx}\right) - 4x^2\varphi = \frac{1}{x}$; а) $\frac{d\varphi}{dx}(1) = -3$, $\varphi(2) = 3$;

б) $\varphi(1) = -3$, $\varphi(1,9) = 3$; в) $\frac{d\varphi}{dx}(1) - 2\varphi(1) = -10$, $20\frac{d\varphi}{dx}(2) - \varphi(2) = 10$.

Вариант 10. $\frac{d}{dx}\left((e^x + 2)\frac{d\varphi}{dx}\right) = e^{x+2}$; а) $\frac{d\varphi}{dx}(0) = -10$, $\varphi(2) = \frac{1}{2}$;

б) $\varphi(0) = 10$, $\varphi(4) = \frac{1}{2}$; в) $\frac{d\varphi}{dx}(0) + 2\varphi(0) = \frac{15}{2}$, $20\frac{d\varphi}{dx}(2) + \varphi(2) = 10$.

3. С помощью МКР найти решение краевой задачи, не имеющей аналитического решения. Показать сходимость.

Вариант 1. $y''(x) + (x+1)y'(x) - 2y(x) = 2(2x-1)$,
 $4y(0,5) - y'(0,5) = 2$, $y(1) + y'(1) = 0$.

Вариант 2. $y''(x) + (x-1)y'(x) - 2y(x) = -4x$, $4y(0,5) - y'(0,5) = 2$,
 $y(1) = 3$.

Вариант 3. $y''(x) + \frac{2}{x+1}y'(x) - (x+1)y(x) = -1$, $y(0,5) = 2/3$,
 $y'(1) + 2y(1) = 0,75$.

Вариант 4. $y''(x) - (x+1)y'(x) - y(x) = \frac{2}{(x+1)^3}$,
 $y(0,5) - y'(0,5) = 2$, $y(1) = 0,5$.

Вариант 5. $y''(x) + \frac{4x}{x^2 + 1} y'(x) - \frac{1}{x^2 + 1} y(x) = -\frac{3}{(x^2 + 1)^2},$
 $y'(0) = 0, \quad y(1) = 0,5.$

Вариант 6. $y''(x) + \frac{2}{x-2} y'(x) + (x-2)y(x) = 1, \quad y(0) = -0,5,$
 $y(1) = -1.$

Вариант 7. $x^2 \ln x \cdot y''(x) - xy'(x) + y(x) = 0, \quad y(1) = 0,$
 $y(e) = e - 2.$

Вариант 8. $y''(x) - \ln x \cdot y'(x) - 2y(x) = 1, \quad y(0,5) + y'(0,5) = 1,$
 $y(1,5) - y'(1,5) = 1.$

Вариант 9. $y''(x) + y'(x) - \frac{2}{\cos^2 x} y(x) = \frac{2}{\cos^2 x}, \quad y(0) = 0,$
 $y'(0,5) = 1/\cos^2 0,5.$

Вариант 10. $y''(x) - \frac{2}{x} y'(x) - \frac{4}{x^2 + 2} y(x) = 8, \quad y'(0,5) = 0,5,$
 $y(1) + y'(1) = 1.$

4*. Решить нелинейное уравнение $e^{-y} y'' = 1$ с краевыми условиями $y(0)=y(1)=0$, используя сетку с шагом не более $h=0,25$. (Это уравнение типично для задач о распространении тепла в химически реагирующих материалах, для которых плотность источника тепла пропорциональна e^y , где y – температура).

Указание. Для решения подобных задач используются стандартные итерационные (с последовательным уточнением) процедуры, сводящиеся к многократному решению систем линейных алгебраических уравнений. Наиболее известные из таких процедур – метод простой итерации и метод Ньютона.

8. МЕТОД РИТЦА

8.1. О прямых методах вариационного исчисления

Дифференциальные уравнения (Эйлера) вариационных задач интегрируются в конечном виде лишь в исключительных случаях. В связи с этим естественно возникает потребность в иных методах решения этих задач. Основная идея так называемых *прямых методов* заключается в том, что вариационная задача рассматривается как предельная для некоторой задачи на экстремум функции конечного числа переменных. Эта задача на экстремум функции конечного числа переменных решается обычными методами математического анализа и алгебры.

Функционал $F[y(x)]$ можно рассматривать как функцию бесконечного множества переменных. Это утверждение становится совершенно очевидным, если предположить, что допустимые функции могут быть разложены в степенные ряды

$$y(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n + \dots,$$

или в ряды Фурье

$$y(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos nx + b_n \sin nx),$$

или вообще в какие-нибудь ряды вида

$$y(x) = \sum_{n=0}^{\infty} a_n \varphi_n(x), \quad (8.1)$$

где $\varphi_n(x)$ – заданные функции. Для задания функции $y(x)$, представимой в виде ряда (8.1), достаточно задать значения всех коэффициентов a_n , и, следовательно, значение функционала $F[y(x)]$ в этом случае определяется заданием бесконечной последовательности чисел $a_0, a_1, a_2, \dots, a_n, \dots$, т. е. функционал является функцией бесконечного множества переменных:

$$F[y(x)] = \Phi(a_0, a_1, \dots, a_n, \dots).$$

Следовательно, различие между вариационными задачами и задачами на экстремум функций конечного числа пере-

менных состоит в том, что в вариационном случае приходится исследовать на экстремум функции бесконечного множества переменных.

Идея метода Рунца заключается в том, что значения некоторого функционала $F[y(x)]$ рассматриваются не на произвольных допустимых кривых данной вариационной задачи, а лишь на всевозможных линейных комбинациях вида

$$y_m = \sum_{i=1}^m \alpha_i N_i(x) \text{ с постоянными коэффициентами, составлен-}$$

ных из m первых функций некоторой выбранной последовательности функций, называемых координатными или *базисными*,

$$N_1(x), N_2(x), \dots, N_m(x), \dots$$

$$\text{Функции } y_m = \sum_{i=1}^m \alpha_i N_i(x) \text{ должны быть допустимыми в}$$

рассматриваемой задаче, что налагает некоторые ограничения на выбор последовательности функций $N_i(x)$. На таких линейных комбинациях функционал $F[y(x)]$ превращается в функцию $\Phi(\alpha_1, \alpha_2, \dots, \alpha_m)$ коэффициентов $\alpha_1, \alpha_2, \dots, \alpha_m$. Эти коэффициенты выбираются так, чтобы функция $\Phi(\alpha_1, \alpha_2, \dots, \alpha_m)$ достигала экстремума; следовательно, $\alpha_1, \alpha_2, \dots, \alpha_m$ должны быть определены из системы уравнений

$$\partial\Phi/\partial\alpha_i = 0, \quad i = 1, 2, \dots, m.$$

Рассмотрим конкретные подходы к решению некоторых задач методом Рунца.

8.2. Простейшая задача вариационного исчисления.

Краевая задача 1-го рода

В этой задаче требуется найти функцию $y(x)$, доставляющую экстремум функционалу

$$F[y(x)] = \int_a^b f(x, y, y') dx$$

при условиях $y(a)=y_0$, $y(b)=y_1$. Одновременно эта функция будет являться решением краевой задачи

$$\frac{\partial f}{\partial y} - \frac{d}{dx} \frac{\partial f}{\partial y'} = 0; \quad y(a)=y_0, y(b)=y_1.$$

Граничные условия 1-го рода обычно играют роль главных условий. В этом случае множество допустимых функций, на которых ищется экстремум, приходится ограничивать только теми функциями, которые априори удовлетворяют этим условиям. Поэтому, если в исходной краевой или вариационной задаче имеются граничные условия 1-го рода

$$y(a)=y_0 \quad \text{и/или} \quad y(b)=y_1,$$

то непосредственный учет этих условий следует производить путем ограничения множества пробных функций вариационной задачи. Применительно к методу Ритца это означает выбор

такой аппроксимации $y_m = \sum_{i=1}^m \alpha_i N_i(x)$, которая заведомо удовлетворяет указанным условиям вне зависимости от того, какие значения принимают параметры $\{\alpha_i\}$.

Если граничные условия однородны, т.е. $y(a)=y(b)=0$, то проще всего в качестве базисных функций выбрать функции, удовлетворяющие этим граничным условиям: $N_i(a)=N_i(b)=0$, $i=1, 2, \dots, m$. Например,

$$N_i(x) = (x-a)(x-b)\varphi_i(x),$$

где $\varphi_i(x)$ – какие-нибудь непрерывные функции, или

$$N_k(x) = \sin \frac{k\pi(x-a)}{b-a} \quad (k=1, 2, \dots).$$

Очевидно, что при этом и аппроксимация $y_m = \sum_{i=1}^m \alpha_i N_i(x)$

при любых α_i будет удовлетворять тем же однородным граничным условиям.

Если условия неоднородны, т.е. $y(a) = y_0$, $y(b) = y_1$, где хотя бы одно из чисел y_0 или y_1 отлично от нуля, то решение вариационной задачи можно искать в виде

$$y_m = \psi(x) + \sum_{i=1}^m \alpha_i N_i(x), \quad (8.2)$$

где $\psi(x)$ удовлетворяет заданным граничным условиям: $\psi(a) = y_0$, $\psi(b) = y_1$, а функции $N_i(x)$ удовлетворяют соответствующим однородным граничным условиям, т.е. в рассматриваемом случае $N_i(a) = N_i(b) = 0$. Очевидно, что при таком выборе при любых α_i функции $y_m(x)$ удовлетворяют заданным граничным условиям. В качестве функций $\psi(x)$ можно выбрать, например, линейную функцию

$$\psi(x) = \frac{y_1 - y_0}{b - a}(x - a) + y_0,$$

а в качестве базисных – функции

$$N_i(x) = (x - a)(x - b)x^{i-1}.$$

(Убедитесь, что оба условия $y(a) = y_0$ и $y(b) = y_1$ выполняется здесь априори).

На аппроксимации вида (8.1) или (8.2) функционал $F[y(x)]$ превращается в функцию $\Phi(\alpha_1, \alpha_2, \dots, \alpha_m)$ коэффициентов $\alpha_1, \alpha_2, \dots, \alpha_m$. Эти коэффициенты выбираются так, чтобы функция $\Phi(\alpha_1, \alpha_2, \dots, \alpha_m)$ достигала экстремума; следовательно, $\alpha_1, \alpha_2, \dots, \alpha_m$ должны быть определены из системы уравнений

$$\partial\Phi/\partial\alpha_i = 0, \quad i = 1, 2, \dots, m.$$

Совершая предельный переход при $m \rightarrow \infty$, получим в случае существования предела функцию $y = \sum_{i=1}^{\infty} \alpha_i N_i(x)$, являющуюся точным решением рассматриваемой вариационной задачи. Если не совершать предельного перехода, а ограничиться лишь m первыми членами, то получим приближенное решение вариационной задачи.

8.3. Вариационная задача Больца.

Краевая задача 2-го и 3-го рода

Задача Больца состоит в определении функции $y(x) \in C_1[a, b]$, доставляющей экстремум функционалу

$$F[y(x)] = \int_a^b f(x, y, y') dx + g(y(a), y(b)).$$

Задача Больца – это особый тип задачи с подвижными границами, в которой граничные точки кривой могут перемещаться вдоль соответствующих вертикальных прямых $x=a$ и $x=b$. Экстремум здесь достигается только на кривых, удовлетворяющих уравнению Эйлера

$$\frac{\partial f}{\partial y} - \frac{d}{dx} \frac{\partial f}{\partial y'} = 0$$

и условиям трансверсальности

$$\left[\frac{\partial f}{\partial y'} - \frac{\partial g}{\partial y(a)} \right]_{x=a} = 0, \quad \left[\frac{\partial f}{\partial y'} + \frac{\partial g}{\partial y(b)} \right]_{x=b} = 0.$$

Надлежащим выбором функций f и g в функционале можно добиться эквивалентной вариационной формулировки дифференциальных краевых задач 2 и 3 рода. Например, функционал

$$F[y(x)] = \frac{1}{2} \int_a^b (py'^2 + qy^2 - 2fy) dx + \frac{\beta_2}{2} p(b)y(b)^2 - \gamma_2 p(b)y(b) - \frac{\beta_1}{2} p(a)y(a)^2 + \gamma_1 p(a)y(a) \quad (8.3)$$

достигает экстремума на функции $y(x)$, которая одновременно удовлетворяет дифференциальному уравнению 2-го порядка (уравнению Эйлера)

$$\frac{d}{dx} \left(p(x) \frac{d}{dx} \right) - q(x)y = -f(x) \quad (8.4)$$

и граничным условиям 3-го рода (условиям трансверсальности)

$$y'(a) + \beta_1 y(a) = \gamma_1, \quad y'(b) + \beta_2 y(b) = \gamma_2. \quad (8.5)$$

Отсюда следует, что если уравнение для некоторой краевой задачи имеет вид (8.4), а граничные условия – (8.5), то эквивалентная вариационная формулировка этой задачи дается функционалом (8.3). При этом заметим, что граничные условия (8.5) могут быть не только 3-го, но и 2-го рода; для этого достаточно положить $\beta_1 = 0$ и/или $\beta_2 = 0$. Указанные граничные условия являются естественными. Это означает, что функция, доставляющая экстремум функционалу на множестве всех допустимых функций, будет автоматически удовлетворять как дифференциальному уравнению, так и естественному граничному условию.

В силу естественности граничных условий аппроксимация неизвестной функции может выбираться без ограничений на границе. В методе Ритца обычно $y(x)$ записывают в виде разложения по системе линейно независимых функций $\{1, x, x^2, x^3, \dots\}$, т.е. в виде полного многочлена с неопределенными коэффициентами:

$$y(x) \approx y_m(x) = \alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots + \alpha_m x^{m-1}.$$

Разумеется, в смешанной задаче, когда присутствуют одновременно и главное, и естественное условия, аппроксимация должна обеспечивать выполнение граничного условия 1-го рода, записанного для соответствующей точки (как в простейшей задаче). Например, если в точке $x=a$ выполняется условие $y(a)=y_0$, а в точке $x=b$ – условие типа $y'(b) + \beta y(b) = \gamma$, то аппроксимация может быть взята в виде

$$y_m(x) = y_0 + (\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots + \alpha_m x^{m-1})(x - a),$$

а функционал (8.3) не будет содержать граничных слагаемых, вычисляемых в точке $x=a$:

$$F = \frac{1}{2} \int_a^b [p(x)y'^2 - q(x)y^2 + 2f(x)y] dx + \frac{1}{2} \beta p(b)y^2(b) - \gamma p(b)y(b).$$

Пример 1. Решить краевую задачу методом Ритца:

$$\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1 - \frac{2}{x}; \quad \varphi(1) = 0, \quad \varphi'(2) = 1.$$

Решение. Сначала найдем эквивалентную вариационную формулировку данной задачи. Сравнивая уравнение (8.4) с нашим уравнением, получаем, что $p(x) = x^2$, $q(x) = 2$, $f(x) = 1 + 2/x$. Аналогично из вида граничного условия $y'(2) + \beta_2 y(2) = \gamma_2$ находим $\beta_2 = 0$, $\gamma_2 = 1$. Граничное условие при $x = 1$ (1-го рода) является главным. Таким образом, в данной задаче используем функционал

$$F[\varphi(x)] = \frac{1}{2} \int_1^2 \left(x^2 \varphi'^2 + 2\varphi^2 - 2(1 + 2/x)\varphi \right) dx - 4\varphi(2).$$

Аппроксимирующие функции, при любых значениях параметров $\{\alpha_i\}$ удовлетворяющие главному условию $\varphi(1)=0$, можно выбрать в виде

$$\varphi_m(x) = (\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots + \alpha_m x^{m-1})(x-1). \quad (8.6)$$

Порядок решения в системе Maple

1. Задание аппроксимации, заведомо удовлетворяющей условию $\varphi(1)=0$.

