МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Московский государственный
гуманитарно-экономический университет (МГГЭУ)

М.С. Максютов

Численные методы решения краевых задач

Учебное пособие

Москва
2014

ББК 22.193
М 19

Рецензент: Григорьев С.М. зав. кафедрой Прикладная математика и информатика, доцент.

М.С. Максютов
М 19 Численные методы решения краевых задач: учебное издание. – М.: МГГЭУ, 2014. – 92 с.

В пособии изложены основные определения и теоретические сведения, посвященные применению численных методов при решении краевых задач, описываемых обыкновенными дифференциальными уравнениями (ОДУ), и задач математической физики, описываемых дифференциальными уравнениями в частных производных (ДУЧП). Теоретические сведения дополняются большим количеством примеров для выполнения на персональном компьютере, приводятся тексты программ на языке FORTRAN. Особое внимание уделено применению матричных вычислений и программного обеспечения в задачах математического моделирования на основе библиотек IMSL и Numerical Recipes. Пособие может быть полезно и студентам других специальностей МГГЭУ.

Оно может быть использовано в качестве учебного материала при изучении курсов «Численные методы» и «Математическое моделирование» студентами специальности 010400.62 «Прикладная математика и информатика» МГГЭУ.

Печатается в авторской редакции.

1. Формулы численного дифференцирования

Предположим, что функция $f(x)$ определена в некоторой окрестности точки x_0 и дифференцируема в окрестности точки x_0 достаточное число раз. Исходя из определения производной, ее значение в точке x при $f(x)$ достаточно малом фиксированном значении h приближенно определяется так:

$$f'(x) \approx \tilde{f}'_+(x) = \frac{f(x+h) - f(x)}{h},$$
$$f'(x) \approx \tilde{f}'_-(x) = \frac{f(x) - f(x-h)}{h}, \tag{1.1}$$

где $\tilde{f}'_+, \tilde{f}'_-$ — соответственно разность вперед и разность назад. Приближенное значение производной, полученное по формулам (1.1), можно уточнить, вычислив среднее значение

$$\tilde{f}'(x) = \left(\tilde{f}'_+(x) + \tilde{f}'_-(x)\right)/2 = \frac{f(x+h) - f(x-h)}{2h} \tag{1.2}$$

где $\tilde{f}'(x)$ — *центральная разность*.

Оценку *погрешности аппроксимации* приближенного значения *односторонней* (1.3) и центрально-разностной производной (1.4):

$$r_+(x,h) = f'(x) - \tilde{f}'_+(x),\ r_-(x,h) = f'(x) - \tilde{f}'_-(x) \tag{1.3}$$
$$r(x,h) = f'(x) - \tilde{f}'(x) \tag{1.4}$$

можно получить, выразив величину $f'(x)$ путем разложения функции $f(x)$ в ряды Тейлора:

$$f(x \pm h) = f(x) \pm f'(x)h + \frac{f''(\xi_\pm)}{2}h^2, \tag{1.5}$$

$$f(x \pm h) = f(x) \pm f'(x)h + \frac{f''(x)}{2}h^2 \pm \frac{f'''(\xi_\pm)}{6}h^3, \tag{1.6}$$

где ξ_+ и ξ_- некоторые точки на интервалах $(x, x+h)$ и $(x-h, x)$ соответственно.

Для оценки односторонней производной в разложении (1.5) достаточно ограничится членами до первой производной включительно, а в разложении (1.6), для оценки центрально-разностной производной, необходимо учитывать члены до второй производной включительно, чтобы избежать появления нуля в знаменателе выражения (1.2).

Подставив (1.5) в (1.3) получим $\left|r_{\pm}(x,h)\right| \le C_1 h$, где $C_1 = \max\left|f''(\xi_{\pm})\right|$ на соответствующем интервале для односторонней производной.

Подставив (1.6) в (1.4) получим $\left|r(x,h)\right| \le C_2 h^2$, где $C_2 = \max\left|f'''(\xi)\right|$, и точка ξ берется на интервале $(x-h, x+h)$ для центрально-разностной производной. Таким образом, формулы (1.1) аппроксимируют производную $f'(x)$ с первым порядком точности по h, а формула (1.2) со вторым порядком точности.

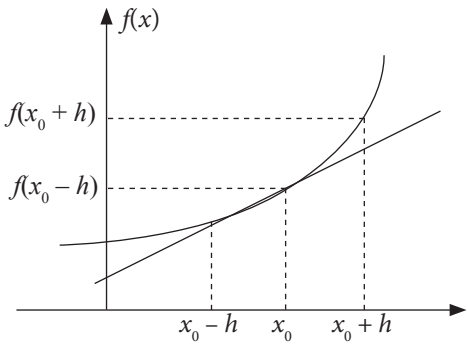


Рис. 1

Для оценки производной $f'(x)$ можно получить формулы любого порядка точности в зависимости от того, сколько членов взять в разложении (1.5)–(1.6). Однако в этом случае формулы становятся более громоздкими и в большинстве случаев не приводят к существенному выигрышу в точности численного решения. Поэтому мы ограничимся здесь формулами второго порядка точности.

Если $x = x_0$ — фиксировано (см. *рис. 1*), то оценку погрешности численного значения производной на шаге h можно получить по формулам:

$$\begin{aligned} e_{\pm}(x_0, h) &= f'(x_0) - \tilde{f}'_{\pm}(x_0) \\ e(x_0, h) &= f'(x_0) - \tilde{f}'(x_0) \end{aligned} \qquad (1.7)$$

где $f'(x_0)$ — точное значение производной в точке x_0.

Для наглядности проведем численный эксперимент по вычислению значения производной для функции $y = e^x$ в точке $x = x_0 = 1$, в зависимости от различных значений шага h. Примем, что вычисления проводятся на четырехразрядной сетке, а «точное» значение $f'(x_0) = 2.7183$. Погрешность численного решения оценим по формулам (1.7).

Листинг компьютерной программы NDF на языке FORTRAN приведен ниже. На входе программы в строке 3 задается единственный параметр — величина x_0. На выходе программы, в строках 12, 13, 14 вычисляются разность вперед (diff1), центральная разность (diff2) и разность назад (diff3), а также соответствующие погрешности (e1, e2, e3).

Листинг 1. Программа NDF

```
1  program NDF
2  integer(4):: i,n=8,deg=0
3  real(4)   :: x0=1.0,f,x,diff1,diff2,diff3,h,e1,e2,e3
4  character(70)   fmt
5    fmt=
6# '(1x,e9.2,2x,f7.4,2x,f7.4,2x,f7.4,2x,f7.4,2x,f7.4,2x,f7.4,2x,
 # f7.4)';  f(x) = exp(x)
7  e=f(1.0)
8   write(*,*)"NUMERICAL DIFFERENTIATION OF EXP FUNCTION"
9 write(*,*)"Stepsize Back Centr Forw Eps1 Eps2 Eps3"
10 do i=1,n
11    h=10.0**deg
12    diff1 = (f(x0+h)-f(x0))/h; e1=abs(e-diff1)
13    diff2 = (f(x0+h)-f(x0-h))/h/2.0;e2=abs(e-diff2)
14    diff3 = (f(x0)-f(x0-h))/h;e3=abs(e-diff3)
15    write(*,fmt)h,diff3,diff2,diff1,e3,e2,e1
16    deg=-i
17 enddo
18 end program NDF
```

Результаты расчета по компьютерной программе NDF сведем в таблицу (*см. табл. 1.1*).

Таблица 1.1

h	$\tilde{f}'_{-}(x_0)$	$\tilde{f}'(x_0)$	$\tilde{f}'_{+}(x_0)$	$e_{-}(x_0,h)$	$e(x_0,h)$	$e_{+}(x_0,h)$
1.0	1.7183	3.1945	4.6708	1.0000	0.4762	1.9525
$1\cdot10^{-1}$	2.5868	2.7228	2.8588	0.1315	0.0045	0.1406
$1\cdot10^{-2}$	2.7047	2.7183	2.7319	0.0136	0.0000	0.0136
$1\cdot10^{-3}$	2.7168	2.7183	2.7198	0.0015	0.0000	0.0015
$1\cdot10^{-4}$	2.7178	2.7191	2.7195	0.0005	0.0008	0.0012

h	$\tilde{f}'_{-}(x_0)$	$\tilde{f}'(x_0)$	$\tilde{f}'_{+}(x_0)$	$e_{-}(x_0,h)$	$e(x_0,h)$	$e_{+}(x_0,h)$
$1\cdot10^{-5}$	2.7137	2.7278	2.7336	0.0046	0.0095	0.0153
$1\cdot10^{-6}$	2.6718	2.6472	2.5401	0.0465	0.0711	0.1782
$1\cdot10^{-7}$	2.4150	3.5917	3.9429	0.3033	0.8734	1.2246

Из таблицы видно, что в рамках принятой разрядной сетки наибольшую точность вычислений дает центрально-разностная аппроксимация производной. Однако, такая ситуация имеет место лишь при оптимально выбранном шаге: когда шаг слишком большой или слишком маленький все формулы численного дифференцирования дают неудовлетворительные результаты. Оптимальным в данном случае является выбор шага в диапазоне значений $h = 0.01 - 0.001$.

В заключение приведем формулы численного дифференцирования для центрально-разностной аппроксимации производных высших порядков:

$$\tilde{f}''(x) = \frac{f(x-h) - 2f(x) + f(x+h)}{h^2} \tag{1.8}$$

$$\tilde{f}'''(x) = \frac{f(x+2h) - 2f(x+h) + 2f(x-h) - f(x-2h)}{2h^3} \tag{1.9}$$

$$\tilde{f}^{IV}(x) = \frac{f(x+2h) - 4f(x+h) + 6f(x) - 4f(x-h) + f(x-2h)}{h^4} \tag{1.10}$$

Формулы (1.8)–(1.10) аппроксимируют производную $f^{(n)}(x)$ со вторым порядком точности.