> **restart;** *# очистка оперативной памяти*

Возьмем четыре параметра аппроксимации, т. е. в (8.6) положим $m=4$.

```
> z:=x->( (a1+a2*x+a3*x^2+a4*x^3) * (1-x) );
      z := x → (a1 + a2 x + a3 x^2 + a4 x^3) (1 - x)
```

Тем самым, аппроксимирующая функция задана как **z (x)**.

2. Вычисление функционала от этой функции

```
> F:=z->(1/2.) *int (x^2*diff (z (x) , x) ^2+2*z (x) ^2-
2* (1+2*1/x) *z (x) , x=1 .. 2) -4*z (2) ;
```

$$F := z \rightarrow \frac{1}{2} \int_1^2 \left(x^2 \left(\frac{d}{dx} z(x) \right)^2 + 2 z(x)^2 - 2 \left(1 + \frac{2}{x} \right) z(x) \right) dx - 4 z(2)$$

> **P:=F(z) ;**

$$F := \frac{154 a_2 a_3}{5} + \frac{271 a_1 a_4}{10} + \frac{79 a_1 a_3}{6} + \frac{13 a_1}{2} + \frac{59 a_2}{6} + \frac{71 a_2^2}{10} + \frac{19 a_2 a_1}{3} + \frac{3 a_1^2}{2} \\ + \frac{229 a_3}{12} - 2 a_1 \ln(2) + \frac{2237 a_4}{60} + \frac{4607 a_4^2}{28} + \frac{458 a_2 a_4}{7} + \frac{597 a_3 a_4}{4} + \frac{2407 a_3^2}{70}$$

Функционал теперь зависит от параметров a_1, a_2, a_3, a_4 , т.е. он превратился в функцию многих переменных (обозначена через **P**).

Замечания. 1) В ряде случаев Maple может некорректно выводить функционал, представляя его в виде дробно-рациональной функции относительно параметров $\{a_i\}$, в то время как он должен быть квадратичным по ним. Подобная ситуация делает применение метода Ритца невозможным. Как правило, действие Maple-функции **expand** на второе слагаемое под знаком интеграла в функционале, т.е. qy^2 , помогает устранить проблему. В частности, в рассматриваемом примере в функционале **F** вместо **2*z(x)^2** следует использовать **expand(2*z(x)^2)**.

2) Как известно, Maple, в первую очередь, стремится проводить символьные вычисления, поэтому и функционал здесь выведен в символьном виде, что не всегда бывает целесообразным. Добиться числового формата можно, как всегда, с помощью **evalf**. Например, вместо строки **P:=F(z) ;** записать **P:=evalf(F(z)) ;**

3. Поиск минимума функционала как функции многих переменных.

Применяя необходимое условие экстремума $\partial F / \partial a_i = 0$, $i = 1, 2, 3, 4$, получим систему алгебраических уравнений:

> **eqns:={diff(P,a1)=0,diff(P,a2)=0,diff(P,a3)=0,diff(P,a4)=0};**

$$eqns := \left\{ \frac{271 a_1}{10} + \frac{2237}{60} + \frac{4607 a_4}{14} + \frac{458 a_2}{7} + \frac{597 a_3}{4} = 0, \right. \\ \left. \frac{271 a_4}{10} + \frac{79 a_3}{6} + \frac{13}{2} + \frac{19 a_2}{3} + 3 a_1 - 2 \ln(2) = 0, \right.$$

$$\frac{154 a3}{5} + \frac{59}{6} + \frac{71 a2}{5} + \frac{19 a1}{3} + \frac{458 a4}{7} = 0,$$

$$\frac{154 a2}{5} + \frac{79 a1}{6} + \frac{229}{12} + \frac{597 a4}{4} + \frac{2407 a3}{35} = 0\}$$

Для решения используем функцию **solve**:

> **r:=solve(eqns, {a1, a2, a3, a4}) ;**

$$r := \{a4 = \frac{232015}{429} - \frac{1336405}{1716} \ln(2), a2 = \frac{1377873}{286} - \frac{23785055}{3432} \ln(2),$$

$$a3 = -\frac{401681}{143} + \frac{6938435}{1716} \ln(2), a1 = -\frac{390421}{143} + \frac{4485175}{1144} \ln(2)\}$$

Maple вывел точный результат, но он неудобен для восприятия. Представим его в числовом формате в виде десятичных дробей:

> **r1:=evalf(r) ;**

$$r1 := \{a1 = -12.658736, a4 = 1.0108637, a2 = 13.966255, a3 = -6.302647\}$$

Это и есть оптимальные параметры, обеспечивающие наилучшее приближение в классе функций вида (8.6).

Решение данной задачи – это функция **z(x)**, заданная в п. 1, с только что найденными числовыми параметрами **a1**, **a2**, **a3** и **a4**, которые хранятся в переменной **r1** (или **r**). Результат подстановки этих параметров в **z(x)** оформим в виде новой функции **phi(x)**:

> **phi:=unapply(subs(r1, z(x)), x) ;**

$$\phi := x \rightarrow (-12.658736 + 13.966255 x - 6.302647 x^2 + 1.0108637 x^3) (1 - x)$$

Следует помнить, что здесь коэффициенты записаны с ограниченным числом значащих цифр. В некоторых случаях это может привести к дополнительной ошибке в решении. Уменьшить такую ошибку можно заданием фиксированного числа разрядов в переменной **Digits**. Другой способ – использование переменной **r** вместо **r1** в определении функции **phi**.

4. Представление полученной функции в виде таблицы и графика.

Обращаться к функции-решению теперь можно обычным способом, например, вычислить значение в некоторой точке:

```
> phi(1.25);
```

```
0.7686150060
```

Можно также протабулировать эту функцию, т.е. вывести ее значения на определенном интервале изменения аргумента в определенных (чаще равноотстоящих) точках. К примеру, изменяя аргумент от 1 до 2 с шагом 0,25, получим

```
> for t from 1 by 0.25 to 2 do printf(`x=%g  
z=%g\n`, t, phi(t)); od;
```

```
x=1          z=0  
x=1.25       z=.768615  
x=1.50       z=1.239322  
x=1.75       z=1.576536  
x=2.00       z=1.849904
```

Построим график решения:

```
> plot(phi(x), x=1..2);
```

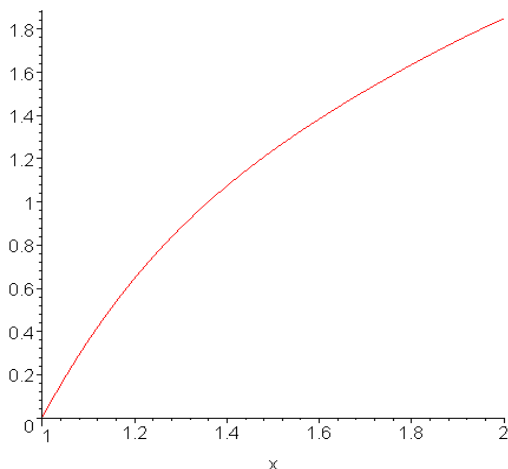


Рис. 8.1

5. Получение точного решения.

Данная краевая задача может быть решена точно аналитическими методами. Точное решение позволит оценить погрешность приближенного решения, полученного по методу Рунге.

```
> st:=dsolve({diff(x^2*diff(y(x),x),x)-2*y(x)=  
-1-2*1/x,y(1)=0,D(y)(2)=1},y(x));
```

$$st := y(x) = -\frac{11}{5x^2} + \frac{7x}{10} + \frac{2+x}{2x}$$

> u:=unapply(subs(st,y(x)),x); # теперь точное решение
есть функция **u(x)**

6. Сравнение точного и приближенного решений.

Сначала сравним результаты в отдельных равноотстоящих точках:

**> for t from 1 by 0.25 to 2 do printf(`x=%g z=%g`
u=%g\n`,t,phi(t),u(t)); od;**

x=1	z=0	u=0
x=1.25	z=.768615	u=.767
x=1.50	z=1.239322	u=1.238889
x=1.75	z=1.576536	u=1.578061
x=2.00	z=1.849904	u=1.85

Видим хорошее соответствие. Расхождение – менее 1%.

Для визуального сравнения построим графики:

**> plot([phi(x),u(x)],x=1..2,color=[red,blue],
thickness=[2,2]);**

Они практически совпадают.

7. Проверка сходимости метода.

Для получения адекватного результата численный или приближенный метод должен обладать свойством сходимости. Это означает, что с увеличением сила параметров аппроксимации приближенное решение должно стремиться к точному решению.

Конечно, в определенной степени дать оценку совпадения результатов можно, анализируя табличные данные или графики функции, как это делалось в предыдущем пункте. Но такой подход математически нестрог и годится только для качественного анализа. Для количественной характеристики процесса сходимости используется особая величина – норма ошибки решения, определяемая по формуле

$$\delta = \left(\int_a^b [\varphi_m(x) - \varphi(x)]^2 dx \right)^{1/2}$$

(здесь $\varphi(x)$ – точное решение). Если метод сходится, то δ приближается к нулю с увеличением m . То, насколько быстро это происходит, характеризует скорость сходимости. Вычисление нормы ошибки можно организовать следующим образом:

```
> evalf(int( (phi(x)-u(x))^2, x=1..2)^(1/2) ;
0.001770384139
```

Итак, для проверки сходимости нужно провести ряд вычислений по предложенной схеме для разного числа параметров аппроксимации. В приведенном алгоритме реализации это потребует изменения текста в п. 1, а также при необходимости п. 3. Например, при $m=3$ поменять надо только выражение для аппроксимации:

```
> z:=x->( (a1+a2*x+a3*x^2)*(1-x) ) ;
```

а все остальные команды просто перезапустить на выполнение. Если же взять больше четырех параметров, к примеру, $m=5$, то помимо аппроксимации

```
> z:=x->( (a1+a2*x+a3*x^2+a4*x^3+a5*x^4)*(1-x) ) ;
```

из-за появления нового параметра **a5** также должны быть модифицированы команды пункта 3:

```
> eqns:={diff(P,a1)=0,diff(P,a2)=0,diff(P,a3)=0,
diff(P,a4)=0, diff(P,a5)=0};
```

```
> r:=solve(eqns, {a1,a2,a3,a4,a5}) ;
```

Помимо оценки сходимости решения интерес представляет сходимость производной $\varphi'_m(2)$ к точному значению 1 (естественное граничное условие) и сходимость функционала. Вывод этих величин осуществляется очень простыми командами:

```
> D(phi)(2) ;
```

```
> F(phi) ;
```

Точное значение функционала определяется так

```
> evalf(F(u)) ;
```

-4.993147181

Результаты вычислений представлены в таблице.

	$m=2$	$m=3$	$m=4$	$m=5$	$m=6$
δ	0.0513	0.00913	$1.77 \cdot 10^{-3}$	$3.57 \cdot 10^{-4}$	$6.81 \cdot 10^{-5}$
$\Phi'_m(2)$	0.673	1.118	0.964	1.010	0.997
$F[\Phi_m]$	-4.916386	-4.986417	-4.992677	-4.993119	-4.993146

Данные из таблицы свидетельствуют о высокой скорости сходимости метода Ритца.

8.4. Метод Ритца в двумерном случае

Пусть требуется найти решение уравнения

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -Q(x, y), \quad (8.7)$$

называемого уравнением Пуассона, внутри двумерной области Ω , принимающее заданные значения на границе Γ (так называемая задача Дирихле).

В вариационном исчислении показано, что эта краевая задача равносильна задаче об исследовании на минимум функционала

$$F[z(x, y)] = \frac{1}{2} \iint_{\Omega} \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 - 2uQ(x, y) \right] dx dy. \quad (8.8)$$

на множестве функций u , удовлетворяющих на границе Γ области Ω граничному условию 1-го рода $u|_{\Gamma} = \varphi(x, y)$.

Таким образом, задача поиска решения задачи Дирихле сводится к задаче поиска функции, минимизирующей функционал. Сама по себе такая задача минимизации ничуть не легче исходной, если функционал рассматривать на всем множестве допустимых (гладких) функций. Однако, если такое множество сузить, ограничив его только линейными комбинациями вида

$$u^*(x, y) = \sum_i \alpha_i N_i(x, y), \quad (8.9)$$

удовлетворяющими граничному условию, то задача существенно упростится. Теперь требуется всего лишь решить задачу на экстремум функции нескольких переменных. Действительно, считая α_i неизвестными параметрами (числами), а N_i – известными функциями координат (базисными функциями), подставим аппроксимацию $u^*(x, y)$ в функционал (8.8):

$$F[u^*] = \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j \iint_{\Omega} \left[\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} \right] dx dy - \sum_i \alpha_i \iint_{\Omega} N_i Q(x, y) dx dy \equiv \Phi(\alpha_1, \alpha_2, \dots, \alpha_m).$$

Как видно, функционал F после такой подстановки превратился в функцию Φ переменных $\alpha_1, \alpha_2, \dots, \alpha_m$. Так как для существования экстремума функции n переменных необходимо выполнение условий

$$\frac{\partial \Phi}{\partial \alpha_i} = 0,$$

получим систему линейных алгебраических уравнений относительно параметров $\{\alpha_i\}$

$$\mathbf{S}\mathbf{a} = \mathbf{G}$$

$$S_{ij} = \iint_{\Omega} \left[\frac{\partial N_i}{\partial x} \frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y} \frac{\partial N_j}{\partial y} \right] dx dy; \quad G_i = \iint_{\Omega} N_i Q(x, y) dx dy.$$

Эти параметры полностью определяют аппроксимацию искомого решения согласно (8.9). Чем больше берется слагаемых в (8.9), или, что то же самое, параметров аппроксимации, тем ближе полученное по методу Ритца решение приближается к точному решению. (Это верно при условии полноты системы базисных функций $\{N_i\}$).

В общей постановке задачи для уравнения (8.7) может ставиться на всей или части границы Γ выполнение условия, включающее нормальную производную, в частности,

$$\frac{\partial u}{\partial n} = -q \quad (\text{условие Неймана})$$

и/или

$$\frac{\partial u}{\partial n} + \beta z = -\gamma \quad (\text{условие 3-го рода})$$

(здесь q , β и γ – заданные функции координат).

В этом случае учет этих условий обеспечивается за счет минимизации функционала, отличающегося от (8.8) появлением дополнительных граничных слагаемых – криволинейных интегралов 1-го рода по соответствующей части границы:

$$F[z(x, y)] = \frac{1}{2} \iint_{\Omega} \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 - 2uQ(x, y) \right] dx dy + \int_{\Gamma_2} qu d\Gamma$$

и

$$F[z(x, y)] = \frac{1}{2} \iint_{\Omega} \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 - 2uQ(x, y) \right] dx dy + \int_{\Gamma_2} \left[\frac{\beta}{2} u^2 + \gamma u \right] d\Gamma \quad (8.10)$$

соответственно.

Пример 2. Найти стационарное распределение температуры в прямоугольной пластине $0 \leq x \leq a$, $0 \leq y \leq b$ ($a = 5$ и $b = 4$), все стороны которой поддерживаются при постоянной температуре $u = 20^\circ$. Пластина нагревается от источников тепла, мощность которых описывается законом $Q(x, y) = x^2 + y^2$.

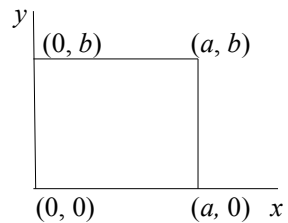


Рис. 8.2

Решение. Задача сводится к решению уравнения

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -(x^2 + y^2)/k$$

(k – коэффициент теплопроводности, положим $k=1$) с граничными условиями $u(0, y) = u(5, y) = u(x, 0) = u(x, 4) = 20$.