2. Метод конечных разностей для обыкновенных дифференциальных уравнений

Рассмотрим дифференциальное уравнение

$$F\left(x,u,u',\ldots,u^{(m)}\right) = 0, \quad x_0 \le x \le x_n, \tag{2.1}$$

Краевая задача для уравнения (2.1) формулируется так: найти функцию u(x), которая удовлетворяет (2.1) на интервале $[x_0,\ x_n]$, а также краевым условиям на краях интервала:

$$f_i\left(u(x_0), u'(x_0), \ldots, u^{(m-1)}(x_0)\right) = 0,\ i = 1, 2, \ldots, M$$
$$f_j\left(u(x_n), u'(x_n), \ldots, u^{(m-1)}(x_n)\right) = 0,\ \ j = M+1,\ M+2, \ldots, m. \tag{2.2}$$

Краевая задача называется линейной, если уравнения (2.1)–(2.2) линейны относительно $u(x), u'(x), \ldots, u^{(m)}(x)$.

Применение численных методов к решению линейной краевой задачи (2.1)–(2.2) рассмотрим на примере дифференциального уравнения второго порядка:

$$u'' + u = 1,\ 0 \le x \le 1; \tag{2.3}$$
$$u(0) = u(1) = 0. \tag{2.4}$$

Краевая задача (2.3)–(2.4) имеет точное решение

$$u(x) = 1 + \sin x\left((b-1)/a\right) - \cos x, \text{ где } a = \sin 1,\ b = \cos 1.$$

Для численного решения задачи (2.3)–(2.4) разобьем область непрерывного изменения аргумента x дискретным набором точек $x_0 \le x_1 < \ldots < x_{n-1} \le x_n$. Полученный набор точек называется сеткой, а точки x_i — узлами сетки, в общем случае не равноотстоящими друг от друга, т.е. $x_i = x_{i-1} + h_i\ (i = 1, \ldots, n)$. Если значение шага h постоянно, т.е. $x_i = x_{i-1} + h\ (i = 1, \ldots, n)$, то сетка называется равномерной. Таким образом, заменим область изменения непрерывного аргумента x сеточной областью, а вместо функции $u(x)$ — решения задачи (2.3)–(2.4) рассмотрим сеточную функцию — $\widetilde{u}_i(x_i)$, которая вычисляется не в произвольных точках x, а только в узлах сетки. Значения сеточной функции $\widetilde{u}_i(x_i)$ будем рассматривать как приближения к значениям $u(x_i)$ во внутренних узлах сетки (с индексами $i = 1, \ldots, n = -1$). Для внешних узлов сетки (с индексами $i = 0, n$) потребуем выполнения равенства $\overline{u}_i(x_i) = u(x_i)$.

Заменим производную u'' в уравнении (2.3) разностным соотношением (1.8). Тогда дифференциальное уравнение (2.3) в узлах равномерной сетки $x_i\ (i = 1, \ldots, n-1)$ заменяется его *конечно-разностным аналогом*

$$\frac{1}{h^2}\left(u_{i-1} - 2u_i + u_{i+1}\right) + u_i = 1, \tag{2.5}$$

с краевыми условиями в узлах с индексами $i = 0, n$

$$u_0 = 0, \quad u_n = 0, \tag{2.6}$$

где $u_i \equiv \tilde{u}_i\left(x_i\right)$.

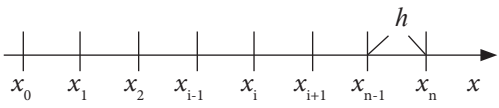


Рис. 2.1

Говорят, что уравнение (2.3) аппроксимируется его конечно-разностным аналогом (2.5) со вторым подядком точности. В формуле (2.5) шаг h определяется как величина $h = x_i - x_{i-1}$.

Дискретная задача (2.5)–(2.6) *называется разностной схемой* для краевой задачи (2.3)–(2.4). Формула (2.5) содержит $n-1$ уравнение с $n-1$ неизвестной, остальные неизвестные определяются из равенства (2.6). Таким образом, разностная схема (2.5)–(2.6) содержит $n+1$ уравнение с $n+1$ неизвестной или образует *систему сеточных уравнений* для определения значений сеточной функции.

Запишем значения сеточной функции в узлах сетки по формуле (2.5):

$$\begin{aligned}
&i = 1, \quad u_0 + \left(h^2 - 2\right)u_1 + u_2 = h^2;\\
&i = 2, \quad u_1 + \left(h^2 - 2\right)u_2 + u_3 = h^2;\\
&i = 3, \quad u_2 + \left(h^2 - 2\right)u_3 + u_4 = h^2;\\
&\cdots\cdots \quad \cdots\cdots \quad \cdots\cdots \quad \cdots\\
&i = n-1, \quad u_{n-2} + \left(h^2 - 2\right)u_{n-1} + u_n = h^2.
\end{aligned} \tag{2.7}$$

Из уравнений (2.7), учитывая условие (2.6), выпишем матрицу коэффициентов при неизвестных значениях сеточной функции u_i в *табл. 2.1*.

В таблице 2.1 диагональный элемент $D = \left(h^2 - 2\right)$. Как видно из таблицы, матрица коэффициентов системы (2.7) не плотно заполненная — по существу структура матрицы A является *ленточной*, с *шириной ленты* 3 и всеми остальными элементами равными 0.

Таким образом, необходимо решить систему уравнений $[A]\{U\} = \{b\}$ (2.8), где матрица A имеет вид, показанный в *табл. 2.1*, компоненты вектора $\{U\}$ есть значения сеточной функции u_i, а компоненты вектора $\{b\}$ $b_i = h^2 \ \left(i = 1, ..., n-1\right)$.

Таблица 2.1

u_i/i	u_1	u_2	u_3	u_{i-1}	u_i	u_{n+1}	u_{n-2}	u_{n-1}
1	**D**	**1**	0	0	.	0	0	0
2	**1**	**D**	**1**	0	.	0	0	0
.	.	.	.	.	.	.	.	.
I	.	.	.	**1**	**D**	**1**	.	.
.	.	.	.	.	.	.	.	.
$n-2$	0	0	0	0	.	**1**	**D**	**1**
$n-1$	0	0	0	0	.	0	**1**	D

$$\begin{bmatrix} D\,1 & & & & & & \\ 1\,D\,1 & & & & & & \\ \cdots & & & & & & \\ 1\,D\;1 & & & & & & \\ \cdots & & & & & & \\ 1\,D\,1 & & & & & & \\ 1\,D & & & & & & \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ . \\ u_i \\ . \\ u_{n-2} \\ u_{n-1} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ . \\ b_i \\ . \\ b_{n-2} \\ b_{n-1} \end{bmatrix}. \qquad (2.9)$$

Существуют специальные численные методы решения систем уравнений с ленточными матрицами, в частности алгоритм Томаса, также называемый методом прогонки. Алгоритм Томаса (1949) пригоден для решения систем уравнений вида $[A]\{V\} = \{d\}$, в случае трехдиагональной матрицы A:

$$\begin{bmatrix} b_1\,c_1 & & & & & & \\ a_2\,b_2\,c_2 & & & & & & \\ \cdots & & & & & & \\ a_i\,b_i\,c_i & & & & & & \\ \cdots & & & & & & \\ a_{N-1}\,b_{N-1}\,c_{N-1} & & & & & & \\ a_N\;\;b_N & & & & & & \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ . \\ v_i \\ . \\ v_{N-1} \\ v_N \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ . \\ d_i \\ . \\ d_{N-1} \\ d_N \end{bmatrix}. \qquad (2.10)$$

Алгоритм Томаса состоит из двух частей. Сначала уравнения (2.10) приводятся к виду

$$\begin{bmatrix} 1 & c_1' & & & & & \\ & 1 & c_2' & & & & \\ & & \cdots & & & & \\ & & & 1 & c_i' & & \\ & & & & \cdots & & \\ & & & & & 1 & c_{N-1}' \\ & & & & & & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ . \\ v_i \\ . \\ v_{N-1} \\ v_N \end{bmatrix} = \begin{bmatrix} d_1' \\ d_2' \\ . \\ d_i' \\ . \\ d_{N-1}' \\ d_N' \end{bmatrix} \tag{2.11}$$

так, чтобы исключить из них коэффициенты a_i, а коэффициенты b_i привести к единичным значениям. Для первого уравнения это можно сделать по формулам:

$$c_i' = \frac{c_1}{b_1}, \quad d_1' = \frac{d_1}{b_1}, \tag{2.12}$$

для всех других уравнений по формулам:

$$c_i' = \frac{c_i}{b_i - a_i c_{i-1}'}, \quad d_1' = \frac{d_i - a_i d_{i-1}'}{b_i - a_i c_{i-1}'}. \tag{2.13}$$

На этом заканчивается процесс *прямой прогонки*. Вторая часть алгоритма сводится к *обратной прогонке*, по формулам

$$v_N = d_N', \;\; v_i = d_i' - v_{i+1} c_i' \tag{2.14}$$

определяются неизвестные v_i.

Алгоритм Томаса устойчив к ошибкам округления, если выполнено следующее условие $|b_i| \geq |a_i| + |c_i|$, (2.15) однако известно, что алгоритм Томаса устойчив к ошибкам округления и при не соблюдении условия (2.15).

Программу, реализующую алгоритм (2.11)–(2.14), несложно составить самостоятельно либо можно воспользоваться готовым модулем из математической библиотеки. В частности, можно подключить модуль tridag.for из пакета Numerical Recipes (см. *приложение 1*).

Применив к решению системы (2.9) алгоритм Томаса, где $b_i = D$, $a_i = c_i = 1$, можем получить решение краевой задачи (2.3)–(2.4) во всех внутренних узлах $(i = 1,..., n-1)$.