Имеем краевые условия 1-го рода вдоль всей границы области Ω . Чтобы обеспечить их выполнение, воспользуемся аппроксимацией вида

$$u(x, y) = \psi + \sum_k \alpha_k N_k(x, y).$$

При этом если за базисные функции взять произведения $x^i y^j (x-a)(y-b)$, $i, j = 1, 2, 3, \dots$, то на границах прямоугольной области они обращаются в нуль, и, тем самым, положив $\psi=20$, граничные условия будут удовлетворены точно вне зависимости от значений $\{\alpha_k\}$. Итак,

$$u_{mn}(x, y) = 20 + \sum_{i=1}^m \sum_{j=1}^n \alpha_{ij} x^i y^j (x-a)(y-b), \quad (8.11)$$

где для удобства использованы индексы двойного суммирования.

Поскольку в задаче есть только условия 1-го рода (Дирихле), то используем функционал

$$F[u(x, y)] = \frac{1}{2} \iint_{\Omega} \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 - 2u(x^2 + y^2) \right] dx dy.$$

Выполняя затем стандартную процедуру Рунта, найдем оптимальные значения параметров $\{\alpha_{ij}\}$, а вместе с тем (см. (8.11)) и приближенное решение задачи.

Порядок решения в системе Maple

1. Зададим исходные данные задачи, а также верхние границы изменения индексов i, j – числа m и n , произведения которых определяет число параметров аппроксимации. Их увеличение приводит к повышению точности решения и одновременно усложняет расчеты.

> **restart;**

> **a:=5: b:=4: f:=x^2+y^2: m:=2: n:=2;**

2. Запишем аппроксимацию решения задачи

```
> z:=(x,y)->20+sum(sum(alpha[i,j]*x^i*y^j*  
(x-a)*(y-b),i=1..m),j=1..n);
```

3. Зададим функционал в виде повторного интеграла

```
> F:=(1/2.)*int(int(diff(z(x,y),x)^2+  
diff(z(x,y),y)^2-2*f*z(x,y),y=0..b),x=0..a);
```

4. Формируем уравнения системы

```
> eqns:=seq(seq(diff(F,alpha[i,j]),i=1..m),j=1..n);  
> p:=seq(seq(alpha[i,j],i=1..m),j=1..n);
```

5. Находим решение этой системы

```
> r:=solve({eqns},{p});
```

6. Определяем функцию $\varphi(x, y)$, хранящую решение

```
> phi:=unapply(subs(r,z(x,y)),x,y);
```

7. Теперь решение задачи можно получить в любых точках прямоугольной области, например, вдоль линии $x=2$:

```
> [phi(2,0),phi(2,1),phi(2,2),phi(2,3),phi(2,4)];  
[20., 29.70911124, 35.72234400, 33.87440476, 20.]
```

8. Визуализация решения – вывод в виде цветовой карты и линий равного уровня

```
> plots[densityplot](phi(x,y),x=0..a,y=0..b,  
grid=[49,49],colorstyle=HUE);  
> plots[contourplot](phi(x,y),x=0..a,y=0..b,  
contours=15);
```

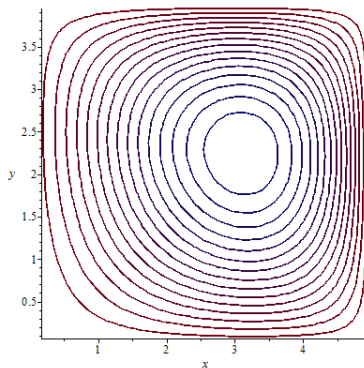


Рис. 8.3

Замечания. Если в задаче имеются краевые условия 2-го или 3-го рода вдоль какой-то стороны, то аппроксимация не должна удовлетворять каким бы то ни было условиям на этой стороне. При этом в функционал необходимо включить дополнительный интеграл по данной стороне (см. (8.10)).

Изучите внимательно нижеследующие примеры.

$$1) u(0, y) = 1, \quad u(a, y) = 2; \quad \frac{\partial u}{\partial n}(x, 0) = 1, \quad \frac{\partial u}{\partial n}(x, b) = -1.$$

Аппроксимация учитывает только два первых условия как 1-го рода

$$u_{mn}(x, y) = 1 + \frac{x}{a} + \sum_{i=1}^m \sum_{j=0}^n \alpha_{ij} x^i y^j (x - a).$$

(Обратите внимание на начало суммирования по j , а также на отсутствие множителя $y - b$). В функционале к кратному интегралу по прямоугольнику добавляются интегральные слагаемые вдоль сторон $x = 0$ и $x = a$:

$$F[u(x, y)] = \frac{1}{2} \iint_{\Omega} \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 - 2uf \right] dx dy - \\ - \int_0^a u(x, 0) dx + \int_0^a u(x, b) dx.$$

Изменения в программе коснутся соответствующих строк:

```
> z:=(x,y)->1+x/a+sum(sum(alpha[i,j]*x^i*y^j*
(x-a),i=1..m),j=0..n);
> F:=(1/2.)*int(int(diff(z(x,y),x)^2+
diff(z(x,y),y)^2-2*f*z(x,y),y=0..b),x=0..a)-
int(z(x,0),x=0..a)+int(z(x,b),x=0..a);
> eqns:=seq(seq(diff(F,alpha[i,j]),i=1..m),j=0..n);
> p:=seq(seq(alpha[i,j],i=1..m),j=0..n);
```

$$2) u(0, y) = 10, \quad u(x, 0) = 10; \quad \frac{\partial u}{\partial n}(x, b) + u(x, b) = 0, \\ \frac{\partial u}{\partial n}(a, y) + u(a, y) = 0.$$

Аппроксимация $u_{mn}(x, y) = 10 + \sum_{i=1}^m \sum_{j=1}^n \alpha_{ij} x^i y^j$;

Функционал

$$F[u(x, y)] = \frac{1}{2} \iint_{\Omega} \left[\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 - 2uf \right] dx dy + \\ + \frac{1}{2} \int_0^a u^2(x, b) dx + \frac{1}{2} \int_0^b u^2(a, y) dy.$$

Упражнения

1. Решить краевую задачу методом Ритца. Сравнить с точным решением. Построить графики. Показать сходимость для $m = 2 \div 6$.

Вариант 1. $\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - \ln x^2, \quad \varphi(1) = 0, \varphi(2) = 0.$

Вариант 2. $\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - \ln x^2, \quad \varphi'(1) = 0, \varphi'(2) = 0.$

Вариант 3. $\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - \ln x^2, \quad \varphi'(1) = 0, \varphi'(4) - \varphi(4) = -2.$

Вариант 4. $\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - 2 \ln x, \\ \varphi(1) + \varphi'(1) = 0, \varphi(2) - \varphi'(2) = 0.$

Вариант 5. $\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - 2 \ln x, \\ \varphi(1) - \varphi(2) = 2, \varphi'(2) + \varphi(1) = 1.$

Вариант 6. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = 0, \quad \varphi'(1) = -1, \varphi'(2) = 1.$

Вариант 7. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = 1 - \frac{1}{x}, \quad \varphi(1) = 0, \varphi'(2) = 1.$

Вариант 8. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = 2 - \ln x, \quad \varphi(1) - 2\varphi'(1) = 0, \varphi'(2) = 1.$

Вариант 9. $\frac{d}{dx}\left((x+1)^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1, \quad \varphi(0) = 0, \varphi'(1) = 1.$

Вариант 10. $\frac{d}{dx}\left((x+1)^2 \frac{d\varphi}{dx}\right) - 2\varphi = -1 - \ln(x+1),$
 $\varphi(0) = 0, \varphi'(2) = 1.$

2. Решить уравнение $d^2u/dx^2 + \omega^2 u = x, \quad 0 \leq x \leq 1$ с краевыми условиями:

- 1) $u=0$ при $x=0$ и $u=0$ при $x=1$;
- 2) $u=1$ при $x=0$ и $du/dx=0$ при $x=1$;
- 3) $u=0$ при $x=0$ и $du/dx+u=0$ при $x=1$.

Сравнить с точным решением.

3. Решить уравнение $d^2u/dx^2 - \omega^2 u = 0, \quad 0 \leq x \leq 1$ с краевыми условиями $u=0$ при $x=0$ и $du/dx + 10u = 20$ при $x=1$. Сравнить с точным решением.

4. Решить краевую задачу методом Рунта для разных типов граничных условий. Сравнить с точным решением. Показать сходимость.

Вариант 1. $\frac{d}{dx}\left((x^2+1) \frac{d\varphi}{dx}\right) - 2\varphi = 1 - x - x^2;$

а) $\varphi(1) = -\frac{2}{3}, \quad \frac{d\varphi}{dx}(3) = -\frac{1}{2};$ б) $\varphi(1) = 2, \quad \varphi(4) = -\frac{1}{2};$

в) $\varphi(1) - 2\frac{d\varphi}{dx}(1) = -2, \quad \varphi(3) + 3\frac{d\varphi}{dx}(3) = -2.$

Вариант 2. $\frac{d}{dx}\left(e^{-x} \frac{d\varphi}{dx}\right) - e^{-x}\varphi = -2\cos(x+1);$

а) $\frac{d\varphi}{dx}(0) = -3, \quad \varphi(2) = 5; \quad \text{б) } \varphi(0) = 1, \quad \varphi(2) = 0;$

в) $\frac{d\varphi}{dx}(0) = -1, \quad \frac{d\varphi}{dx}(3) + 2\varphi(3) = -6.$

Вариант 3. $\frac{d}{dx}\left(\frac{1}{1+x^2} \frac{d\varphi}{dx}\right) - \varphi = -2; \quad \text{а) } \frac{d\varphi}{dx}(-1) = \frac{5}{2}, \quad \varphi(1) = 3;$

б) $\varphi(-1) = 2, \quad \varphi(1.5) = 4; \quad \text{в) } \frac{d\varphi}{dx}(-1) + 2\varphi(-1) = 0, \quad \frac{d\varphi}{dx}(1) + \varphi(1) = -4.$

Вариант 4. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - \varphi = 5\ln x; \quad \text{а) } \frac{d\varphi}{dx}(1) = -\frac{3}{2}, \quad \varphi(2.4) = 2;$

б) $\varphi(1) = 1, \quad \varphi(2) = 0; \quad \text{в) } \varphi(1) + \frac{d\varphi}{dx}(1) = 4, \quad \frac{d\varphi}{dx}(2) - 2\varphi(2) = -3.$

Вариант 5. $\frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) + 2\varphi = 2 - x; \quad \text{а) } \varphi(1) = 1, \quad \frac{d\varphi}{dx}(2) = -1;$

б) $\varphi(1) = 1, \quad \varphi(2) = -1; \quad \text{в) } \varphi(1) - \frac{d\varphi}{dx}(1) = 5, \quad \frac{d\varphi}{dx}(2) + 2\varphi(2) = 0.$

Вариант 6. $\frac{d}{dx}\left(e^{-x} \frac{d\varphi}{dx}\right) + e^{-x}\varphi = -4; \quad \text{а) } \varphi(0) = 0, \quad \frac{d\varphi}{dx}(1) = -2;$

б) $\varphi(0) = 1, \quad \varphi(1) = \frac{1}{2}; \quad \text{в) } \varphi(0) - 5\frac{d\varphi}{dx}(0) = -2, \quad \frac{d\varphi}{dx}(1) + \varphi(1) = 2.$

Вариант 7. $\frac{d}{dx}\left((x+2) \frac{d\varphi}{dx}\right) - \frac{1}{x+2}\varphi = x^2 - \frac{1}{2}; \quad \text{а) } \varphi(0) = -1, \quad \frac{d\varphi}{dx}(2) = \frac{1}{2};$

б) $\varphi(0) = 1, \quad \varphi(2) = \frac{1}{2}; \quad \text{в) } \frac{d\varphi}{dx}(0) - 2\varphi(0) = 1, \quad \frac{d\varphi}{dx}(2) = \frac{1}{2}.$

Вариант 8. $\frac{d^2\varphi}{dx^2} = -\varphi + 1 - x + x^2; \quad \text{а) } \frac{d\varphi}{dx}(1) = -1, \quad \varphi(3) = 1;$

$$\text{б) } \varphi(1) = -1, \varphi(3) = 1; \text{ в) } \varphi(1) - \frac{d\varphi}{dx}(1) = -1, \frac{d\varphi}{dx}(2,5) + 2\varphi(2,5) = 1.$$

$$\text{Вариант 9. } \frac{d}{dx}\left(x^4 \frac{d\varphi}{dx}\right) - 4x^2\varphi = \frac{1}{x}; \text{ а) } \frac{d\varphi}{dx}(1) = -3, \varphi(2) = 3;$$

$$\text{б) } \varphi(1) = -3, \varphi(1,9) = 3; \text{ в) } \frac{d\varphi}{dx}(1) - 2\varphi(1) = -10, 20\frac{d\varphi}{dx}(2) - \varphi(2) = 10.$$

$$\text{Вариант 10. } \frac{d}{dx}\left((e^x + 2)\frac{d\varphi}{dx}\right) = e^{x+2}; \text{ а) } \frac{d\varphi}{dx}(0) = -10, \varphi(2) = \frac{1}{2};$$

$$\text{б) } \varphi(0) = 10, \varphi(4) = \frac{1}{2}; \text{ в) } \frac{d\varphi}{dx}(0) + 2\varphi(0) = \frac{15}{2}, 20\frac{d\varphi}{dx}(2) + \varphi(2) = 10.$$

5. Решить краевую задачу методом Рунге. Сравнить с точным решением. Показать сходимость.

$$1. \frac{d^2\varphi}{dx^2} + 2\frac{d\varphi}{dx} + 1 = 0; \varphi(0) = 1, \frac{d\varphi}{dx}(1) = -1; [\varphi(0) = 1, \varphi(1) = -1].$$

$$2. x^2 \frac{d^2\varphi}{dx^2} + 2x \frac{d\varphi}{dx} - 2\varphi = 0; \varphi(1) = 1, \frac{d\varphi}{dx}(2) = -1; [\varphi(1) = 1, \varphi(2) = -1].$$

$$3. \frac{1}{4} \frac{d^2\varphi}{dx^2} + \varphi = 1; \varphi(0) + 2\frac{d\varphi}{dx}(0) = 0, \varphi(1) = 1; [\varphi(0) = 0, \varphi(1) = 1].$$

$$4. \frac{d^2\varphi}{dx^2} - \frac{d\varphi}{dx} = e^x; \varphi(0) = 0, \frac{d\varphi}{dx}(4) = -10; [\varphi(0) = 0, \varphi(4) = -10].$$

$$5. \frac{d}{dx}\left(e^{-x} \frac{d\varphi}{dx}\right) + e^x \varphi = 0; \varphi(0) = 0, \frac{d\varphi}{dx}(1) = 1; [\varphi(0) = 0, \varphi(1) = 1].$$

$$6. \frac{d^2\varphi}{dx^2} + \varphi = \frac{1}{\cos x}; \varphi(0) = 1, \frac{d\varphi}{dx}(1) = 1; [\varphi(0) = 1, \varphi(1) = 1].$$

$$7. \frac{d}{dx}\left(x^2 \frac{d\varphi}{dx}\right) - 2\varphi = 0; \varphi(1) = 0, \frac{d\varphi}{dx}(2) = 1; [\varphi(1) = 0, \varphi(2) = 1].$$

$$8. \frac{d}{dx}\left(e^{-x/2} \frac{d\varphi}{dx}\right) = 1 - x; \varphi(0) = 0, \frac{d\varphi}{dx}(4) = 2; [\varphi(0) = 0, \varphi(4) = 2].$$

$$9. \frac{d}{dx} \left(\frac{1}{1+x} \frac{d\varphi}{dx} \right) = -x; \varphi(0) + \frac{d\varphi}{dx}(0) = 1, \varphi(1) = 0; [\varphi(0) = 1, \varphi(1) = 0].$$

$$10. \frac{d}{dx} \left(\frac{1}{x} \frac{d\varphi}{dx} \right) = \frac{1}{1+x^2}; \varphi(1) + \frac{d\varphi}{dx}(1) = 0, \varphi(2) = 0; [\varphi(1) = 0, \varphi(2) = 0].$$

6. Найти приближенные решения задач с различными граничными условиями. Провести анализ сходимости, имея в виду, что аналитического решения у этих задач нет.

Вариант 1. $\frac{d}{dx} \left(e^{-x} \frac{d\varphi}{dx} \right) + \frac{x}{1+x^2} \varphi = 1-x$; а) $\frac{d\varphi}{dx}(1) = 1, \varphi(2) = 3$;

б) $\varphi(1) = 1, \varphi(2) = 3$; в) $\varphi(1) + \frac{d\varphi}{dx}(1) = 1, \varphi(2) - \frac{d\varphi}{dx}(2) = 2$.

Вариант 2. $\frac{d}{dx} \left((1+x^2) \frac{d\varphi}{dx} \right) - 8x\varphi = x^2 - x$; а) $\frac{d\varphi}{dx}(1) = 1, \varphi(3) = 0$;

б) $\varphi(1) = 1, \varphi(3) = 0$; в) $\varphi(1) + \frac{d\varphi}{dx}(1) = 1, \varphi(3) - \frac{d\varphi}{dx}(3) = 2$.

Вариант 3. $\frac{d^2\varphi}{dx^2} - (1+x^2)\varphi = 1 + \frac{2}{x+1}$; а) $\frac{d\varphi}{dx}(0) = 0, \varphi(3) = 0$;

б) $\varphi(0) = 4, \varphi(3) = 2$; в) $\frac{d\varphi}{dx}(0) + \varphi(0) = 0, \frac{d\varphi}{dx}(3) - \varphi(3) = 1$.

Вариант 4. $e^{-x} \frac{d^2\varphi}{dx^2} - e^{-x} \frac{d\varphi}{dx} = \frac{2}{\sqrt{2+x^2}}$; а) $\frac{d\varphi}{dx}(0) = 0, \varphi(1) = 1$;

б) $10\varphi(0) = 1, \varphi(1) = 1$; в) $\frac{d\varphi}{dx}(0) + \varphi(0) = 0, 20 \frac{d\varphi}{dx}(1) - \varphi(1) = 10$.

Вариант 5. $\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 16\varphi = \sin x$; а) $\frac{d\varphi}{dx} \left(\frac{\pi}{2} \right) = -10, \varphi(\pi) = \frac{1}{2}$;

б) $\varphi \left(\frac{\pi}{2} \right) = \frac{1}{5}, \varphi(\pi) = \frac{1}{2}$; в) $\varphi \left(\frac{\pi}{2} \right) + \frac{d\varphi}{dx} \left(\frac{\pi}{2} \right) = -10, \frac{d\varphi}{dx}(\pi) = 0$.

$$\frac{d}{dx} \left(p(x) \frac{dy}{dx} \right) - q(x)y = -f(x) \Leftrightarrow F[y(x)] = \frac{1}{2} \int_a^b (py'^2 + qy^2 - 2fy) dx + g(y(a), y(b))$$

Граничные условия	Добавка к функционалу $g(y(a), y(b))$	Аппроксимация y_m
$y(a)=y_0,$ $y(b)=y_1$	0	$(y_1 - y_0)(x - a)/(b - a) + y_0 +$ $+ (\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots + \alpha_m x^{m-1})(x - a)(x - b)$
$\frac{dy}{dx}(a) = \gamma_1, y(b) = y_1$	$\gamma_1 p(a) y(a)$	$y_1 + (\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots + \alpha_m x^{m-1})(x - b)$
$y(a) = y_0, \frac{dy}{dx}(b) = \gamma_2$	$-\gamma_2 p(b) y(b)$	$y_0 + (\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots + \alpha_m x^{m-1})(x - a)$
$\frac{dy}{dx}(a) = \gamma_1, \frac{dy}{dx}(b) = \gamma_2$	$\gamma_1 p(a) y(a) - \gamma_2 p(b) y(b)$	$\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots + \alpha_m x^{m-1}$
$y'(a) + \beta_1 y(a) = \gamma_1,$ $y'(b) + \beta_2 y(b) = \gamma_2$	$-\frac{\beta_1}{2} p(a) y(a)^2 + \gamma_1 p(a) y(a) +$ $+\frac{\beta_2}{2} p(b) y(b)^2 - \gamma_2 p(b) y(b)$	$\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \dots + \alpha_m x^{m-1}$

7. Решить краевую задачу методом Рунге, предварительно преобразовав дифференциальное уравнение 2-го порядка

$$y'' + s(x)y' + t(x)y = r(x)$$

к виду

$$d(p(x)dy/dx)/dx - q(x)y = -f(x),$$

путем умножения обеих частей на множитель $\mu(x) = e^{\int s(x)dx}$.

Вариант 1. $y'' + y' - e^{-x}y = 1$; $y(0) = 4$, $y(1) + 2y'(1) = 1$.

Вариант 2. $y'' - y' - e^x y = -x^2$; $y(0) = 6$, $y'(1) = 4$.

Вариант 3. $y'' + \frac{1}{x}y' - y = -\frac{1}{x} - 1$; $y'(1) = -1/8$, $y(2) = 4$.

Вариант 4. $y'' + \operatorname{ctg} x y' - \sin x y = -1/\sin^2 x$;

$$y'(\pi/4) - y(\pi/4) = 0, y(3\pi/4) = -1.$$

Вариант 5. $y'' + \frac{2x}{1+x^2}y' - \frac{1}{1+x^2}y = -(1+x^2)^3$;

$$y(0) = 0, 2y(1) + 3y'(1) = -1.$$

8. Решить задачу из примера 2, но с другой функцией плотности источника:

$$Q(x, y) = c \sin(m\pi x/a) \sin(n\pi y/b) \quad (m, n - \text{целые}).$$

9. Найти стационарное распределение температуры в прямоугольной пластине размеров $0 \leq x \leq a$, $0 \leq y \leq b$:

а) в отсутствие источников тепла;

б) при нагреве с плотностью $Q = 1$.

№ варианта	a	B	Граничные условия			
			$x = 0$	$x = a$	$y = 0$	$y = b$
1	1	1	$u = 0$	$\frac{\partial u}{\partial n} = 1$	$u = 0$	$\frac{\partial u}{\partial n} = -1$
2	0.8	1.5	$u = 1$	$u = 2$	$\frac{\partial u}{\partial n} = 0$	$\frac{\partial u}{\partial n} = -2$
3	1.25	1	$u = 0$	$u = 0$	$u = 0$	$\frac{\partial u}{\partial n} = 2$
4	2	1	$u = 5$	$u = -5$	$\frac{\partial u}{\partial n} + u = 0$	$\frac{\partial u}{\partial n} = -2$
5	0.9	1.6	$\frac{\partial u}{\partial n} = 1$	$\frac{\partial u}{\partial n} = -1$	$u = 0$	$\frac{\partial u}{\partial n} = -1$
6	1	2	$u = -y$	$\frac{\partial u}{\partial n} = 2$	$\frac{\partial u}{\partial n} = 0$	$\frac{\partial u}{\partial n} = 0$
7	1	1	$u = -y$	$\frac{\partial u}{\partial n} = 2$	$u = x$	$\frac{\partial u}{\partial n} = -2$
8	1	1.5	$u = 0$	$u = 1.5$	$u = 0$	$u = x$
9	1	0.6	$u = 0$	$u = 0$	$u = 0$	$\frac{\partial u}{\partial n} + u = 10$
10	1	2	$\frac{\partial u}{\partial n} = 1$	$\frac{\partial u}{\partial n} = 0$	$u = 10$	$\frac{\partial u}{\partial n} + u = 0$

10. Найти стационарное распределение температуры u в изображенном на рис. 8.4 секторе круга. Использовать показанные краевые условия.

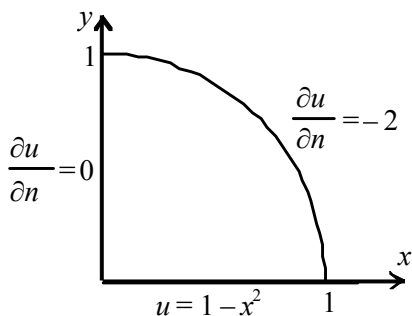


Рис. 8.4

11. Решить задачу Дирихле

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -4,$$

в области, заданной неравенством $x^2 + y^2 \leq 1$, с условием на границе Γ $u|_{\Gamma} = 2xy$.

12. Решить задачу Дирихле

$$\text{а) } \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -xe^y, \quad \text{б) } \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = xe^y,$$

в области, заданной неравенствами $0 \leq x \leq 1$, $0 \leq y \leq 1$, с условиями на границах:

$$u(x, 0) = x, \quad u(x, 1) = e \cdot x, \quad u(0, y) = 0, \quad u(1, y) = e^y.$$

9. МЕТОД ГАЛЁРКИНА

Основная идея проекционных методов основана на свойстве гильбертовых функциональных пространств, согласно которому, если проекции некоторой функции f на систему линейно независимых функций $\{W_i\}$ равны нулю, то такая функция $f \equiv 0$. За такую функцию, которая должна обращаться в ноль, принимают невязку – разность между левой и правой частью дифференциального уравнения и граничного условия (соответственно R_Ω и R_Γ). Поскольку в функциональном анализе проекция f на функцию g определяется как скалярное произведение (f, g) , то система уравнений имеет вид

$$(R_\Omega, W_i) = 0, \quad i = 1, \dots, m$$

или

$$(R_\Omega, W_i) + (R_\Gamma, \bar{W}_i) \Big|_\Gamma = 0$$

в зависимости от того учитывает ли аппроксимация неизвестной функции граничные условия или не учитывает.

Рассмотрим основные моменты реализации метода взвешенных невязок для уравнения

$$\frac{d^2 y}{dx^2} + s(x) \frac{dy}{dx} + t(x)y = r(x)$$

и граничных условий

$$\mu_1 y'(a) + \beta_1 y(a) = \gamma_1, \quad \mu_2 y'(b) + \beta_2 y(b) = \gamma_2.$$

Вначале строится аппроксимация с помощью базисных функций, как обычно, в виде суммы

$$y_m(x) = \sum_{i=1}^m \alpha_i N_i(x) \tag{9.1}$$

Для этой аппроксимации записываются невязки по отрезку $[a, b]$ и граничным точкам

$$R_\Omega = \frac{d^2 y_m}{dx^2} + s(x) \frac{dy_m}{dx} + t(x)y_m - r(x),$$

$$R_a = \mu_1 y'_m(a) + \beta_1 y_m(a) - \gamma_1,$$

$$R_b = \mu_2 y'_m(b) + \beta_2 y_m(b) - \gamma_2.$$

Выбираются системы линейно независимых функций – весовых функций W_i , $W_i^{(1)}$ и $W_i^{(2)}$. Эти системы произвольны, а, значит, в частности, могут быть каким-либо образом связаны между собой. Важный частный случай, когда $W_i = N_i$, $W_i^{(1)}(a) = \pm N_i(a)$, $W_i^{(2)}(b) = \pm N_i(b)$, т.е. весовые функции совпадают с аппроксимирующими, носит название метода Галеркина. Далее составляется система алгебраических уравнений

$$(R_\Omega, W_i) + [R_a W_i^{(1)}]_{x=a} + [R_b W_i^{(2)}]_{x=b} = 0, \quad i=1, \dots, m, \quad (9.2)$$

или в развернутом виде

$$\begin{aligned} & \int_a^b \left[\frac{d^2 y_m}{dx^2} + s(x) \frac{dy_m}{dx} + t(x) y_m - r(x) \right] W_i dx + \\ & + (\mu_1 y'_m(a) + \beta_1 y_m(a) - \gamma_1) W_i^{(1)}(a) + \\ & + (\mu_2 y'_m(b) + \beta_2 y_m(b) - \gamma_2) W_i^{(2)}(b) = 0. \end{aligned} \quad (9.2^*)$$

Эта система решается относительно параметров $\{\alpha_i\}$, которые при подстановке в (9.1) дают приближенное решение краевой задачи.

Возможен вариант, когда аппроксимация выбирается так, что изначально удовлетворяет одному или двум граничным условиям вне зависимости от того, какие значения принимают параметры $\{\alpha_i\}$. Эта аппроксимация, как правило, имеет вид

$$y_m(x) = \psi(x) + \sum_{i=1}^m \alpha_i N_i(x),$$

где $\psi(x)$ удовлетворяет граничным условиям задачи при $x=a$ и/или $x=b$, а функции $\{N_i\}$ – соответствующим однородным условиям. Тогда в уравнениях (9.2) одно или оба граничных слагаемых исчезают. Действительно, поскольку соответствующие граничные условия будут выполняться точно для функции y_m , то граничная невязка будет равна нулю.

Замечание. Невязку R_Ω (так же, как и R_Γ) можно рассматривать как функционал, поскольку она определяется выбором функции $y_m(x)$. Фактически метод Галеркина минимизирует $|R_\Omega|$ и его можно считать вариационным.

Пример. Решить краевую задачу методом Галеркина:

$$\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - \frac{2}{x}; \quad \varphi(1) = 0, \quad \varphi'(2) = 1.$$

Решение. Включая в систему уравнений взвешенных невязок дополнительно невязки для левой и правой граничных точек, в соответствии с (9.2) получим

$$\left(\frac{d}{dx} \left(x^2 \varphi' \right) - 2\varphi + 1 + \frac{2}{x}, W_i \right) + (\varphi - 0, W_i^{(1)}) \Big|_{x=1} + (\varphi' - 1, W_i^{(2)}) \Big|_{x=2} = 0,$$

или

$$\int_1^2 \left[\frac{d}{dx} \left(x^2 \frac{d\varphi_m}{dx} \right) - 2\varphi_m + 1 + \frac{2}{x} \right] W_i dx + \varphi_m(1) W_i^{(1)}(1) + \varphi'_m(2) W_i^{(2)}(2) = 0.$$

В качестве аппроксимации возьмем линейную комбинацию степенных функций:

$$\varphi_5(x) = \alpha_1 + \alpha_2 x + \alpha_3 x^2 + \alpha_4 x^3 + \alpha_5 x^4, \quad (9.3)$$

и составим уравнения взвешенных невязок, используя одинаковые (с точностью до знака) весовые функции W_i , $W_i^{(1)}$ и $W_i^{(2)}$: $1, x, x^2, x^3, x^4$. Как видно, такой выбор означает галеркинскую аппроксимацию.

Порядок решения в системе Maple

1. Зададим аппроксимацию и три невязки

```
> restart;
> z:=x->(a1+a2*x+a3*x^2+a4*x^3+a5*x^4);      #[1]
      z:=x -> a1+a2 x+a3 x^2+a4 x^3+a5 x^4
> R1:=diff(x^2*diff(z(x),x),x)-2*z(x)+1+2/x;
R2:=z(1);   R3:=D(z)(2)-1;
```

2. Определим пять весовых функций – по числу неизвестных параметров аппроксимации – соответственно для всех трех введенных невязок:

```
> W:=[1,x,x^2,x^3,x^4,x^5]; WL:=subs(x=1,W);
WR:=subs(x=2,W); # [2]
```

3. Построим пять уравнений как сумму взвешенных невязок:

```
> g1:=int(R1*W[1],x=1..2)+R2*WL[1]+R3*WR[1]=0;
> g2:=int(R1*W[2],x=1..2)+R2*WL[2]+R3*WR[2]=0;
> g3:=int(R1*W[3],x=1..2)+R2*WL[3]+R3*WR[3]=0;
> g4:=int(R1*W[4],x=1..2)+R2*WL[4]+R3*WR[4]=0;
> g5:=int(R1*W[5],x=1..2)+R2*WL[5]+R3*WR[5]=0;
```

4. Решим полученную систему линейных алгебраических уравнений:

```
> r:=solve({g1,g2,g3,g4,g5},{a1,a2,a3,a4,a5});
> r1:=evalf(r);
{a3=-23.167272,a4=8.348938,a5=-1.1531431,a2=30.546853,a1=-14.707363}
```

Это и есть оптимальные параметры, обеспечивающие наилучшее приближение в классе функций вида (9.3).