Для решения краевой задачи (2.3)–(2.4) составим простую компьютерную программу FDIFF1 на языке FORTRAN. На входе программы, в строке 3 задается общее число узлов N0, а также число внутренних узлов NP. В строке 5 задаются краевые условия (2.6), в строке 6 задается начальная и конечная точка интегрирования. Компоненты трехдиагональной матрицы и вектора правой части системы уравнений (2.9) задаются в строке 12. Точное решение краевой задачи записано в строке 8. Алгоритм Томаса реализован в виде функции tridag (строка 14). В данном случае использован готовый модуль из пакета Numerical Recipes.

На выходе программы, после вызова функции tridag (строка 14) заполняется массив значений сеточной функции u(NP).

Листинг 2. Программа FDIFF1

```
1  program FDIFF1
2  integer(4)          i,     indx
3  integer(4),parameter:: N0=14,  NP=N0-2
4  real(4)             diag(NP),  superd(NP), subd(NP)
5  real(4)          :: rhs(NP),u0=0.0,un=0.0,u(NP),err
6  real(4)          :: h, D, X0=0.0, Xn=1.0, tf, x
7  character(70)       fmt
8  tf(x) = - cos(x)+((cos(1.0)-1.0)/sin(1.0))*sin(x)+1.0
9  h=(Xn-X0)/(N0-1); D=h*h-2.0; x=0.0
10 fmt='(1x,i3,2x,e9.2,2x,f12.8,2x,f12.8,2x,e9.2)'
11 ! forming the tridiagonal coefficients
12 diag=D;rhs =h*h; subd=1.0; superd=1.0
13 ! carry out solution
14 call tridag(subd,diag,superd,rhs,u,NP)
15 write(*,*)"Nods Stepsize  Numerically   Exactly   Error"
16 do i=0,N0
17   x=i*h;  indx=i+1
18   if(indx==1)then
19     err=0.0
20     write(*,fmt)indx,L0,u0,u0,err
21   elseif(indx>1.and.indx<=NP+1)then
22     err=abs(tf(x)-u(i))
23     write(*,fmt)indx,x,u(i),tf(x),err
24   elseif(indx==N0)then
25     err=0.0
26     write(*,fmt)indx,Ln,un,un,err
```

```
27   endif
28 enddo
29 end program FDIFF1
```

Необходимо отметить, что результаты, получаемые по компьютерной программе FDIFF1, определяются алгоритмом решения системы линейных уравнений (алгоритмом Томаса).

На практике, когда вычисления производят на компьютере, неизбежно возникают ошибки округления, вызванные ограниченностью разрядной сетки компьютера. Поэтому, на самом деле, вместо значений сеточной функции $\tilde{u}(x_i)$ мы получаем значения некоторой другой функции $\overline{u}(x_i)$, связанной с численным решением системы линейных уравнений. Значения $\overline{u}(x_i)$ можно рассматривать как приближения к значениям $\tilde{u}(x_i)$ во внутренних узлах.

Запишем результаты, полученные по программе DIFF1 в следующую таблицу:

Таблица 2.2

i	x_i	$\tilde{u}(x_i)$	$u(x_i)$	$e_i = \lvert u(x_i) - \overline{u}(x_i) \rvert$
1.	0.0000	0.0000	0.0000	$0.00\ x\ 10^{-5}$
2.	0.0769	-0.0390	-0.0390	$2.12\ x\ 10^{-5}$
3.	0.1538	-0.0719	-0.0719	$3.92\ x\ 10^{-5}$
4.	0.2308	-0.0985	-0.0984	$5.39\ x\ 10^{-5}$
5.	0.3077	-0.1186	-0.1185	$6.51\ x\ 10^{-5}$
6.	0.3846	-0.1320	-0.1319	$7.26\ x\ 10^{-5}$
7.	0.4615	-0.1387	-0.1387	$7.63\ x\ 10^{-5}$
8.	0.5385	-0.1387	-0.1387	$7.63\ x\ 10^{-5}$
9.	0.6154	-0.1320	-0.1319	$7.26\ x\ 10^{-5}$
10.	0.6923	-0.1186	-0.1185	$6.51\ x\ 10^{-5}$
11.	0.7692	-0.0985	-0.0984	$5.39\ x\ 10^{-5}$
12.	0.8462	-0.0719	-0.0719	$3.92\ x\ 10^{-5}$
13.	0.9231	-0.0390	-0.0390	$2.12\ x\ 10^{-5}$
14.	1.0000	0.0000	0.0000	$0.00\ x\ 10^{-5}$

Вычислим среднеквадратичную погрешность численного решения краевой задачи (2.5)–(2.6) по формуле

$$r_m = \left[\left(\sum_{i=1}^{n} \left(u(x_i) - \tilde{u}(x_i) \right)^2 \right) / n \right]^{1/2}, \qquad (2.16)$$

откуда $r_m = 5.37 \cdot 10^{-5}$. Построим график изменения среднеквадратичной погрешности (2.16) в зависимости от числа узлов *n*:

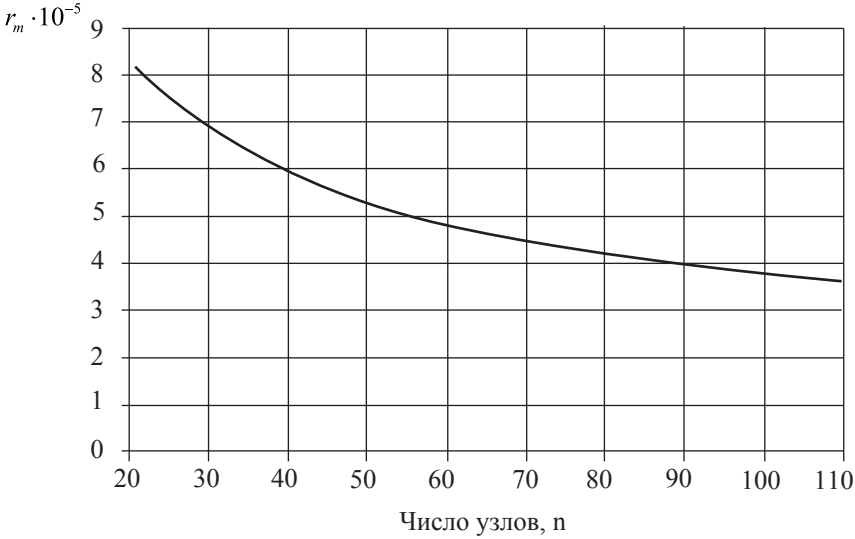


Рис. 2.2

Анализ данных таблицы 2.2 и графика на рис. 2.2 позволяет утверждать, что численное решение краевой задачи (2.3)–(2.4) методом конечных разностей (МКР) получено с хорошей точностью и сходится к точному решению со скоростью порядка $O\left(h^2\right)$.

Отличительной особенностью краевых задач для дифференциальных уравнений высших порядков является то, что независимой переменной является не только искомая функция, но и ее производные, в отличие от уравнения второго порядка, где не требовалось значений производных на краях.

Рассмотрим дифференциальное уравнение

$$\frac{d^4u}{dx^4} + u = 1, \tag{2.17}$$

при следующих граничных условиях:

$$u(0) = u(L) = 0; u'(0) = u'(L) = 0. \tag{2.18}$$

Краевая задача (2.17)–(2.18) имеет точное решение

$$\begin{aligned} u(x) = e^{\frac{\sqrt{2}}{2}x}(c_1 \cos\frac{\sqrt{2}}{2}x + c_2 \sin\frac{\sqrt{2}}{2}x) + \\ + e^{-\frac{\sqrt{2}}{2}x}(c_3 \cos\frac{\sqrt{2}}{2}x + c_4 \sin\frac{\sqrt{2}}{2}x) + 1, \end{aligned} \tag{2.19}$$

а постоянные с находятся из условий на краях (2.18) в зависимости от длины интервала L.

Запишем конечно-разностный аналог дифференциального уравнения (2.1) для случая центральной разностной аппроксимации, заменив производную d^4u / dx^4 по формуле (1.10)

$$\frac{1}{h^4}\left(u_{i-2} - 4u_{i-1} + 6u_i - 4u_{i+1} + u_{i+2}\right) + u_i = 1. \tag{2.20}$$

Наличие граничных условий в виде производных налагает на них требования центрально-разностной аппроксимации в случае применения общей схемы как центральной разности. Однако, в этом случае требуется наличие *законтурных узлов* в граничных точках (см. *рис. 2.3*).

Рис. 2.3

При этом, краевые условия запишутся следующим образом:

$$\frac{du}{dx}(0) = \frac{1}{2h}\left(u_1 - u_{-1}\right), \quad \frac{du}{dx}(L) = \frac{1}{2h}\left(u_{n+1} - u_{n-1}\right); \tag{2.21}$$

Тогда, разностная схема краевой задачи (2.17)–(2.18) с учетом граничных условий (2.20) запишется так:

$$u_{i-2} - 4u_{i-1} + \left(6 + h^4\right)u_i - 4u_{i+1} + u_{i+2} = h^4; \tag{2.22}$$

$$u_0 = 0, u_1 - u_{-1} = 0, u_n = 0, u_{n+1} - u_{n-1} = 0. \tag{2.23}$$

Так, можем записать

$$i = 1, u_{-1} - 4u_0 + \left(6 + h^4\right)u_1 - 4u_2 + u_3 =)h^4, \text{ или}$$
$$\left(7 + h^4\right)u_1 - 4u_2 + u_3 = h^4;$$
$$i = 2, u_0 - 4u_1 + \left(6 + h^4\right)u_2 - 4u_3 + u_4 = h^4, \text{ или}$$
$$-4u_1 + \left(6 + h^4\right)u_2 - 4u_3 + u_4 = h^4;$$
$$i = 3, u_1 - 4u_2 + \left(6 + ah^4\right)u_3 - 4u_4 + u_5 = b(x_i)h^4.$$

Для узла с номером $i = n$ невозможно записать уравнение, так как требуется знать величину u в узле с номером $i = n + 2$. Поэтому последнее уравнение запишем в узле

$$i = n-1, u_{n-3} - 4u_{n-2} + \left(6 + ah^4\right)u_{n-1} - 4u_n + u_{n+1} = h^4, \text{ или}$$
$$u_{n-3} - 4u_{n-2} + \left(7 + ah^4\right)u_{n-1} = h^4.$$

Таким образом, решение удается получить для внутренних узлов $i = 1,..., n-1$. Из полученных уравнений выпишем матрицу коэффициентов

Таблица 2.3

	U_1	U_2	U_3	U_4	U_5	∴	U_{n-3}	U_{n-2}	U_{n-1}
$I = 1$	**1 + D**	**-4**	**1**	0	0	∴	0	0	0
$I = 2$	**1**	D	**-4**	**1**	0	∴	0	0	0
$I = 3$	**1**	**-4**	D	**-4**	**1**	∴	0	0	0
∴	∴	∴	∴	∴	∴	∴	∴	∴	∴
$n - 1$	0	0	0	0	0	0	**1**	**-4**	**1 + D**

В таблице 2.3 диагональный элемент $D = \left(6 + h^4\right)$.