Решение данной задачи – это функция $\mathbf{z}(\mathbf{x})$, заданная в п. 1, с только что найденными числовыми параметрами $\mathbf{a1}$, $\mathbf{a2}$, $\mathbf{a3}$, $\mathbf{a4}$ и $\mathbf{a5}$, которые хранятся в переменной $\mathbf{r1}$ (или $\mathbf{r}$). Результат подстановки этих параметров в $\mathbf{z}(\mathbf{x})$ оформим в виде новой функции $\mathbf{phi}(\mathbf{x})$:

```
> phi:=unapply(subs(r1,z(x)),x);
phi:=x→-14.707363+30.546853x-23.167272x^2+8.348938x^3-1.1531431x^4
```

5. Получим точное решение задачи.

Данная краевая задача может быть решена точно аналитическими методами. Точное решение позволит оценить погрешность приближенного решения, полученного по методу Галеркина.

```
> st:=dsolve({diff(x^2*diff(y(x),x),x)-2*y(x)=-1-2*1/x,y(1)=0,D(y)(2)=1},y(x));
```

$$st := y(x) = -\frac{11}{5x^2} + \frac{7x}{10} + \frac{2+x}{2x}$$

```
> u:=unapply(subs(st,y(x)),x); # теперь точное решение
есть функция u(x)
```

6. Сравним точное и приближенное решения.

Сначала выведем результаты в отдельных равноотстоящих точках:

```
> for t from 1 by 0.25 to 2 do printf(`x=%g
z=%8.6g      u=%g\n`,t,phi(t),evalf(u(t))); od;
x=1          z=-.131987      u=0
x=1.25       z= .76857      u=.767
x=1.50       z=1.326433     u=1.238889
x=1.75       z=1.729728     u=1.578061
x=2.00       z=2.058469     u=1.85
```

Полученные по методу Галеркина значения не вполне хорошо согласуются с точным решением, что наглядно видно на графике:

```
> plot([phi(x),u(x)],x=1..2, color=[red,blue],
thickness=[2,2]);
```

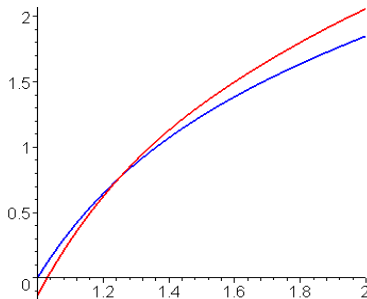


Рис. 9.1

7. Проверим сходимость.

Чтобы проверить адекватность решения, следует доказать сходимость метода. Для этого нужно выполнить все шаги метода с увеличенным на единицу числом параметров аппроксимации:

$$\varphi_6(x) = \alpha_1 + \alpha_2 x + \alpha_3 x^2 + \alpha_4 x^3 + \alpha_5 x^4 + \alpha_6 x^5.$$

Для этого строку [1] следует поменять на

```
> z:=x->(a1+a2*x+a3*x^2+a4*x^3+a5*x^4+a6*x^5);
```

Список весовых функций W можно оставить как есть, так как он уже содержит требуемую шестую функцию x^5 :

```
> W:=[1,x,x^2,x^3,x^4,x^5];
```

Далее в пункте 3 следует проделать вычисления для **g1**, **g2**, **g3**, **g4**, **g5** и добавить еще **g6**:

```
> g6:=int(R1*W[6],x=1..2)+R2*WL[6]+R3*WR[6]=0;
```

Решим полученную систему из шести уравнений

```
> r:=solve({g1,g2,g3,g4,g5,g6},{a1,a2,a3,a4,a5,a6});
```

Решение будет иметь вид

$$\phi := x \rightarrow -18.43098 + 47.0351x - 48.6239x^2 + 26.5966x^3 - 7.45583x^4 + 0.848104x^5$$

Построим, как и раньше, графики. Видно, что новое решение более точное, чем предыдущее. Этот же вывод следует и из поточечного сравнения решений.

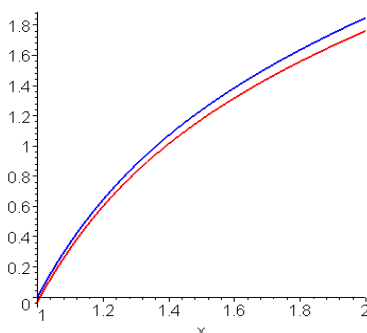


Рис. 9.2

Если взять семь параметров, т.е. аппроксимацию

$$\begin{aligned} \phi_7(x) = & \alpha_1 + \alpha_2 x + \alpha_3 x^2 + \alpha_4 x^3 + \\ & + \alpha_5 x^4 + \alpha_6 x^5 + \alpha_7 x^6, \end{aligned}$$

то обнаружим на порядок еще более точное решение. Итак, данный метод сходится. Для тщательного анализа сходимости можно вычислить норму ошибки

```
> evalf(int((phi(x)-u(x))^2,x=1..2))^(1/2);
```

и осуществить проверку граничных условий, поскольку они выполняются приближенно:

```
> D(phi)(2); # вычисление значения производной  $\phi'_m(2)$ 
```

```
> phi(1); # вычисление значения функции  $\phi_m(1)$ 
```

Таблица 9.1

	$m=5$	$m=6$	$m=7$
δ	0.1188	0.06313	0.00744
$\varphi'_m(2)$	1.164	0.961	1.010
$\varphi_m(1)$	-0.132	-0.0309	-0.0078

Замечание. По умолчанию Maple использует в вычислениях 10 значащих цифр. С увеличением порядка аппроксимации может потребоваться более высокая точность. Поэтому рекомендуется проверять расчеты с измененным параметром **Digits**. Например,

> Digits:=25; # устанавливает 25 значащих цифр
(вводится сразу за командой **restart**).

Другой подход. Рассмотрим вариант метода Галеркина, когда одно или два граничных условия удовлетворяются точно путем специального выбора аппроксимации.

Выберем аппроксимацию

$$\varphi_m(x) = (\alpha_1 + \alpha_2 x + \alpha_3 x^2 + \alpha_4 x^3 + \dots + \alpha_m x^{m-1})(x-1), \quad (9.4)$$

автоматически удовлетворяющую граничному условию 1-го рода $\varphi(1)=0$. Тогда невязка **R2** обнулится и формально ее присутствие в уравнениях **g1**, **g2**, ... ни на что не влияет. Таким образом, переопределив только функцию **z(x)** в соответствии с (9.4) в самом начале листинга, все остальные команды можно просто перезапустить. В общем случае такой подход, когда одно или два граничных условия учитываются точно через аппроксимацию, дает более высокую сходимость, чем в случае, когда граничные условия удовлетворяются приближенно путем введения соответствующих невязок. В табл. 9.2 показана сходимость нормы ошибки и производной функции $\varphi(x)$ в точке $x=2$.

Таблица 9.2

	$m=4$	$m=5$	$m=6$	$m=7$
δ	0.56170	0.11291	0.03008	0.007037
$\varphi'_m(2)$	1.504	0.899	1.027	0.994

Сходимость, пусть не намного, но выше, чем в предыдущем подходе.

Для удобства проведения анализа сходимости предлагается следующий листинг, в котором при изменении числа параметров аппроксимации достаточно только поменять значение параметра m :

```
> restart; Digits:=25;
> m:=5; a:=1: b:=2: # a и b – границы интервала изменения x
> a:=vector(m); g:=vector(m);
> z:=x->(sum(alpha[j]*x^(j-1), j=1..m)*(x-1)); # Для
# хранения параметров аппроксимации используем массив alpha
> R1:=diff(x^2*diff(z(x), x), x)-2*z(x)+1+2/x;
> R2:=z(a); R3:=D(z)(b)-1;
> W:=[1, x, x^2, x^3, x^4, x^5, x^6]; WL:=subs(x=a, W);
WR:=subs(x=b, W);
> for i from 1 to m do
g[i]:=int(R1*W[i], x=a..b)+R2*WL[i]+R3*WR[i]; od;
> r:=solve({g[k]$k=1..m}, {alpha[k]$k=1..m});
# Дальнейший текст почти полностью повторяет
# соответствующий фрагмент предыдущего варианта
> r1:=evalf(r);
> phi:=unapply(subs(r1, z(x)), x);
> st:=dsolve({diff(x^2*diff(y(x), x), x)-2*y(x)=-1-2*1/x,
y(1)=0, D(y)(2)=1}, y(x));
> u:=unapply(subs(st, y(x)), x);
> for t from a by 0.25 to b do
printf("x=%g z=%8.6g u=%g\n", t, phi(t), u(t)); od;
> plot([phi(x), u(x)], x=a..b, color=[red, blue],
thickness=[2, 2]);
> evalf(int((phi(x)-u(x))^2, x=a..b)^(1/2);
> D(phi)(b);
```

Упражнения

1. Произвести вычисления по той же схеме, что в примере, но с другими весовыми функциями W_i :

а) $W_i = 1, -x, x^2, -x^3, \dots$ б) $W_i = 1, x^2, x^4, x^6, \dots$

в) $W_i = 1, \operatorname{ch} x, \operatorname{ch} 2x, \operatorname{ch} 3x, \dots$ г) $W_i = e^x, x-1, e^{2x}, x^2-x, e^{3x}, x^3-x^2, \dots$

д) $W_i = 1, \sin \pi x, \sin 2\pi x, \sin 3\pi x, \dots$

е) $W_i = \delta(x - x_i)$, где x_i – узлы отрезка $[1, 2]$.

2. Выполнить расчет по приведенной схеме различных вариантов метода взвешенных невязок, отличающихся только знаками у весовых функций $W_i^{(1)}$ и $W_i^{(2)}$:

а) $W_i = N_i, W_i^{(1)} = -N_i, W_i^{(2)} = N_i$;

б) $W_i = N_i, W_i^{(1)} = N_i, W_i^{(2)} = -N_i$;

в) $W_i = N_i, W_i^{(1)} = -N_i, W_i^{(2)} = -N_i$.

В каком случае сходимость аппроксимации выше?

3. Предложить вариант решения задачи из примера с использованием аппроксимирующих базисных функций

$$N_1 = \sin \frac{\pi(x-1)}{2}, N_2 = \sin \frac{3\pi(x-1)}{2}, \dots, N_k = \sin \frac{(2k-1)\pi(x-1)}{2}$$

и любых подходящих весовых функций. Выполнить анализ сходимости.

4. Методом взвешенных невязок решить уравнение на указанном отрезке с указанными граничными условиями

$$\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - \frac{2}{x}; \quad \text{а) } x \in [0,5; 2,5]; \quad \varphi(0,5)=0, \varphi'(2,5)=1;$$

$$\text{б) } x \in [1; 2,5]; \quad \varphi'(1)=2, \varphi(2,5)=0.$$

5. Методом Галеркина решить краевые задачи из упражнения 4 раздела 8 («Метод Рунге»):

а) включая в уравнения невязки в обеих граничных точках;

б) удовлетворяя краевому условию 1-го рода выбором аппроксимации, а условиям 2-3-рода – вводом соответствующей невязки.

Сравнить с точным решением. Провести анализ сходимости аппроксимаций.

6*. С помощью метода Галёркина найти решение краевой задачи, не имеющей аналитического решения. Условия задач взять из упражнения 6 раздела 8. Показать сходимость.

10. МЕТОД КОНЕЧНЫХ ЭЛЕМЕНТОВ

Метод конечных элементов (МКЭ) на сегодняшний день является одним из наиболее эффективных методов решения краевых задач математической физики. Он обычно базируется на вариационной или проекционной формулировке и тем самым несет в себе преимущества методов Ритца или Галеркина. С другой стороны, подобно методу конечных разностей, решение здесь строится в виде каркаса значений в узлах сетки, а значит, он хорошо приспособлен для компьютерных вычислений.

Основное отличие метода конечных элементов от методов Ритца или Галеркина – в построении базисных функций. Теперь они имеют малый носитель, т.е. отличны от нуля только в окрестности некоторой геометрически выделенной точки – узла. В классическом методе конечных элементов область задачи разделяется на подобласти (конечные элементы). На каждой такой подобласти локально строится аппроксимация в виде интерполяционного полинома. Коэффициенты каждого полинома находятся из решения глобальной системы алгебраических уравнений, полученной путем применения процедуры Ритца или Галеркина и требования непрерывности решения.

Пример. Рассмотрим ту же задачу, что была решена ранее другими методами:

$$\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - \frac{2}{x}, \quad \varphi(1) = 0, \quad \varphi'(2) = 1.$$

Решение.

Разобьем отрезок $1 \leq x \leq 2$ на n равные части: $[x_k, x_{k+1}]$, $k = 1, 2, \dots, n$; $x_k \equiv 1 + (k-1)/n$. (Точки x_k называются узлами.) Каждую такую часть будем называть конечным элементом. На k -м элементе – отрезке $[x_k, x_{k+1}]$ – неизвестную функцию представим в виде линейной комбинации

$$\varphi^{(k)}(x) = \varphi(x_k) N_{1k}(x) + \varphi(x_{k+1}) N_{2k}(x), \quad x_k \leq x \leq x_{k+1}, \quad (10.1)$$

что соответствует случаю линейной интерполяции. Интерполирующие функции $N_{1k}(x)$ и $N_{2k}(x)$ (их называют *базисными функциями* или *функциями формы*) равны

$$N_{1k}(x) = \frac{x_{k+1} - x}{x_{k+1} - x_k}, \quad N_{2k}(x) = \frac{x - x_k}{x_{k+1} - x_k}. \quad (10.2)$$

Учитывая, что длины всех элементов одинаковы и равны $x_{k+1} - x_k = 1/n$, а также $x_k = 1 + (k-1)/n$, можно записать

$$N_{1k}(x) = n + k - nx, \quad N_{2k}(x) = nx - n - k + 1.$$

Итак, на всей области задачи $1 \leq x \leq 2$ неизвестная функция строится кусочным образом:

$$\hat{\varphi}(x) = \begin{cases} \varphi^{(1)}(x), & \text{если } x_1 \leq x \leq x_2, \\ \varphi^{(2)}(x), & \text{если } x_2 \leq x \leq x_3, \\ \dots\dots\dots \\ \varphi^{(n)}(x), & \text{если } x_n \leq x \leq x_{n+1}. \end{cases} \quad (10.3)$$

и рассматривается как пробная функция в методах Ритца или Галеркина. Проверьте, что эта функция принадлежит множеству S_0 , т.е. является непрерывной, но ее производная терпит разрыв 1-го рода в точках $x = x_k$, $k = 2, 3, \dots, n-1$. Важно отметить, что функция $\hat{\varphi}(x)$ однозначно определяется своими узловыми значениями $\varphi_k \equiv \varphi(x_k)$ по формулам (10.1)–(10.3), и поэтому метод конечных элементов сводится к нахождению этих значений.

Для данного дифференциального уравнения существует вариационная формулировка. Поэтому для решения воспользуемся методом Ритца. Подставляя в функционал задачи

$$F[\varphi(x)] = \frac{1}{2} \int_1^2 (x^2 \varphi'^2 + 2\varphi^2 - 2(1 + 2/x)\varphi) dx - 4\varphi(2)$$

кусочную аппроксимацию (10.3), получим

$$\begin{aligned}
F[\hat{\varphi}(x)] = & \frac{1}{2} \int_{x_1}^{x_2} \left[x^2 (\varphi^{(1)})'^2 + 2(\varphi^{(1)})^2 - 2(1 + 2/x)(\varphi^{(1)}) \right] dx + \\
& + \int_{x_2}^{x_3} \left[x^2 (\varphi^{(2)})'^2 + 2(\varphi^{(2)})^2 - 2(1 + 2/x)(\varphi^{(2)}) \right] dx + \\
& + \dots + \frac{1}{2} \int_{x_n}^{x_{n+1}} \left[x^2 (\varphi^{(n)})'^2 + 2(\varphi^{(n)})^2 - 2(1 + 2/x)(\varphi^{(n)}) \right] dx - 4\varphi^{(n)}, \quad (10.4)
\end{aligned}$$

т.е. сумму функционалов отдельных конечных элементов.

Учитывая конкретный вид функции $\varphi^{(k)}(x)$ из (10.1) с базисными функциями (10.2), в этом выражении можно выполнить интегрирование по конечным элементам (при этом считая $\varphi_k \equiv \varphi(x_k)$ постоянными числами). В итоге функционал превратится в функцию многих переменных $\Phi(\varphi_2, \dots, \varphi_{n+1})$. Числа φ_k теперь выступают как параметры. Обратите внимание, что в списке аргументов Φ отсутствует параметр $\varphi_1 = \varphi(1)$, поскольку он известен из соответствующего граничного условия Дирихле при $x=1$ и должен быть зафиксирован. Далее, как обычно необходимое условие экстремума $\Phi(\varphi_2, \dots, \varphi_{n+1})$ приводит к системе алгебраических уравнений

$$\frac{\partial \Phi}{\partial \varphi_k} = 0, \quad k = 2, \dots, n.$$

Решая эту систему, найдем требуемые значения параметров $\{\varphi_2^*, \varphi_3^*, \dots, \varphi_{n+1}^*\}$ и тем самым приближенное решение краевой задачи. В отличие от метода конечных разностей, МКЭ позволяет непосредственно найти значения функции во всех точках отрезка $1 \leq x \leq 2$ (а не только в узловых) согласно представлению (10.3).

Порядок решения в системе Maple.

1. В соответствии с МКЭ, разбиваем отрезок на 4 конечных элемента, и на каждом из них строим аппроксимацию:

> restart;

```

> N1:=[5-4.*x,6-4*x,7-4*x,8-4*x];
  N2:=[4.*x-4.,4*x-5,4*x-6,4*x-7];
> z:=unapply([y1*N1[1]+y2*N2[1],y2*N1[2]+y3*N2[2],
y3*N1[3]+y4*N2[3],y4*N1[4]+y5*N2[4]],x);
  z:=x->[y1(5-4x)+y2(4x-4),y2(6-4x)+y3(4x-5),
  y3(7-4x)+y4(4x-6),y4(8-4x)+y5(4x-7)]

```

Таким образом, ввели функцию $z(x)$ в виде вектора, т.е. вектор-функцию, i -й элемент которой есть аппроксимация на i -м элементе. Обращение к ней дополнительно содержит номер конечного элемента в квадратных скобках: например, $z(x)[2]$ означает аппроксимацию второго элемента и, следовательно, будет замещаться выражением $y_2(6-4x)+y_3(4x-5)$.

2. Для каждого конечного элемента составляем функционалы (вид функционалов – как в методе Ритца). Для 1-го элемента

```

> f1:=(1/2)*int(x^2*diff(z(x)[1],x)^2+
2*expand(z(x)[1]^2)-2*(1+2./x)*z(x)[1],x=1..1.25);

```

Для второго элемента

```

> f2:=(1/2)*int(x^2*diff(z(x)[2],x)^2+
2*expand(z(x)[2]^2)-2*(1+2./x)*z(x)[2],x=1.25..1.5);

```

Для третьего элемента

```

> f3:=(1/2)*int(x^2*diff(z(x)[3],x)^2+
2*expand(z(x)[3]^2)-2*(1+2./x)*z(x)[3],x=1.5..1.75);

```

Для 4-го элемента в функционале присутствует граничная добавка $-4 * z(2)[4]$, поскольку этот элемент примыкает к границе $x=2$:

```

> f4:=(1/2)*int(x^2*diff(z(x)[4],x)^2+
2*expand((z(x)[4])^2)-2*(1+2./x)*z(x)[4],
x=1.75..2)-4*z(2)[4];

```

3. Из условия, что суммарный функционал должен достигать экстремума, строим систему уравнений. При этом, заметьте, что первое уравнение, соответствующее узлу с номером 1, заменено на $y_1=0$, поскольку в левой граничной точке задано условие 1-го рода $\varphi(1)=0$.

```

> F:=f1+f2+f3+f4;

```

```
> eqns:={y1=0,diff(F,y2),diff(F,y3),diff(F,y4),
diff(F,y5)};
eqns := {-14.00000000 y4 + 14.25000000 y5 - 4.255560503,
-5.000000000 y1 + 13.00000000 y2 - 0.6527102710 - 7.500000000 y3,
-7.500000000 y2 + 18.50000000 y3 - 0.5848939496 - 10.50000000 y4,
-10.50000000 y3 + 25.00000000 y4 - 0.5366941241 - 14.00000000 y5, y1 = 0}
```

4. Находим ее решение:

```
> r:=solve(eqns,{y1,y2,y3,y4,y5});
> r1:=evalf(r);
r1 := {y1 = 0, y5 = 1.8374183725364, y2 = 0.75877819934262, y4 = 1.5662608075276,
y3 = 1.2281875093920}
```

Эти числа и есть значения искомой функции в узловых точках. Чтобы сравнить их с точными значениями, воспользуемся набором команд:

```
> x:=[1,1.25,1.5,1.75,2]:
Y:=subs(r1,[y1,y2,y3,y4,y5]);
> st:=dsolve({diff(x^2*diff(y(x),x),x)-2*y(x)=
-(1+2.*1/x),y(1)=0,D(y)(2)=1},y(x));
> u:=unapply(subs(st,y(x)),x);
> for i from 1 to 5 do printf(`x=%6.3g z=%6.4g
u=%g\n`,X[i],Y[i],u(X[i])); od;

x= 1.000 z=0.0000 u=0
x= 1.250 z= .7587 u=.767
x= 1.500 z=1.2281 u=1.238888
x= 1.750 z=1.5662 u=1.578061
x= 2.000 z=1.8374 u=1.85
```

5. Решение конечно-элементной задачи как непрерывная функция однозначно определяется своими узловыми значениями. Она строится кусочным образом по всем конечным элементам, причем для задания функции внутри конечного элемента берутся узловые значения только этого элемента и используется формула $y^e = \sum_i y_i N_i^e(x)$.

С учетом сказанного определим искомую функцию **phi**, являющуюся решением нашей задачи. Это можно сделать с помощью встроенной функции **piecewise**:

```
> phi:=x->piecewise(x<X[2],subs(r1,z(x)[1]),
x<X[3],subs(r1,z(x)[2]),x<X[4],subs(r1,z(x)[3]),
subs(r1,z(x)[4])): evalf(phi(x),4);
```

Понимается эта запись так: если значение x попадает в интервал до $X[2]$, т.е. принадлежит первому конечному элементу, то функции $\phi(x)$ присваивается $z(x)[1]$ – первая функция вектора $z(x)$, если x принадлежит отрезку $[X[2], X[3]]$ (второй конечный элемент), присваивается $z(x)[2]$ – вторая функция списка $z[x]$ и т.д. Теперь к этой функции можно обращаться как к обычной функции системы Maple. Например, построить график:

```
> g4:=plot(phi(x),x=X[1]..X[5],color=magenta,
thickness=2):
> g2:=plot([[X[i],Y[i]]$i=1..5],x=X[1]..X[5],
style=POINT,symbol=CIRCLE,color=black):
plots[display](g2,g4);
```

Чтобы этот график сравнить с точным решением, следует задать:

```
> plot([phi(x),u(x)],x=X[1]..X[5],color=[red,blue],
thickness=[2,2]);
```

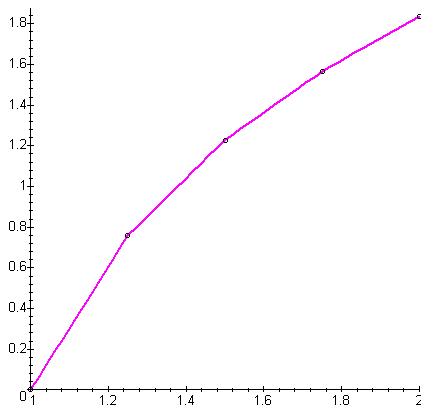
Найти норму погрешности аппроксимации можно так:

```
> int((phi(x)-u(x))^2,x=1..2)^(1/2);
0.02838184159
```

6. Изобразим на графике в виде точек полученные конечно-элементные значения, для чего на координатной плоскости отдельно нанесем точки, а затем совместим два построения:

```
> g2:=plot([[X[i],Y[i]]$i=1..5],x=X[1]..X[5],
style=POINT,symbol=CIRCLE,color=black):
plots[display](g2,g4);
```

Вид этого рисунка наводит на мысль, что искомую функцию можно получить посредством обычной локальной интерполяции. В данном случае имеем кусочно-линейную интерполяцию (используются линейные конечные элементы). А это значит, что две соседние точки соединяются прямой линией.



7. Чтобы доказать сходимость метода, надо провести несколько вычислений с большим числом конечных элементов. Это удобно сделать с помощью следующего фрагмента

```
> restart;
> a:=1; b:=2; n:=8; h:=(b-a)/n; m:=n; #[1]
> N1:=x->seq(a/h-x/h+i,i=1..n);
  N2:=x->seq(x/h-a/h-i+1,i=1..n); #[2]
> z:=x->seq(y[i-1]*N1(x)[i]+y[i]*N2(x)[i],i=1..n); #[3]
> F:=0; for i from 1 to n do
F:=F+(1/2)*int(x^2*diff(z(x)[i],x)^2+
2*expand((z(x)[i]^2))-2*(1+2./x)*z(x)[i],
x=a+(i-1)*h..a+i*h);od: #[4]
F:=F-4*z(2)[n]; #[5]
> eqts:=[seq(diff(F,y[i])=0,i=0..m)];
eqts[1]:=y[0]=0; #[6]
> rez:=solve({seq(eqts[i],i=1..m+1)},
{seq(y[i],i=0..m)}); #[7]
> X:=[seq(a+i*h,i=0..m)];
  Y:=subs(rez,[seq(y[i],i=0..m)]); #[8]
> readlib(spline);
> f1:=spline(X,Y,x,linear);
> g4:=plot(f1,x=a..b,color=magenta,thickness=2);
> g2:=plot([X[j],Y[j]]$j=1..m+1,x=X[1]..X[m],
style=POINT,symbol=CIRCLE,color=black):
```

```

> st:=dsolve({diff( x^2*diff(y(x),x) , x)-2*y(x)=
- (1+2.*1/x), y(1)=0,D(y)(2)=1},y(x));
> u:=unapply(subs(st,y(x)),x);
> g5:=plot(u(x),x=a..b,color=blue,thickness=2):
> plots[display](g2,g4,g5);
> delta:=evalf(int((f1-u(x))^2,x=a..b)^(1/2); #[9]

```

Комментарий. Строка 1: задание переменных **a**, **b** (начало и конец интервала), **n** (число элементов), **h** (длина конечного элемента). Строка 2: ввод базисных функций для линейных элементов; **N1** и **N2** – последовательности из **n** базисных функций, соответствующих левым и правым узлам конечных элементов. Строка 3: определение последовательности **z[x]**, включающего аппроксимирующие выражения для всех конечных элементов. Строки 4: вычисление функционала через цикл по всем элементам. Строка 5: учет граничного условия 2-го рода в виде добавки к функционалу, соответствующей правой граничной точке. Строка 6: формирование системы уравнений; их количество равно $(n+1)$ – по числу узлов. Здесь же учет граничного условия 1-го рода в левой граничной точке. Строка 7: решение полученной системы; результат записывается в **rez**. Строка 8 и далее: построение графиков, сравнение с точным решением. Строка 9: вычисление нормы ошибки решения.

Для проверки сходимости нужно менять всего один параметр в приведенном фрагменте – число **n**. В результате получим таблицу:

	$n = 4$	$n = 8$	$n = 16$
δ	0.02838184159	0.007361378471	0.001858766257

8. Точность МКЭ повышается не только с ростом числа элементов (или узлов), но и с увеличением порядка интерполяции на каждом конечном элементе. Найдём решение задачи, используя не линейные элементы, как раньше, а квадратичные. Это означает, что неизвестная функция внутри конечного элемента строится как квадратичный полином

$$\varphi^e = C_1^e x^2 + C_2^e x + C_3^e. \quad (10.5)$$

Самый простой способ – за такой полином взять многочлен Лагранжа

$$\varphi = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}\varphi_0 + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}\varphi_1 + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}\varphi_2,$$

где x_0, x_1 и x_2 – координаты трех узлов элемента – двух граничных (x_0 и x_2) и одной внутренней (x_1) точек; $\varphi_0, \varphi_1, \varphi_2$ – значения функции в соответствующих узлах. Очевидно, многочлен Лагранжа обеспечивает непрерывность функции при переходе от элемента к элементу.

Сравнивая последнее выражение с общим видом аппроксимации $\varphi^e = \sum_i \varphi_i N_i^e(x)$, получаем на конечном элементе e три базисные функции

$$N_1^e(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}, \quad N_2^e(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)},$$

$$N_3^e(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)},$$

каждая из которых приписывается одному из узлов x_0, x_1 и x_2 элемента по правилу $N_i^e(x_j) = \delta_{ij}$ (т.е. для «своего» узла эта функция равна единице, а для остальных – нулю). Эти базисные функции меняются от элемента к элементу. Поэтому при программировании придется вводить $3n$ таких функций. Это легко реализуется, если все элементы имеют одинаковую длину. К примеру,

> **restart;**

> **a:=1;b:=2;n:=4;h:=(b-a)/n; m:=2*n;**

> **N1:=x->seq(subs(x0=a+(i-1)*h,x1=a+(i-1)*h+h/2, x2=a+i*h,(x-x1)*(x-x2)/((x0-x1)*(x0-x2))),i=1..n):**

Аналогично определяются две другие базисные функции:

> **N2:=x->seq(subs(x0=a+(i-1)*h,x1=a+(i-1)*h+h/2, x2=a+i*h,(x-x0)*(x-x2)/((x1-x0)*(x1-x2))),i=1..n):**


```
> N3:=x->seq(subs(x0=a+(i-1)*h,x1=a+(i-1)*h+h/2,
x2=a+i*h,(x-x0)*(x-x1)/((x2-x0)*(x2-x1))),i=1..n);
```

Есть другой, более универсальный, способ задания базисных функций. Он основан на вычислении коэффициентов $\{C_i^e\}$ в выражении типа (10.5) из каких-либо условий, например, что функция в узлах $\{x_i\}$ должна принимать значения $\{\phi_i\}$. Кстати, эти условия могут быть наложены и на производные, и здесь мы приходим к эрмитовой интерполяции, сплайн-интерполяции и др. Разберите следующий листинг программы, реализующий ввод этим универсальным способом базисных функций интерполяции Лагранжа 2-го порядка. (Кстати, этот способ лучше приспособлен для случая неравноотстоящих узлов.)

```
> restart;
> a:=1;b:=2;n:=4;h:=(b-a)/n; m:=2*n;
> wt:=x->c1*x^2+c2*x+c3;
> wr:=solve({wt(xi)=w1,wt(xi+h/2)=w2,wt(xi+h)=w3},
{c1,c2,c3});
> uf:=collect(subs(wr,wt(x)),{w1,w2,w3});
uv:=[diff(uf,w1),diff(uf,w2),diff(uf,w3)];
> N1:=x->seq(subs(xi=a+(i-1)*h,uv[1]),i=1..n):
N2:=x->seq(subs(xi=a+(i-1)*h,uv[2]),i=1..n):
N3:=x->seq(subs(xi=a+(i-1)*h,uv[3]),i=1..n):
```