Применение конечно-разностных схем повышенного порядка приводит к матрицам с более широкими лентами, чем трехдиагональная матрица. Как нетрудно убедиться, матрица в таблице 4 — пятидиагональная. Однако можно провести на этот случай обобщение алгоритма Томаса. Допустим, что нам необходимо решить следующую пятидиагональную систему:

$$\begin{bmatrix} b_1 & c_1 & f_1 & & & & \\ a_2 & b_2 & c_2 & f_2 & & & \\ e_3 & a_3 & b_3 & c_3 & f_3 & & \\ & \ddots & & & & & \\ & e_i & a_i & b_i & c_i & f_i & \\ & & \ddots & & & & \\ & & e_{N-1} & a_{N-1} & b_{N-1} & c_{N-1} & \\ & & & e_N & a_N & b_N & \end{bmatrix} \begin{bmatrix} v_1 \\ \cdot \\ \cdot \\ \cdot \\ v_i \\ \cdot \\ \cdot \\ v_N \end{bmatrix} = \begin{bmatrix} d_1 \\ \cdot \\ \cdot \\ \cdot \\ d_i \\ \cdot \\ \cdot \\ d_N \end{bmatrix}. \tag{2.24}$$

Сначала исключаются все элементы e_i, что дает

$$\begin{bmatrix} b_1 & c_1 & f_1 & & & & \\ a_2 & b_2 & c_2 & f_2 & & & \\ & a_3' & b_3' & c_3' & f_3' & & \\ & & \ddots & & & & \\ & & a_i' & b_i' & c_i' & f_i' & \\ & & \ddots & & & & \\ & & \ddots & & & & \\ & & & a_{N-1}' & b_{N-1}' & c_{N-1}' & \\ & & & & a_N & b_N & \end{bmatrix} \begin{bmatrix} v_1 \\ \cdot \\ \cdot \\ \cdot \\ v_i \\ \cdot \\ \cdot \\ \cdot \\ v_N \end{bmatrix} = \begin{bmatrix} d_1 \\ \cdot \\ \cdot \\ \cdot \\ d_i' \\ \cdot \\ \cdot \\ \cdot \\ d_N \end{bmatrix}, \tag{2.25}$$

где

$$a_i' = a_i - \frac{e_i b_{i-1}'}{a_{i-1}'}, \tag{2.26}$$

$$
\begin{aligned}
b_i' &= b_i - \frac{e_i c_{i-1}'}{a_{i-1}'}, \\
c_i' &= c_i - \frac{e_i f_{i-1}'}{a_{i-1}'}, \\
f_i' &= f_i, \\
d_i' &= d_i - \frac{e_i d_{i-1}'}{a_{i-1}'}.
\end{aligned}
\tag{2.26}
$$

Второй этап эквивалентен обычному алгоритму Томаса. Здесь исключаются все элементы a_i', что в итоге дает

$$
\begin{bmatrix}
1 & c_1'' & f_1 & & & & & & \\
 & 1 & c_2'' & f_2'' & & & & & \\
 & & \ddots & \ddots & \ddots & & & & \\
 & & & \ddots & \ddots & & & & \\
 & & & & 1 & c_i'' & f_i'' & & \\
 & & & & & \ddots & \ddots & & \\
 & & & & & & 1 & c_{N-1}'' & \\
 & & & & & & & 1 &
\end{bmatrix}
\begin{bmatrix} v_1 \\ \cdot \\ \cdot \\ \cdot \\ v_i \\ \cdot \\ \cdot \\ \cdot \\ v_N \end{bmatrix}
=
\begin{bmatrix} d_1 \\ \cdot \\ \cdot \\ \cdot \\ d_i'' \\ \cdot \\ \cdot \\ \cdot \\ d_N'' \end{bmatrix},
\tag{2.27}
$$

где

$$
\begin{aligned}
c_i'' &= \frac{c_i' - a_i' f_{i-1}''}{b_i' - a_i' c_{i-1}''}, \\
f_i'' &= \frac{f_i'}{b_i' - a_i' c_{i-1}''}, \\
d_i'' &= \frac{d_i' - a_i' d_{i-1}''}{b_i' - a_i' c_{i-1}''}.
\end{aligned}
\tag{2.28}
$$

Далее, для решения уравнений (2.26) проводится обратная подстановка по формуле:

$$
v_i = d_i'' - c_i'' v_{i+1} - f_i'' v_{i+2.} \tag{2.29}
$$

Таким образом, для решения пятидиагональной системы необходимы две прямые и одна обратная прогонки.

Все этапы обобщенного алгоритма Томаса можно представить как последовательность прямых прогонок для приведения матрицы A к верхнетреугольному виду (LU-разложение), и обратной подстановки для определения вектора V.

Программу, реализующую алгоритм (2.24)–(2.28) также несложно составить самостоятельно, либо можно воспользоваться готовым модулем из математической библиотеки. В частности, можно подключить модуль bandec.for для LU-разложения ленточной матрицы и модуль banbks.for для обратной подстановки из пакета Numerical Recipes (см. *приложение 1*). Применив для численного решения краевой задачи (2.17)–(2.18) обобщенный алгоритм Томаса (2.24)–(2.28), можно получить значения сеточной функции во всех внутренних узлах.

Результаты численного решения краевой задачи для $L = 1$ и $n = 14$ показаны в табл. 2.4.

Таблица 2.4

i	x_i	$\tilde{u}(x_i)$	$u(x_i)$	$e_i = \lvert u(x_i) - \tilde{u}(x_i) \rvert$
1.	0.0000	0.0000	0.0000	0.00×10^{-4}
2.	0.0769	0.0002	0.0002	0.35×10^{-4}
3.	0.1538	0.0008	0.0007	0.64×10^{-4}
4.	0.2308	0.0014	0.0013	0.87×10^{-4}
5.	0.3077	0.0020	0.0019	0.10×10^{-3}
6.	0.3846	0.0024	0.0023	0.12×10^{-3}
7.	0.4615	0.0027	0.0026	0.12×10^{-3}
8.	0.5385	0.0027	0.0026	0.12×10^{-3}
9.	0.6154	0.0024	0.0023	0.12×10^{-3} 9.
10.	0.6923	0.0020	0.0019	0.10×10^{-3}
11.	0.7692	0.0014	0.0013	0.87×10^{-4}
12	0.8462	0.0008	0.0007	0.35×10^{-4}
13.	0.9231	0.0002	0.0002	0.35×10^{-4}
14.	1.0000	0.0000	0.0000	0.00×10^{-4}

Среднеквадратичная погрешность по формуле (2.16) составит $r_m = 8.64 \cdot 10^{-5}$. Рассмотрим график изменения среднеквадратичной погрешности численного решения в зависимости от числа узлов (*рис. 2.4*).

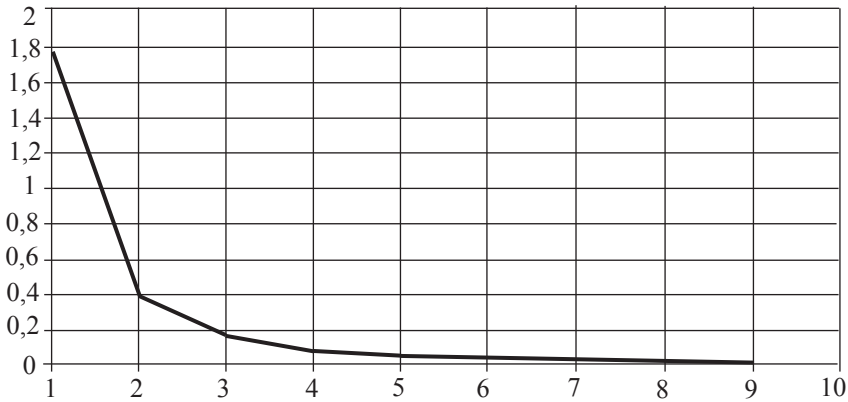


Рис. 2.4. Число узлов, $n \times 10$

Анализ данных графика на *рис. 2.4*. показывает, что с увеличением густоты сетки, погрешность уменьшается скачкообразно при увеличении числа узлов сетки с 10 до 20, далее погрешность уменьшается более плавно.

Как и в случае краевой задачи (2.3)–(2.4) численное решение краевой задачи (2.17)–(2.18) методом конечных разностей получено с хорошей точностью и также сходится к точному решению со скоростью порядка $O\left(h^2\right)$.

3. Метод ортогональной прогонки

Этот метод удобно применять в краевых задачах, связанных с численным интегрированием систем дифференциальных уравнений.

Рассмотрим на интервале $\left[x_0, x_1\right]$ следующую краевую задачу:

$$\overline{y}' = \left[A(x)\right]\overline{y} + \overline{b}(x); \tag{3.1}$$

$$y_{n/2+i}(x_0) = y^{(0)}_{n/2+i}; y_{n/2+i}(x_L) = y^{(L)}_{n/2+i}; (i = 1,\ldots,\frac{n}{2}). \tag{3.2}$$

Разобьем интервал $\left[x_0, x_L\right]$ на m подынтервалов, в результате чего получим $m + 1$ точек с координатами

$$x_s = x_0 + s(x_L - x_0)/m \quad (s = 0,1,...,m). \tag{3.3}$$

Общее решение краевой задачи (3.1)–(3.2) в точке x_0 запишем в виде

$$\overline{y}(x_0) = \overline{y}_0(x_0) + \left[Y(x_0)\right]\beta^{(0)}, \tag{3.4}$$

где $\overline{y}_0(x_0)$ — частное решение неоднородной системы (3.1), $\left[Y(x_0)\right]$ — матрица, столбцами которой являются линейно-независимые решения однородной системы, соответствующей системе (3.1):

$$\left[Y(x_0)\right] = \left[\overline{y}_1(x_0)\ldots\overline{y}_k(x_0)\right]; \tag{3.5}$$

$\overline{\beta}^{(0)}$ — вектор, компонентами которого являются произвольные постоянные:

$$\overline{\beta}^{(0)} = [\beta_1^{(0)}\ldots\beta_k^{(0)}]^T;\ k = n/2. \tag{3.6}$$

Систему векторов $\overline{y}_0(x_0)$ и $\overline{y}_j(x_0)(j = 1,\ldots k)$ выберем следующим образом:

$$\overline{y}_0(x_0) = \begin{Bmatrix} 0 \\ 0 \\ \ldots \\ 0 \\ y_{k+1}^{(0)} \\ y_{k+2}^{(0)} \\ \ldots \\ y_n^{(0)} \end{Bmatrix};\quad \left[Y(x_0)\right] = \begin{bmatrix} 10\ldots0 \\ 01\ldots0 \\ \ldots\ldots\ldots \\ 00\cdots1 \\ 00\ldots0 \\ 00\ldots0 \\ 00\ldots0 \\ 00\ldots0 \end{bmatrix}, \tag{3.7}$$

Решение (3.4) краевой задачи (3.1)–(3.2) в точке x_0 удовлетворяет граничным условиям (3.2) при любом выборе произвольных постоянных β, которые и являются недостающими компонентами вектора-решения $\overline{y}(x_0)$ в начальной точке.

Общее решение краевой задачи (3.1)–(3.2) на подинтервале $[x_0, x_l]$ представим в виде

$$\overline{y}(x) = \overline{y}_0(x) + \left[Y(x)\right]\beta^{(0)}, \tag{3.8}$$

где $\overline{y}_0(x)$ — решение системы неоднородных дифференциальных уравнений $\overline{y}_0'(x) = \left[A(x)\right]\overline{y}_0(x) + \overline{b}(x)$, с начальными условиями

$$\overline{y}_0(x) = \overline{y}_0(x_0); \tag{3.9}$$

$\left[Y(x)\right] = \left[\overline{y}_1(x) \ldots \overline{y}_k(x)\right]$ — решение системы однородных дифференциальных уравнений $\left[Y(x)\right]' = \left[A(x)\right]\left[Y(x)\right]$, с начальными условиями

$$\left[Y(x)\right] = \left[Y(x_0)\right]. \tag{3.10}$$

Для конечной точки подынтервала $\left[x_0, x_1\right]$ общее решение (3.8) записывается следующим образом

$$\overline{y}(x_1) = \overline{y}_0(x_1) + \left[Y(x_1)\right]\overline{\beta}^{(0)}. \tag{3.11}$$

Решение (3.11) удовлетворяет первому граничному условию в (3.2) и дифференциальному уравнению (3.1).

Проортонормируем полученную в точке x_1 систему векторов $y_j(x_1)$ $(j = 1, \ldots, k)$ матрицы Y в выражении (3.11) по формулам Грамма-Шмидта:

$$\begin{aligned}
&\omega_{11}^{(1)} = \sqrt{(\overline{y}_1, \overline{y}_1)}\,; \ \overline{z}_1^{(1)} = \overline{y}_1 / \omega_{11}^{(1)}\,; \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&\omega_{i1}^{(1)} = \left(\overline{y}_i, \overline{z}_1^{(1)}\right), \ldots, \omega_{i,i-1}^{(1)} = \left(\overline{y}_i, \overline{z}_{i-1}\right); \\
&\omega_{ii}^{(1)} = \sqrt{(\overline{y}_i, \overline{y}_i) - \sum_{j=1}^{i-1} \omega_{ij}^2}\,; \\
&\overline{z}_i^{(1)} = \left(y_i - \sum_{j=1}^{i-1} \omega_{ij}^{(1)} \overline{z}_j^{(1)} \right) / \omega_{ii}^{(1)} \quad (2 \le i \le k).
\end{aligned} \tag{3.12}$$

Вектор $\overline{y}_0(x_1)$ в выражении (3.11) ортогонализируем в конце подынтервала $[x_0, x_1]$ по формулам:

$$\omega_{0i}^{(1)} = \left(\overline{y}_0, \overline{z}_i^{(1)}\right); \overline{z}_0^{(1)} = \overline{y}_0 - \sum_{j=1}^{k} \omega_{0j}^{(1)} \overline{z}_j^{(1)} \quad (i = 1,\ldots,k). \tag{3.13}$$

В результате ортонормирования и ортогонализации векторов $\overline{y}_0(x_1)$, $\overline{y}_j(x_1)(j = 1,\ldots,k)$ в точке x_1 по формулам (3.12)–(3.13) получим матрицу $\left[Y^{(1)}\right] = \left[\overline{z}_1^{(1)} \ldots \overline{z}_k^{(1)}\right]$, столбцами которой являются линейно-независимые векторы z, а также треугольную матрицу вида

$$\left[\Omega^{(1)}\right] = \begin{bmatrix} \omega_{11}^{(1)} \omega_{21}^{(1)} \ldots\ldots \omega_{k1}^{(1)} \\ 0 \quad \omega_{22}^{(1)} \ldots\ldots \omega_{k2}^{(1)} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ 0 \quad 0 \ldots\ldots\ldots \omega_{kk}^{(1)} \end{bmatrix} \tag{3.14}$$

и векторы $\overline{z}_0^1, \overline{\omega}_0^{(1)} = \left[\omega_{01}^{(1)} \ldots \omega_{0k}^{(1)}\right]^T$. (3.15)

Таким образом, ортонормирование и ортогонализация векторов $\overline{y}_0(x_1)$, $\overline{y}_j(x_1)(j = 1,\ldots,k)$ эквивалентны следующему преобразованию

$$\left[Z^{(1)} z_0^{(1)}\right] = \left[Y(x_1)\overline{y}_0(x_1)\right]\left[\Omega_0^1\right]^{-1}, \quad \left[\Omega_0^{(1)}\right] = \begin{bmatrix} \Omega^{(1)} \omega_0^1 \\ 0 \quad 1 \end{bmatrix}. \tag{3.16}$$

На следующем подынтервале $[x_1, x_2]$ решение краевой задачи (3.1)–(3.2) представим в виде

$$\overline{y}(x) = \overline{y}_0(x) + \left[Y(x)\right]\beta^{(1)}, \tag{3.17}$$

где $\overline{y}_0(x)$ — решение задачи Коши для системы неоднородных дифференциальных уравнений (3.9), но с начальными условиями $\overline{y}_0(x) = \overline{z}_0^{(1)}$; $\overline{y}_j(x)$ —

решение задачи Коши для системы однородных дифференциальных уравнений (3.10), но с начальными условиями $\overline{y}_j(x) = \overline{z}_j^{(1)}$. Решение в точке x_1 может быть представлено как в виде (3.11), так и в следующей форме

$$\overline{y}(x_1) = \left[Z^{(1)} \overline{z}_0^{(1)} \right] \begin{Bmatrix} \beta^{(1)} \\ 1 \end{Bmatrix},$$

откуда, учитывая выражение (3.16), получим

$$\overline{y}(x_1) = \left[Y(x_1) \overline{y}_0(x_1) \right] \left[\Omega_0^1 \right]^{-1} \begin{Bmatrix} \beta^{(1)} \\ 1 \end{Bmatrix}. \tag{3.18}$$

Из сравнения решений (3.11) и (3.18) получаем зависимость $\overline{\beta}^{(1)} = \left[\Omega^{(1)} \right] \overline{\beta}^{(0)} + \omega_0^{(1)}$, связывающую постоянные $\beta^{(0)}$ на подынтервале $[x_0, x_1]$ с постоянными $\beta^{(0)}$ на подынтервале $[x_1, x_2]$.

Для полученных решений $\overline{y}_0(x_2)$ и $\overline{y}_j(x_2)$ в точке x_2 выполним процедуру ортонормирования векторов $\overline{y}_j(x_2)$ по формулам (3.12) и ортогонализацию вектора $\overline{y}_0(x_2)$ по формулам (3.13). В результате получим векторы $\overline{z}_0^2$ и $\overline{z}_j^2$, которые будут использоваться в качестве начальных данных для интегрирования на следующем подынтервале $[x_2, x_3]$. Выполнив последовательное интегрирование по всем подынтервалам интервала $[x_0, x_L]$, в конечной точке $x_m = x_L$ получим систему векторов $\overline{y}_0(x_m), \overline{y}_j(x_m) (j = 1, \ldots, k)$, которую тоже ортонормируем и ортогонализуем. Решение в точке xm имеет вид

$$\overline{y}(x_m) = \overline{z}_0^{(m)} + \left[Z^{(m)} \right] \overline{\beta}^{(m)}. \tag{3.19}$$

Подставив это решение во второе граничное условие (3.2), получим систему линейных алгебраических уравнений для определения постоянных $\overline{\beta}^{(m)}$:

$$\sum_{j=1}^{k} z_{k+j,\,j}\, \beta_j = y_{k+i}^{(l)} - z_{k+i,\,0} \quad (i = 1, \ldots, k) \tag{3.20}$$

Недостающие компоненты решения в точке x_m находим по формуле (3.19). На этом заканчивается первый этап ортогональной прогонки — прямая прогонка.