Следующий шаг – задание аппроксимации неизвестной функции. Задавая узловые величины через массив **y**, имеем

```
> z:=x->seq(y[2*i-2]*N1(x)[i]+y[2*i-1]*N2(x)[i]+
y[2*i]*N3(x)[i],i=1..n);
```

Дальнейшие шаги почти полностью повторяют строки [4]–[8] листинга из пункта 7. Следует только учесть, что переменная **m** – число узлов минус один – здесь определяется иначе.

```
> F:=0; for i from 1 to n do
F:=F+(1/2)*int(x^2*diff(z(x)[i],x)^2+
2*expand((z(x)[i]^2))-2*(1+2./x)*z(x)[i],
x=a+(i-1)*h..a+i*h);od:
> F:=F-4*z(2)[n]:
```

[4]
[5]

```

> eqts:=[seq(diff(F,y[i])=0,i=0..m)]:
eqts[1]:=y[0]=0: # [6]
> rez:=solve({seq(eqts[i],i=1..m+1)},
{seq(y[i],i=0..m)}); # [7]
> X:=[seq(a+i*h/2,i=0..m)];
Y:=subs(rez,[seq(y[i],i=0..m)]); # [8]

```

Изображение вычисленных значений в узловых точках:

```

> g2:=plot([X[j],Y[j]]$j=1..m+1,x=X[1]..X[m],
style=POINT,symbol=CIRCLE,color=black):

```

Нахождение точного решения:

```

> st:=dsolve({diff(x^2*diff(y(x),x),x)-2*y(x)=
-(1+2.*1/x),y(1)=0,D(y)(2)=1},y(x));
> u:=unapply(subs(st,y(x)),x);
> g5:=plot(u(x),x=a..b,color=blue,thickness=2):

```

Представление конечно-элементного решения в виде кусочной функции:

```

> v1:=x<=X[1],0;
for k from 1 to n do
v1:=v1,x<=X[2*k+1],subs(rez,z(x)[k]); od:
> z2:=piecewise(v1,0):
> g4:=plot(z2,x=a..b):

```

Совмещение всех построений:

```

> plots[display](g2,g4,g5);

```

Вычисление нормы ошибки решения:

```

> Digits:=20;
> evalf(int((z2-u(x))^2,x=a..b))^(1./2); # [9]

```

Заметим, данный листинг позволяет проанализировать сходимость МКЭ в рамках квадратичной аппроксимации при увеличении числа элементов (нужно менять только **n**).

Приведенные способы реализации МКЭ в системе следует рассматривать лишь в контексте учебно-методического использования. Они весьма наглядны, хорошо иллюстрируют сущность метода, но с практической точки зрения малопригодны. Действительно, эти способы основаны на непосред-

венной минимизации функционала, что требует применения символьных методов алгебры, интегрирования и дифференцирования. Эти методы – удел компьютерной математики – развиваются не так давно, очень сложны и пока еще недостаточно совершенны. Обратите внимание, какой громоздкий вид имеет функционал для четырех элементов, как резко он усложняется для большего числа элементов. Система при этом должна проинтегрировать выражение, считая неизвестные параметры постоянными, а затем произвести дифференцирование по этим параметрам. Такая операция становится вообще невозможной, если интегралы или очень сложны, или не берутся в аналитическом виде. Сочетание же численных и символьных вычислений система не поддерживает. Стоит ли говорить, как усложняются и замедляются все операции, если в них задействуются еще и булевы функции. Понятно, что в специализированных программных реализациях алгоритм должен строиться без привлечения средств символьной математики – только с использованием численных методов. С этих позиций формализация задачи доводится до уровня построения системы линейных алгебраических уравнений, причем такая система формируется отдельно для каждого конечного элемента и добавляется в общую (глобальную) систему. Вычисление коэффициентов каждой локальной системы сводится к численному интегрированию по области конечного элемента, которое хорошо разработано в современной вычислительной математике. Также успешно развиты методы решения систем линейных алгебраических уравнений, в том числе и специального вида, какими являются системы в МКЭ. (Для матриц систем, возникающих в МКЭ, характерны высокий порядок, симметричность, положительная определенность и сильная разреженность).

В качестве примера обратимся к краевой задаче 3-го рода

$$\frac{d}{dx} \left(p(x) \frac{dy}{dx} \right) - q(x)y = -f(x), \quad a \leq x \leq b, \quad (10.6)$$

$$p(a)y'(a) + \beta_0 y(a) = \gamma_0, \quad p(b)y'(b) + \beta_1 y(b) = \gamma_1.$$

Если в эквивалентной вариационной задаче воспользоваться свойством аддитивностью функционала

$$F[y(x)] = \sum_e F_e[y_e(x)],$$

где

$$F_e[y_e(x)] = \frac{1}{2} \int_{\Omega_e} (py_e'^2 + qy_e^2 - 2fy_e) dx + \left[\frac{\beta_2}{2} y_e^2 - \gamma_2 y_e \right]_{x=b} - \left[\frac{\beta_1}{2} y_e^2 - \gamma_1 y_e \right]_{x=a},$$

то, аппроксимируя на каждом конечном элементе функцию y_e выражением $\hat{y}_e = \sum_i Y_i^e N_i^e(x)$, путем процедуры Ритца придем к системе

$$\sum_j S_{ij} Y_j = B_i,$$

$$\mathbf{S} = \sum_e \mathbf{C}_e \mathbf{S}^e \mathbf{C}_e^T, \quad \mathbf{B} = \sum_e \mathbf{C}_e \mathbf{B}^e \mathbf{C}_e^T.$$

$\mathbf{C}_e$ – матрица, устанавливающая связь между глобальной и локальной нумерацией узлов (состоит из нулей и единиц; реально обычно в программе не хранится); $\mathbf{S}$, $\mathbf{B}$ – матрица и вектор правых частей локальной системы уравнений, определяемые на каждом конечном элементе по формулам

$$S_{ij}^e = \int_{\Omega_e} p(x) N_i^e(x) N_j^e(x) dx + \left[\beta_1 N_i^e N_j^e \right]_{x=b} - \left[\beta_0 N_i^e N_j^e \right]_{x=a},$$

$$B_i^e = \int_{\Omega_e} f(x) N_i^e(x) dx + \left[\gamma_1 N_i^e \right]_{x=b} - \left[\gamma_0 N_i^e \right]_{x=a}.$$

Замечания. 1) Во всех интегралах интегрирование проводится по области конечного элемента с номером e ; выражения в квадратных скобках учитываются только в том случае, если элемент примыкает к соответствующим граничным точкам.

2) Граничное условие 1-го рода легко учесть, модифицируя конечную систему уравнений. Соответствующее граничное слагаемое при этом в функционале, конечно же, должно отсутствовать.

Разберите следующий листинг программы, реализующий метод конечных элементов для уравнения (10.6) с граничными условиями 1–3 рода

$$\alpha_0 y'(a) + \beta_0 y(a) = \gamma_0, \quad \alpha_1 y'(b) + \beta_1 y(b) = \gamma_1.$$

```
> restart; Digits:=15;
> n:=10; approx_order:=2;a:=0: b:=1:                                     #1
> p:=x->exp(-x): q:=x->-exp(-x): f:=x->4:                               #2
> alpha0:=-5: beta0:=1: gamma0:=-2:                                     #3
> alpha1:=1: beta1:=1: gamma1:=2:                                       #4
> npel:=`if`(approx_order=2,3,2):
  nd:=n+(npel-2)*n+1:                                                  #5
> h:=(b-a)/n/`if`(approx_order=2,2,1);                                #6
> Bound:=[ [0]$k=1..nd ];                                              #7
> if alpha0=0 then Bound[1]:=[1,gamma0/beta0];fi; #8
> if alpha1=0 then Bound[nd]:=[1,gamma1/beta1];fi; #9
> if (alpha0<>0) then
Bound[1]:=`if`(beta0=0, [2,gamma0/alpha0],
[3,beta0/alpha0,gamma0/alpha0]);fi;                                     #10
> if (alpha1<>0) then
Bound[nd]:=`if`(beta1=0, [2,gamma1/alpha1],
[3,beta1/alpha1,gamma1/alpha1]);fi;                                     #11
> NLg:=[ (x1-x)/(x1-x0), (x-x0)/(x1-x0) ];                             #12
> if approx_order=2 then
  NLg:=[ (x-x1)*(x-x2)/((x0-x1)*(x0-x2)),
(x-x0)*(x-x2)/((x1-x0)*(x1-x2)),
(x-x0)*(x-x1)/((x2-x0)*(x2-x1)) ]; fi;                               #13
```

```

> NN1:=[ ( (npel-1) * (k-1) + m $ m=1..npel) $ k=1..n]; #14
> X:=[a+ (k-1) * h $ k=1..nd]; #15
> S:=[ [0. $ m=1..nd] $ k=1..nd]; #16
> B:=[0. $ k=1..nd]; #17
> for el from 1 to n do #18
  xm:=[seq(X[ NN1[el,i]], i=1..npel), 0];
  W:=subs(x0=xm[1], x1=xm[2], x2=xm[3], NLg);
  Wd:=diff(W,x);
  for i from 1 to npel do
    for j from 1 to npel do
      S[NN1[el,i], NN1[el,j]]:=evalf(S[NN1[el,i], NN1[el,j]]+
int(p(x)*Wd[i]*Wd[j]+q(x)*W[i]*W[j], x=xm[1]..xm[npel]));
      od;
    ni:=NN1[el,i];
    sig:='if' (ni=nd, 1, -1);
    if (Bound[ni,1]=3) then S[ni,ni]:=evalf(S[ni,ni]+
sig*p(xm[i])*Bound[ni,2]*(subs(x=xm[i], W[i]))^2); fi;
    B[ni]:=evalf(B[ni]+evalf(int(f(x)*W[i],
x=xm[1]..xm[npel]))+
'if' (Bound[ni,1]=2,
sig*p(xm[i])*Bound[ni,2]*subs(x=xm[i], W[i]), 0)+
'if' (Bound[ni,1]=3,
sig*p(xm[i])*Bound[ni,3]*subs(x=xm[i], W[i]), 0));
    od;
  od: #18'
> if Bound[1,1]=1 then
  B[1]:=Bound[1,2]; S[1,1]:=1; #19
  for i from 2 to nd do
    B[i]:=B[i]-Bound[1,2]*S[i,1];
    S[i,1]:=0; S[1,i]:=0; od;
  fi;

```

```

> if Bound[nd,1]=1 then
  B[nd]:=Bound[nd,2]; S[nd,nd]:=1;

  for i from 1 to nd-1 do
    B[i]:=B[i]-Bound[nd,2]*S[i,nd];
    S[i,nd]:=0; S[nd,i]:=0; od;

  fi;
#19'

> evalm(B);

> evalm(S);

> Y:=linalg[linsolve](S,B);
#20

> g2:=plot([[X['k'],Y['k']]$'k'=1..nd],x=X[1]..X[nd],
style=POINT,symbol=CIRCLE,color=black);

> st:=dsolve({diff(p(x)*diff(y(x),x),x)-
q(x)*y(x)=-f(x),alpha0*D(y)(a)+beta0*y(a)=gamma0,
alpha1*D(y)(b)+beta1*y(b)=gamma1},y(x));

> u:=unapply(subs(st,y(x)),x);

> g5:=plot(u(x),x=a..b,color=blue,thickness=2);

> plots[display](g2,g5);

```

Комментарий. Строка 1. Задание коэффициентов дифференциального уравнения.

Строки 1–4. Входные данные программы: **n** – число конечных элементов, **approx_order** –порядок интерполяции на каждом элементе (возможные значения: 1 и 2); **a,b** – границы интервала; **p,q,f** – коэффициенты дифференциального уравнения; **alpha0,beta0,gamma0, alpha1,beta1,gamma1** – коэффициенты граничных условий.

Строка 5. Переменная **nd** задает число узлов, **npel** – число узлов на конечный элемент.

Строки 7–11. Формирование массива **Bound**, хранящем информацию о граничных условиях. Число **Bound[i,1]** сообщает о наличии граничного условия в узле с номером *i*; если

оно равно 0, то граничного условия нет, если 1, 2 или 3, то это граничное условие 1-го рода, 2-го или 3-го рода соответственно.

Строки 12–13. Базисные функции.

Строка 14. В массиве **nn1** хранятся списки узлов конечных элементов.

Строка 15. Массив **x** состоит из координат узлов.

Строки 16–17. Инициализация матрицы системы линейных алгебраических уравнений **S** и вектора правых частей **b** (обнуление их значений).

Строки 18–18'. Цикл, перебирающий конечные элементы (цикл по переменной **e1**). Тело цикла включает вычисление двух или трех базисных функций конечного элемента с номером **e1**; цикл по элементам локальной матрицы S^e ; вычисление коэффициентов этой матрицы и добавление их в соответствующие позиции глобальной матрицы **S**; то же для вектора правых частей **b**.

Строки 19–19'. Учет граничных условий 1-го рода путем модификации глобальной системы. Уравнение с номером i , соответствующим узлу с указанным условием, заменяется соотношением, задающим условие явно: $y(x_i) = y_0$.

Строка 20. Решение системы линейных алгебраических уравнений $SY = b$. Получение вектора **Y**, состоящего из узловых значений – решения конечноэлементной задачи.

Далее – восстановление непрерывного решения и его сравнение с точным решением краевой задачи.

Упражнения

1. Решить задачу из примера

$$(x^2 \varphi')' - 2\varphi = -1 - 2/x, \quad \varphi(1) = 0, \quad \varphi'(2) = 1$$

для разных чисел конечных элементов ($n = 2 \div 6$ и более). Показать сходимость решения, исследуя поведение функционала, аппроксимации и граничного значения $\varphi'(2)$.

2. Решить ту же задачу, используя квадратичные элементы Лагранжа. Показать сходимость решения, исследуя поведение функционала, аппроксимации и граничного значения $\varphi'(2)$ для разного количества конечных элементов (начать с $n=2$). Провести также анализ сходимости решения в аспекте сравнения задач с одинаковым полным числом узлов, но с разными типами конечных элементов.

3. Решить методом конечных элементов уравнение

$$\frac{d}{dx} \left(x^2 \frac{d\varphi}{dx} \right) - 2\varphi = -1 - \ln x^2$$

- а) на отрезке $1 \leq x \leq 2$ с краевыми условиями $\varphi(1) = 2, \varphi'(2) = -1$;
- б) на отрезке $1 \leq x \leq 2$ с краевыми условиями $\varphi'(1) = 0, \varphi(2) = 1$;
- в) на отрезке $1 \leq x \leq 2$ с краевыми условиями $\varphi'(1) = 1, \varphi'(2) = 0$;
- г) на отрезке $0,5 \leq x \leq 2,5$ с краевыми условиями $\varphi(0,5) = 0, \varphi'(2,5) = 1$;
- д) на отрезке $1 \leq x \leq 2,5$ с краевыми условиями $\varphi'(1) = 2, \varphi(2,5) = 0$;
- е) на отрезке $0,6 \leq x \leq 3,2$ с краевыми условиями $\varphi(0,6) = 2, \varphi(3,2) = 0$.

4. Решить методом конечных элементов краевые задачи из упражнения 4 раздела 8 «Решение краевых задач для обыкновенных дифференциальных уравнений методом Рунге». Использовать линейные элементы Лагранжа. Сравнить с точным решением. Построить графики. Провести анализ сходимости аппроксимации.

5. Решить ту же задачу, но с использованием квадратичных и кубических элементов.

6. Сравнить вычисления по МКЭ для случаев одинаковых и неодинаковых размеров конечных элементов. В последнем случае длины элементов выбираются в зависимости от характера поведения решения.

7. С помощью МКЭ найти решение краевой задачи, не имеющей аналитического решения. Условия задач взять из упражнения 6 раздела 8. Показать сходимость аппроксимаций.

8. Преобразовать дифференциальное уравнение 2-го порядка

$$y'' + s(x)y' + t(x)y = r(x)$$

к виду, допускающему применение процедуры Рунге, путем умножения обеих частей на некоторый множитель $\mu(x)$ (см. упражнение 7 раздела 8), после чего выполнить конечно-элементные вычисления для данной граничной задачи.

9. Вывести базисные функции для эрмитовых конечных элементов. Решить задачу из № 4, используя эрмитовы элементы.

Указание. Эрмитовы конечные элементы характеризуются тем, что обеспечивают непрерывность не только самой функции, но и ее производной (производных) на границах элементов. Чтобы получить простейшую систему таких функций, следует, используя представление

$$\varphi^e = C_1 x^3 + C_2 x^2 + C_3 x + C_4,$$

составить и решить систему уравнений

$$\varphi^e(x_1) = \alpha_1, (d\varphi^e/dx)|_{x=x_1} = \alpha_2, \quad \varphi^e(x_2) = \alpha_3, (d\varphi^e/dx)|_{x=x_2} = \alpha_4$$

относительно коэффициентов C_1, C_2, C_3, C_4 . Затем выражение для функции φ^e необходимо привести к виду $\varphi^e = \sum_{i=1}^4 \alpha_i N_i^e(x)$,

откуда и выводятся искомые функции.

10. Решить задачу одномерной теплопроводности

$$\frac{d}{dx} \left(k \frac{d\varphi}{dx} \right) = -Q; \quad 0 \leq x \leq 1, \quad \varphi(0) = 0, \quad k \frac{d\varphi}{dx}(1) = 1,$$

где

а) $k = 1, Q = 1$ при $0 \leq x \leq 1/2$ и $Q = 0$ при $1/2 \leq x \leq 1$;

б) $Q = 1, k = 1$ при $0 \leq x \leq 1/2$ и $k = 3$ при $1/2 \leq x \leq 1$.

11. С помощью МКЭ найти решение краевой задачи из упр. 4 или 5 разд. 8 «Решение краевых задач для обыкновенных дифференциальных уравнений методом Рунге», дискретизируя уравнения методом Галеркина. Показать сходимость.

Указание. Весовые функции $\{W_i\}$ взять теми же, что и аппроксимирующие функции $\{N_i\}$, т.е. с малым носителем. При этом максимальный порядок производных от $\{N_i\}$ под интегралом следует понизить методом интегрирования по частям.

12. Решить задачу из упражнения 4, 6 раздела 8, используя конечные элементы с иерархическими базисными функциями.

Указание. Рассмотрим последовательность конечно-элементных аппроксимаций

$$\varphi^e = \alpha_1^e N_1^e + \alpha_2^e N_2^e,$$

$$\varphi^e = \alpha_1^e N_1^e + \alpha_2^e N_2^e + \alpha_3^e N_3^e,$$

$$\varphi^e = \alpha_1^e N_1^e + \alpha_2^e N_2^e + \alpha_3^e N_3^e + \alpha_4^e N_4^e \text{ и т.д.,}$$

где

$$N_1^e = -(\xi - 1)/2, \quad N_2^e = (1 + \xi)/2$$

– обычные функции формы лагранжевого элемента 1-го порядка,

$$N_p^e = \begin{cases} (\xi^{p-1} - 1)/(p-1)!, & \text{если } p \text{ четно,} \\ (\xi^{p-1} - \xi)/(p-1)!, & \text{если } p \text{ нечетно,} \end{cases}$$

– при $p \geq 3$ так называемые иерархические базисные функции. Приведенные аппроксимации справедливы для стандартного конечного элемента: $-1 \leq \xi \leq 1$. Нетрудно видеть, что первые два параметра α_1^e и α_2^e здесь совпадают со значениями функции φ^e в точках $\xi = -1$ и $\xi = 1$ соответственно, а всем остальным параметрам придать подобный смысл не так просто. Да в этом и нет особой необходимости, так как слагаемые $\alpha_3^e N_3^e, \alpha_4^e N_4^e, \dots$

играют роль поправок к линейной аппроксимации. Тем не менее, можно записать

$$\alpha_p^e = d^{p-1} \varphi^e / dx^{p-1} \Big|_{\xi=0}, \quad p \geq 3,$$

т.е. равны производной $(p-1)$ -го порядка от φ^e в середине конечного элемента. Проверьте, что приведенные функции формы обеспечивают непрерывность решения на границах элементов.

Каждая следующая функция N_p^e имеет степень многочлена на единицу больше, чем предыдущая. Предположим, получено решение задачи $\varphi^{(1)}$ для обычных линейных конечных элементов. Добавляя к аппроксимации на каждом конечном элементе слагаемое $\alpha_3^e N_3^e$, получим новое решение $\varphi^{(2)}$, более точное чем $\varphi^{(1)}$, поскольку оно основано уже на квадратичной аппроксимации. Точно так же, используя еще одно дополнительное слагаемое $\alpha_4^e N_4^e$, найдем решение $\varphi^{(3)}$ – приближение кубическим полиномом. И так далее. Подумайте, в чем преимущество такого подхода?

И в заключение отметим, что переход от произвольного элемента $x \in [x_1, x_2]$ к стандартному $\xi \in [-1, 1]$ можно осуществить по формуле

$$x = \frac{x_2 - x_1}{2} \xi + \frac{x_1 + x_2}{2}.$$

При этом требуется всего лишь совершить замену переменной в интегралах, вычисляемых по области конечного элемента.

13. Указанная в предыдущем упражнении система иерархических функций отнюдь не единственно возможная. Приведем еще одну систему, определяемую интегрированием от полиномов Лежандра:

$$N_3^e = \xi^2 - 1, \quad N_4^e = 2(\xi^3 - \xi),$$

$$N_5^e = 1/4(15\xi^4 - 18\xi^2 + 3), \quad N_6^e = 7\xi^5 - 10\xi^3 + 3\xi, \dots$$

Производные этих функций обладают ортогональностью – свойством, весьма полезным при вычислениях. Решите одну из задач упражнения 4 раздела 8, используя последовательно указанные иерархические функции. Выведите явно матрицу S системы уравнений. Сделайте вывод.

14. Для нелинейного дифференциального уравнения

$$\frac{d}{dx} \left(\mu(\psi) \frac{d\psi}{dx} \right) = f(x, \psi), \quad 0 \leq x \leq 1$$

с краевыми условиями $\psi(0) = 0$, $\psi(1) = 1$ провести дискретизацию и получить численное решение на основе МКЭ

- 1) $\mu = \psi$; $f = -2$; 2) $\mu = \cos(\pi\psi/2)$; $f = 0$; 3) $\mu = 1/(\psi+1)$; $f = 0$;
 4) $\mu = e^\psi$; $f = x$; 5) $\mu = 1+0.1\psi$; $f = -10x$; 6) $\mu = 1$; $f = e^\psi$.

15. Края $x=0$ и $x=1$ плиты с коэффициентом теплопроводности $k=1$ поддерживаются при температуре 10 и 80°C. Найти стационарное распределение температуры φ , если тепло генерируется внутри плиты со скоростью $a\varphi^2$ ($a=0,01 \div 0,03$) на единицу длины.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Турчак, Л.И. Основы численных методов [Текст] / Л.И. Турчак. – М.: ФИЗМАТЛИТ, 2003. – 304 с.
2. Вержбицкий, В.М. Численные методы (линейная алгебра и нелинейные уравнения) [Текст]: учеб. пособие для вузов / В.М. Вержбицкий. – М.: Высшая школа, 2000. – 266 с.
3. Вержбицкий, В.М. Численные методы (математический анализ и обыкновенные дифференциальные уравнения) [Текст] / В.М. Вержбицкий. – М.: Высшая школа, 2001. – 382 с.
4. Формалев, В.Ф. Численные методы [Текст] / В.Ф. Формалев, Д.Л. Ревизников. – М.: ФИЗМАТЛИТ, 2004. – 400 с.
5. Сборник задач по методам вычислений: учебное пособие для вузов [Текст] / Под ред. П.И. Монастырного. – М.: ФИЗМАТЛИТ, 1994. – 320 с.
6. Форсайт, Дж. Численное решение систем линейных алгебраических уравнений [Текст]: пер. с англ. В.П. Ильина и Ю.И. Кузнецова / Дж. Форсайт, К. Молер. – М.: Мир, 1969. – 163 с.
7. Лаевский, Ю.М. Метод конечных элементов (основы теории, задачи) [Текст]: учеб. пособие / Ю.М. Лаевский. – Новосибирск: Новосиб. гос. ун-т, 1999. – 166 с.
8. Зенкевич, О. Метод конечных элементов и аппроксимация [Текст]: пер. с англ. / О. Зенкевич, К. Морган. – М.: Мир, 1986. – 318 с.
9. Сабоннадьер, Ж.-К. Метод конечных элементов и САПР [Текст]: пер. с франц. / Ж.-К. Сабоннадьер, Ж.-Л. Кулон. – М.: Мир, 1989. – 190 с.
10. Zienkiewicz, O.C. The finite element method. V.1: The basis [Text] / O.C. Zienkiewicz, R.L. Taylor. – Oxford: Butterworth Heinemann, 2000. – 689 p.
11. Hutton, D.V. Fundamentals of finite element analysis [Text] / D.V. Hutton. – The McGraw-Hill Companies, 2004. – 494 p.
12. Флетчер, К. Численные методы на основе метода Галеркина [Текст]: пер. с англ. / К. Флетчер. – М.: Мир, 1988. – 352 с.

13. Эльсгольц, Л.Э. Дифференциальные уравнения и вариационное исчисление [Текст]: учебник. Изд. 5-е. / Л.Э. Эльсгольц. – М.: Едиториал УРСС, 2002. – 320 с.

14. Кострюков, С.А. Основы вариационного исчисления [Текст]: учеб. пособие / С.А. Кострюков, В.В. Пешков, Г.Е. Шунин. – Воронеж: ФГБОУ ВПО «Воронежский государственный технический университет», 2011. – 158 с.

15. Боглаев, Ю.П. Вычислительная математика и программирование [Текст]: учеб. пособие для студентов втузов / Ю.П. Боглаев. – М.: Высшая школа, 1990. – 544 с.

16. Калиткин, Н.Н. Численные методы [Текст] / Н.Н. Калиткин. – М.: Наука, 1978. – 512 с.

17. Бахвалов, Н.С. Численные методы (анализ, алгебра, обыкновенные дифференциальные уравнения) [Текст] / Н.С. Бахвалов. – М.: Наука, 1975. – 632 с.

18. Соболев, И.М. Численные методы Монте-Карло [Текст] / И.М. Соболев. – М.: Наука, 1973. – 312 с.

19. Поршнев, С.В. Вычислительная математика. Курс лекций [Текст] / С.В. Поршнев. – СПб.: БХВ-Петербург, 2004. – 320 с.

20. Дьяконов, В.П. Maple 7: учебный курс [Текст] / В.П. Дьяконов. – СПб: Питер, 2002. – 672 с.

21. Кетков, Ю.Л. MATLAB 7: программирование, численные методы [Текст] / Ю.Л. Кетков, А.Ю. Кетков, М.М. Шульц. – СПб.: БХВ-Петербург, 2005. – 752 с.

22. Алексеев, Е.Р. Решение задач вычислительной математики в пакетах Mathcad 12, MATLAB 7, Maple 9 (Самоучитель) [Текст] / Е.Р. Алексеев, О.В. Чеснокова. – М.: ИТ Пресс, 2006. – 496 с.

23. Мэтьюз, Дж. Г. Численные методы. Использование MATLAB [Текст] / Дж. Г. Мэтьюз, К.Д. Финк. – М.: Издательский дом "Вильямс", 2001. – 720 с.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	3
1. РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И СИСТЕМ УРАВНЕНИЙ	5
1.1. Понятие о линейных и нелинейных уравнениях	5
1.2. О методах решения нелинейных уравнений	7
1.3. Решение нелинейных уравнений	9
1.3.1. Метод деления отрезка пополам	9
1.3.2. Метод простой итерации	12
1.3.3. Метод Ньютона	14
1.3.4. Метод секущих	17
1.4. Решение систем нелинейных уравнений. Метод Ньютона	19
1.5. Использование стандартных функций системы Maple	22
Упражнения	23
2. РЕШЕНИЕ ЗАДАЧ ЛИНЕЙНОЙ АЛГЕБРЫ	27
2.1. Матричные и векторные операции	27
2.2. Решение систем линейных алгебраических уравнений	33
2.2.1. Прямые методы решения СЛАУ. Факторизация матриц	35
2.2.1.1. Правило Крамера	35
2.2.1.2. Метод Гаусса	37
2.2.1.3. Метод Холесского	46
2.2.2. Итерационные методы решения СЛАУ	47
2.2.2.1. Методы Якоби и Гаусса–Зейделя	48
2.2.2.2. Метод сопряженных градиентов	52
Упражнения	54
3. ЧИСЛЕННОЕ ДИФФЕРЕНЦИРОВАНИЕ	60
Упражнения	72
4. ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ	74
4.1. Постановка задачи	74
4.2. Метод прямоугольников	77
4.3. Метод трапеций	80
4.4. Параболическая интерполяция. Формула Симпсона	82
4.5. Использование сплайнов	87
4.6. Квадратурные формулы Гаусса	88
4.7. Двойной пересчет	91

4.8. Метод статистических испытаний Монте–Карло	92
4.9. Приближенное вычисление интегралов в системе Maple	94
Упражнения	111
5. РЕШЕНИЕ ОБЫКНОВЕННЫХ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ	115
5.1. Основные понятия	115
5.2. Численное решение задачи Коши	117
5.3. Решение краевой задачи методом стрельбы	122
Упражнения	129
6. ПРИБЛИЖЕНИЕ (АППРОКСИМАЦИЯ) ФУНКЦИЙ	133
6.1. Введение	133
6.2. Интерполирование	134
6.3. Локальная интерполяция	139
6.4. Интерполирование сплайнами	143
6.5. Интерполяция Эрмита	146
6.6. Среднеквадратическое приближение	150
6.7. Аппроксимация с помощью взвешенных невязок	157
Упражнения	170
7. МЕТОД КОНЕЧНЫХ РАЗНОСТЕЙ	182
Упражнения	191
8. МЕТОД РИТЦА	195
8.1. О прямых методах вариационного исчисления	195
8.2. Простейшая задача вариационного исчисления. Краевая задача 1-го рода.	196
8.3. Вариационная задача Больца. Краевая задача 2-го и 3-го рода.	199
8.4. Метод Ритца в двумерном случае	207
Упражнения	213
9. МЕТОД ГАЛЁРКИНА	222
Упражнения	229
10. МЕТОД КОНЕЧНЫХ ЭЛЕМЕНТОВ	231
Упражнения	247
БИБЛИОГРАФИЧЕСКИЙ СПИСОК	253

Учебное издание

Кострюков Сергей Александрович
Пешков Вадим Вячеславович
Шунин Геннадий Евгеньевич
Шунина Валентина Алексеевна

ПРАКТИКУМ ПО ЧИСЛЕННЫМ МЕТОДАМ

В авторской редакции

Компьютерный набор В.В. Пешкова

Подписано к изданию 21.11.2017.
Объем данных 4,2 Мб.

ФГБОУ ВО «Воронежский государственный технический
университет»
394026 Воронеж, Московский просп., 14