На следующем этапе — обратной прогонке определяются постоянные $\overline{\beta}^{(s)}\left(s=m-1,m-2,\ldots,0\right)$ на подынтервале $\left[x_s,x_{s+1}\right]$ из соотношений

$$\beta^{(s)}=\left[\Omega^{(s+1)}\right]\left\{\overline{\beta}^{(s+1)}-\overline{\omega}_0^{(s+1)}\right\}, \tag{3.21}$$

где $\left[\Omega^{(s+1)}\right]$ — матрица (3.14), $\overline{\omega}_0^{(s+1)}$ — вектор (3.15).

Решение в промежуточных точках ортогонализации $x_s\left(s=1,\ldots,m\right)$ находится по формуле $\overline{y}\left(x_s\right)=\overline{z}_0^{(s)}+\left[Z^{(s)}\right]\overline{\beta}^{(s)}$, а решение в начальной точке x_0 — по формуле (3.4).

В заключении отметим, что метод ортогональной прогонки устойчив к ошибкам округления и позволяет получить решение краевой задачи с высокой точностью.

В качестве примера применения метода ортогональной прогонки рассмотрим задачу из механики, где метод ортогональной прогонки используется наиболее широко.

Пусть требуется определить функцию прогиба $u(x)$ полого цилиндра радиуса r, толщиной h и длиной $2L$. В цилиндр, жестко закрепленный в торцах, подается равномерное внутреннее давление p. Дифференциальное уравнение прогиба имеет вид

$$\frac{d^4u}{dx^4}+4\beta^4u=\frac{p(2-\mu)}{2D}, \tag{3.22}$$

где $D=Eh^3/12(1-\mu^2)$, $\mu=0.3, E=2.06\cdot10^5, \beta=\left[\frac{3\left(1-\mu^2\right)}{r^2h^2}\right]^{\frac{1}{4}}$.

Ввиду симметрии задачи цилиндр рассматривается на отрезке $x\in[0, L]$. Краевые условия в этом случае запишутся так

$$u\left(0\right)=u'\left(0\right)=0,\quad u''\left(L\right)=u'''\left(L\right)=0. \tag{3.23}$$

Краевая задача (3.22)–(3.23) имеет точное решение

$$u(0) = u'(0) = 0, \quad u''(L) = u'''(L) = 0. \tag{3.24}$$

Решение краевой задачи (3.22)–(3.23) при L = 2м, h = 0.012м методом конечных разностей и методом ортогональной прогонки дает результаты (двойная точность), которые представлены в табл. 3.1.

Таблица 3.1

i	x_i	$\tilde{u}(x_i)$ м-д к. разностей	$\tilde{u}(x_i)$ м-д орт. прогонки	$u(x_i)$
1.	0.0000	0.0000000	0.0000000	0.0000000
2.	0.1111	0.0002703	0.0002292	0.0002292
3.	0.2222	0.0003478	0.0003528	0.0003528
4.	0.3333	0.0003492	0.0003536	0.0003536
5.	0.4444	0.0003448	0.0003446	0.0003446
6.	0.5556	0.0003438	0.0003432	0.0003432
7.	0.6667	0.0003438	0.0003437	0.0003437
8.	0.7778	0.0003438	0.0003439	0.0003439
9.	0.8889	0.0003439	0.0003439	0.0003439
10.	1.0000	0.0003439	0.0003439	0.0003439

Оценка среднеквадратичной погрешности по формуле (2.16) для каждого из методов дает $r_m = 1.35 \cdot 10^{-5}$ для метода конечных разностей и $r_m = 0$ для метода ортогональной прогонки в рамках семиразрядной арифметики для данных двойной точности.

4. Дифференциальные уравнения в частных производных

Рассмотрим дифференциальное уравнение в частных производных с двумя независимыми переменными

$$A\frac{\partial^2 u}{\partial x^2} + B\frac{\partial^2 u}{\partial x \partial y} + C\frac{\partial^2 u}{\partial y^2} + D\frac{\partial u}{\partial x} + E\frac{\partial u}{\partial y} + Fu + G = 0, \tag{4.1}$$

где коэффициенты A–G — некоторые постоянные. Существует классификация, позволяющая различить три разновидности такого уравнения:

Эллиптическое $B^2 - 4AC < 0$,

Параболическое $B^2 - 4AC = 0$,

Гиперболическое $B^2 - 4AC > 0$.

Прототипом гиперболического уравнения может служить одномерное *волновое* уравнение

$$\frac{\partial^2 u}{\partial t^2} = v^2 \frac{\partial^2 u}{\partial x^2}, \tag{4.2}$$

где v — постоянная, характеризующая скорость распространения волны.

Примером параболического уравнения может служить уравнение *диффузии* (или распространения тепла)

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(\alpha \frac{\partial u}{\partial x}\right), \tag{4.3}$$

где a — коэффициент диффузии.

Прототипом эллиптического уравнения является уравнение *Пуассона*

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x, y), \tag{4.4}$$

где функция ρ — задана. Если $\rho = 0$ то эллиптическое уравнение называется уравнением *Лапласа*.

Физическая задача, описываемая дифференциальными уравнениями в частных производных, называется *стационарной*, если ее решение внутри некоторой области определяется лишь условиями на границе этой области.

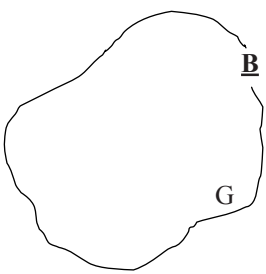


Рис. 4.1

В качестве примера рассмотрим область G решения стационароной задачи, показанной на *рис. 4.1*. В области G решение должно удовлетворять уравнениям в частных производных. На границе B области G решение должно удовлетворять граничным условиям.

Стационарная задача в физическом смысле описывает установившийся процесс, а математи-

чески стационарная задача сводится к решению задачи с граничными условиями, то есть к краевой задаче.

Таким образом, решение стационарной задачи в любой внутренней точке области G, определяется условиями, заданными на ее границе B. Стационарные задачи описываются уравнениями в частных производных эллиптического типа.

Нестационарная задача, описываемая дифференциальными уравнениями в частных производных, называется *маршевой*, если требуется найти решение дифференциального уравнения в частных производных в незамкнутой области при заданных граничных и начальных условиях. Для примера рассмотрим область решения маршевой задачи, показанной на *рис. 4.2*.

В области G решение должно удовлетворять уравнениям в частных производных. На линиях B задаются граничные условия, на линии B’ — начальные условия. Решение, удовлетворяющее граничным условиям на линиях B, должно быть получено последовательным продвижением в маршевом направлении (направлении, вдоль которого область незамкнута) в зависимости от начальных условий на линии B’. Математически такие задачи относятся к задачам типа задачи Коши, а физически описывают процессы распространения.

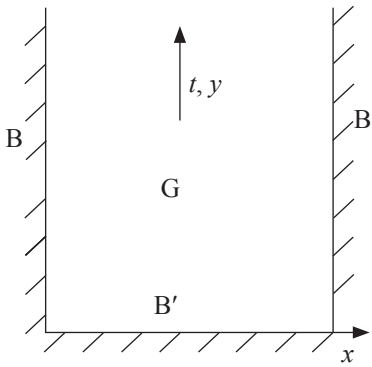


Рис. 4.2

Маршевые задачи описываются дифференциальными уравнениями гиперболического или параболического типа. Характер физических процессов при этом для задач, описываемых гиперболическими и параболическими уравнениями, различен. Поэтому методы решения этих уравнений и их математические свойства также отличаются.

Математическая задача, связанная с решением дифференциальных уравнений в частных производных (ДУЧП) вместе с начальными и граничными условиями, называется *корректно поставленной*, если она имеет единственное решение, непрерывно зависящее от начальных и граничных условий. В качестве примера, иллюстрирующего некорректно поставленную задачу, рассмотрим двумерное уравнение Лапласа

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=0 \text{ в области } -\infty < x < \infty,\ y \geq 0, \tag{4.5}$$

при следующих условиях на границе $y = 0$

$$u(x,0)=0,\ \frac{\partial u}{\partial y}(x,0)=\frac{1}{n}\sin(nx) \tag{4.6}$$

для $n > 0$. Решение задачи (4.5)–(4.6) методом разделения переменных дает следующее решение

$$u(x,y)=\frac{1}{n^2}\sin(nx)\frac{e^{ny}-e^{-ny}}{2}. \tag{4.7}$$

Из второго условия в (4.6) видно, что с ростом n величина $\frac{\partial u}{\partial y}$ уменьшается, однако для решения u это не так. При $n \to \infty$ величина $u \to \frac{e^{ny}}{n^2}$, то есть решение (4.7) неограниченно растет с ростом n, даже при малых y, что противоречит и первому условию в (4.6). Таким образом, непрерывность решения по начальным условиям отсутствует, и задача поставлена некорректно.

В самом деле, так как уравнение Лапласа эллиптическое, то граничные условия необходимо задавать по всей границе замкнутой области. В рассмотренном же примере область оставалась открытой, так как граничные условия задавались только на линии $y = 0$. Следовательно, вместо краевой задачи, что характеризует уравнения типа (4.4), в действительности решалась задача Коши, с начальными значениями (4.6), что характерно для уравнений типа (4.2)–(4.3).

Необходимо отметить, что классификация дифференциальных уравнений в частных производных в трех канонических формах (4.2)–(4.4) с вычислительной точки зрения не так существенна, как определение принадлежности данной задачи к задаче с начальными значениями — задаче Коши (маршевое решение) или к краевой задаче (стационарное решение).

5. Методы построения разностных схем

Методы построения рассмотрим на примере параболического уравнения диффузии, которое имеет вид

$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2}, \tag{5.1}$$

начальные и граничные условия

$$\begin{cases} T(0,t) = T_A, T(L,t) = T_B; \\ T(x,0) = T_0(x). \end{cases} \tag{5.2}$$

Дифференциальное уравнение в частных производных (5.1) и условия (5.2) описывают процесс нагревания теплоизолированного стержня длиной L с двух торцов — A и B. Температура на обоих торцах стержня задана и равна T_A, T_B — соответственно. Распределение температуры по длине стержня x в начальный момент времени $t = 0$ известно и задано функцией $T_0(x)$. Требуется найти температуру по длине стержня за время нагрева t, $T(x, t)$ – ?.

Расчетная область для задачи может быть представлена *рис. 5.1*.

Разобьем расчетную область равномерной конечно-разностной сеткой, при этом узлы, отсчитываемые по ординате, назовем *временными слоями* t, а узлы, отсчитываемые по абсциссе, — пространственными координатами x. Размеры *ячейки* конечно — разностной сетки при этом составят $\Delta x = L / (j\max - 1)$, $\Delta t = t / n\max$.

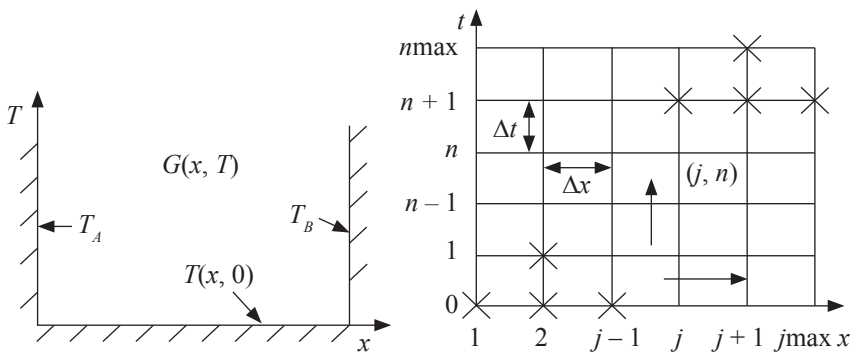


Рис. 5.1

Разложим искомую функцию $T(x, t)$ в узле (j, n) в ряд Тейлора по пространственной координате x:

$$T_{j+1}^{n} = \sum_{m=0}^{\infty} \frac{\Delta x^m}{m!} \left[\frac{\partial^m T}{\partial x^m} \right]_j^n. \tag{5.3}$$

Ряд (5.3) можно оборвать на любом числе членов. Поэтому рассмотрим разложение (5.3) до второго члена включительно. Тогда можем записать:

$$T_{j+1}^{n} = T_j^n + \Delta x \left[\frac{\partial T}{\partial x} \right]_j^n + \frac{1}{2} \Delta x^2 \left[\frac{\partial^2 T}{\partial x^2} \right]_j^n + 0(\Delta x^3),$$

откуда, отбросив в разложении члены для прозводных выше 1-го порядка, выразим величину $\left[\frac{\partial T}{\partial x} \right]_j^n$:

$$\left[\frac{\partial T}{\partial x} \right]_j^n \approx \left(\tilde{T}_{j+1}^n - \tilde{T}_j^n \right) / \Delta x,$$

ошибка аппроксимации при этом составит $O(\Delta x)$:

$$\left[\frac{\partial T}{\partial x} \right]_j^n = \frac{\left(T_{j+1} - T_j \right)}{\Delta x} + 0(\Delta x).$$

Это разности вперед. Соответственно можно получить и разности назад:

$$\left[\frac{\partial T}{\partial x} \right]_j^n \approx \left(\tilde{T}_j^n - \tilde{T}_{j-1}^n \right) / \Delta x.$$

Для более точного представления производной в узле (j, n) необходимо знать информацию о значении сеточной функции между предыдущим $(j - 1, n)$ и последующем $(j + 1, n)$ узлами, т.е. среднее между двумя узлами. Для этого, используя *трехточечный симметричный шаблон*,

представим выражение $\left[\frac{\partial T}{\partial x}\right]_j^n$ в виде:

$$\left[\frac{\partial T}{\partial x}\right]_j^n = aT_{j-1}^n + bT_j^n + cT_{j+1}^n + 0(\Delta x^m), \tag{5.4}$$

где весовые коэффициенты a, b, c подлежат определению, а число $0(\Delta xm)$ показывает точность полученного решения. Определим неизвестные коэффициенты a, b, c, для случая погрешности аппроксимации второго порядка ($m = 2$). Сначала по формуле (5.3) необходимо получить выражение для T_{j-1}^n путем замены Δx на $-\Delta x$. Тогда можем записать

$$\begin{aligned} aT_{j-1}^n &= a\sum_{m=0}^{\infty}(-1)^m \frac{\Delta x^m}{m!}\left[\frac{\partial^m T}{\partial x^m}\right]_j^n; \\ bT_j^n &= bT_j^n; \\ cT_{j+1}^n &= c\sum_{m=0}^{\infty}\frac{\Delta x^m}{m!}\left[\frac{\partial^m T}{\partial x^m}\right]_j^n. \end{aligned} \tag{5.5}$$

Сложив левые и правые части в формулах (5.5), коэффициенты a, b, c выберем таким образом, чтобы в разложениях (5.5) обращались в нуль все члены, начиная с $m = 3$. После несложных преобразований следует

$$a = -\frac{1}{2\Delta x},\ b = 0,\ c = \frac{1}{2\Delta x}. \tag{5.6}$$

Подставив (5.6) в (5.4), получим

$$\left[\frac{\partial T}{\partial x}\right]_j^n = \frac{T_{j+1}^n - T_{j-1}^n}{2\Delta x} + 0(\Delta x^2). \tag{5.7}$$

Таким образом, получим центрально-разностную аппроксимацию производной $\left[\frac{\partial T}{\partial x}\right]_j^n$ в узле (j, n) $\left[\frac{\partial T}{\partial x}\right]_j^n \approx \frac{\tilde{T}_{j+1}^n - \tilde{T}_{j-1}^n}{2\Delta x}$. (5.8)

Очевидно, что точность аппроксимации (5.8) в два раза выше, чем для односторонних производных.

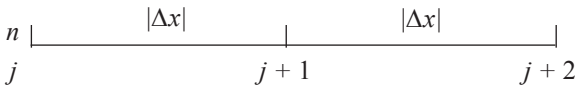
Если воспользоваться подобным представлением для производной второго порядка

$$\left[\frac{\partial^2 T}{\partial x^2}\right]_j^n = aT_{j-1}^n + bT_j^n + cT_{j+1}^n + 0(\Delta x^m),$$

то, выбрав соответствующим образом коэффициенты a, b, c, можем получить:

$$\left[\frac{\partial^2 T}{\partial x^2}\right]_j^n = \frac{T_{j-1}^n - 2T_j^n + T_{j+1}^n}{\Delta x^2} + 0(\Delta x^2) \text{ или } \left[\frac{\partial^2 T}{\partial x^2}\right]_j^n \approx \frac{\tilde{T}_{j-1}^n - 2\tilde{T}_j^n + \tilde{T}_{j+1}^n}{\Delta x^2}. \quad (5.9)$$

Используя описанный выше метод построения разностных схем при помощи весовых коэффициентов, можно получить представления с более высоким порядком точности ($m > 1$) и для односторонних производных (разностей вперед и разностей назад). Используя *трехточечный несимметричный шаблон*



получим соответствующую ему формулу для разностей вперед

$$\left[\frac{\partial T}{\partial x}\right]_j^n = \frac{-1.5T_j^n - 2T_{j+1}^n - 0.5T_{j+2}^n}{\Delta x^2} + 0(\Delta x^2). \quad (5.10)$$

Как видно, ошибка аппроксимации (5.10) такая же, как и в случае центральной разности.

Существуют также схемы с *пятиточечным несимметричным шаблоном* для представления аппроксимации односторонней производной с четвертым порядком точности с пятью весовыми коэффициентами, и схемы с *пятиточечным симметричным шаблоном* для аппроксимации центрально-разностной производной. Однако все они из-за своей громоздкости и

плохой устойчивости применяются редко, так как с вычислительной точки зрения на практике вполне достаточно аппроксимации производных со вторым порядком точности.

Разложим теперь искомую функцию $T(x, t)$ в узле (j, n) в ряд Тейлора, по временной координате t

$$T_j^{n+1} = \sum_{m=0}^{\infty} \left[\frac{\partial^m T}{\partial t^m} \right]_j^n \cdot \frac{\Delta t^m}{m!}. \tag{5.11}$$

Раскладывая ряд (5.11) до второго члена включительно, получим разности вперед по координате t:

$$\left[\frac{\partial T}{\partial t} \right]_j^n = \frac{\left(T_j^{n+1} - T_j^n \right)}{\Delta t} + 0(\Delta t) \text{ или } \left[\frac{\partial T}{\partial t} \right]_j^n \approx \frac{\left(\tilde{T}_j^{n+1} - \tilde{T}_j^n \right)}{\Delta t}. \tag{5.12}$$

Действуя по аналогии с разложением по пространственной координате, можно получить выражения для аппроксимации центрально-разностной и односторонних производных на трех и пятиточечных шаблонах.

Используя метод весовых коэффициентов, можно построить различные варианты разностных схем для численного решения уравнения теплопроводности.

На основе изложенной методики построения разностных схем рассмотрим применение метода конечных разностей к решению некоторых модельных уравнений.

6. Разностные схемы для решения уравнения диффузии

Пусть теплоизолированный стержень в задаче (5.1)–(5.2) имеет распределение температуры по всей длине в начальный момент времени $(n = 0) : T_0(x) \equiv 0$, температура нагрева на торцах $T_A = T_B = 100\ C^0$, коэффициент теплопроводности $a = 1 \cdot 10^{-5}$, длину $L = 1m$. Необходимо найти распределение температуры по длине стержня $T(x, t)$ за первые $t = 3000$ с.

Рассмотрим процедуру численного решения задачи (5.1)–(5.2) методом конечных разностей с указанными исходными данными. Метод конечных разностей, основанный на построении описанной выше дискретной сетки, состоит в том, что исходное дифференциальное уравнение в частных производных заменяется на эквивалентные ему конечно-разностные соотно-

шения, с тем, чтобы получить в дальнейшем удобный для программирования алгоритм.

Схематически этапы применения метода конечных разностей к решению маршевой задачи можно представить диаграммой (*рис. 6.1*).

Так как маршевой координатой для задачи (5.1)–(5.2) является время *t*, то одним из простейших конечно-разностных аналогов уравнения теплопроводности является схема, аппроксимирующая исходное дифференциальное уравнение в частных производных (5.1) *вперед по времени t* и *центрировано по пространству x* или *ВВЦП*.

Дискретизация исходного дифференциального уравнения в частных производных по времени при помощи разностного соотношения (5.12) и по пространству при помощи разностного соотношения (5.9) с последующей подстановкой выражений (5.9) и (5.12) в (5.1) позволяет заменить исходное дифференциальное уравнение в частных производных конечно-разностным уравнением:

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} + O\left(\Delta t\right) = \alpha \frac{T_{j-1}^n - 2T_j^n + T_{j+1}^n}{\Delta x^2} + O\left(\Delta x^2\right). \tag{6.1}$$

Перегруппировка членов в (6.1) и переход к значениям сеточной функции $\tilde{T}_j^n$ позволяет получить удобный для программирования алгоритм

$$\tilde{T}_j^{n+1} = s\tilde{T}_{j-1}^n + (1-2s)\tilde{T}_j^n + s\tilde{T}_{j+1}^n, \;\; s = \alpha \frac{\Delta t}{\Delta x^2}, \tag{6.2}$$

где *s* — *параметр дискретизации*.

Схема (6.2) — явная, двухслойная (для получения информации на следующем слое используются данные, полученные на предыдущем слое) и имеет общий порядок точности $O\left(\Delta t, \Delta x^2\right)$.

Для заданных значений *n*max и *j*max, схема (6.2) начинает работать по шаблону, обозначенному на *рис. 5.1*, с нижнего левого угла расчетной сетки (рис. 5.1) до верхнего правого угла сетки, в направлениях, указанных стрелками. При этом значения $\tilde{T}_1^n$, $\tilde{T}_{j\max}^n \left(n = 1,..,n\max\right)$ выбираются из граничных условий (5.2) в виде величин T_A и T_B соответственно, а значения $T_j^0 \left(j = 1,..,j\max\right)$ берутся из начального условия (5.2) в виде значений функции $T_0(x_j)$ в узлах *j*. Однако при формировании граничных условий в

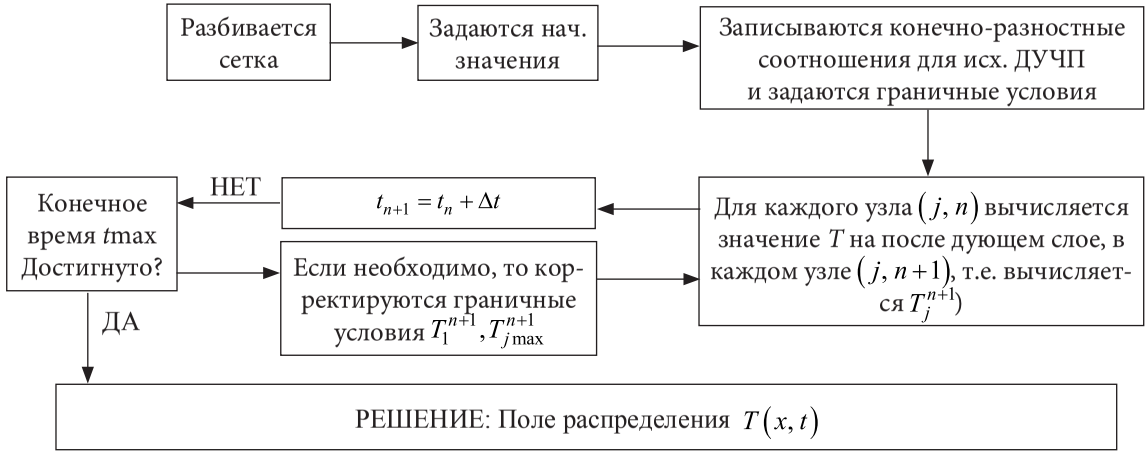

Разбивается сетка
Задаются нач. значения
Записываются конечно-разностные соотношения для исх. ДУЧП и задаются граничные условия
Для каждого узла (j, n) вычисляется значение T на после дующем слое, в каждом узле $(j, n+1)$, т.е. вычисляется T_j^{n+1})
$t_{n+1} = t_n + \Delta t$
НЕТ
Конечное время tmax Достигнуто?
Если необходимо, то корректируются граничные условия $T_1^{n+1}, T_{j\max}^{n+1}$
ДА
РЕШЕНИЕ: Поле распределения $T(x, t)$

Рис. 6.1

первом слое ($n = 1$) следует учитывать некоторые физические ограничения, что отражено и в диаграмме, в виде корректировки граничных условий.

Материал стержня в задаче (5.1)–(5.2) имеет физические характеристики, в частности коэффициент теплопроводности α, поэтому при достаточно малом шаге по времени Δt торцы стержня (узлы $j = 1$ и $j = j\max$) не могут также быстро нагреться до температуры на границах T_A и T_B в первом временном слое ($n = 1$, $t = \Delta t$). В этом случае, в качестве граничных значений температуры в первом слое принимают усредненное значение между начальным распределением температуры в стержне и температурой на границах:

$$\tilde{T}_1^1 = \frac{T_A - T_0(0)}{2}, \tilde{T}_{j\max}^1 = \frac{T_B - T_0(L)}{2}. \qquad (6.3)$$

Рассмотрим простую компьютерную программу GDIFF1 на языке Fortran, реализующую схему ВВЦП. Для краткости записи приводится только процедура, реализующая алгоритм (6.2).

В начале принимается $j\max = 11$ узлов, а величина Δt вычисляется путем задания параметру s значения $s = 0.5$. Выход из внешнего цикла процедуры (строки 13–32), чтобы избежать «зацикливания», контролируется не только временем нагрева tmax (строка 13), но максимальным числом временных слоев nmax, которое принимается, равным $n\max = 500$.

Для уменьшения влияния погрешности значения T масштабируются, т.е. вводится безразмерная температура в массиве tn, равная $tn(j) = \dfrac{\tilde{T}_j^n}{\max(T_A, T_B)}$, которая затем переводится в размерную, в массиве td.

Задача (5.1)–(5.1) допускает точное решение, полученное методом разделения переменных:

$$T(x,t) = T_A - \sum_{m=1}^{\infty}\left[\frac{400}{(2m-1)\pi}\right]\sin\left[(2m-1)\pi x\right]e^{-\alpha(2m-1)^2\pi^2 t}, \qquad (6.4)$$

где m-число членов разложения, в котором достаточно принять $10 \le m \le 20$. Точное решение накапливается в массиве te. Массивы задаются в строке 10.

Листинг 3. Программа GDIFF1[1]

```
 1 Subroutine FTCSproc(id)
 2 use task
 3 integer(4) id, tmp
 4 tmp=id; cvs='f7.2'
 5 write(cvn,'(i3)')jmax
 6 fmt='('//'a3'//','//'f7.0'//','//'a4'//','//cvn//cvs//')'
 7 fmt2= '('//'1x'//','//'a3'//','//'10x'//','//cvn//cvs//')'
 8 jmap=jmax-1; ajm=jmap; Pi=atan(1.)*4.
 9 delx=L/ajm; delt=delx*delx*s/alpha
10 allocate(tn(jmax),dum(jmax),td(jmax),x(jmax),te(jmax))
11 write(1,'(/,a80)')"THE NUMERICAL SOLUTIONN OF ↳
DIFFUSION EQUATION"
12 tn=0.0; n=0; t=0
13 do while(t<tmax)
14   tn(1)=1.0; tn(jmax)=1.0
15   if(t<tol)then
16     tn(1)=0.5
17     tn(jmax)=0.5
18   endif
19   td(1) = 100.*tn(1);td(jmax)=100.*tn(jmax)
20   do j=2, jmap
21         dum(j)=(1.-2.*s)*tn(j)+s*(tn(j-1)+tn(j+1))
22   enddo
23   do j=2,jmap
24         tn(j)=dum(j)
25   enddo
26   do j=2, jmap
27         td(j)=100.*tn(j)
28   enddo
29   t=t+delt; n=n+1
30   if(n>=nmax)exit
31   write(1,fmt)"t=",t,"TD=",(td(i),i=1,jmax)
```

[1] В этом и других листингах в конце некоторых строк кода встречается символ ↳. он означает разрыв строки в книге. В коде при этом разрыва строки быть не должно.

```
32 enddo
33 end
```

Начальные условия задаются в строке 12, а граничные условия в строках 14–18. Вычисления по алгоритму (4.18) происходят во внутреннем цикле в строках 20–22.

Для контроля числа погрешности численного решения можно использовать формулу среднеквадратичной погрешности

$$r_m = \left[\left(\sum_{j=1}^{JMAX} (T_j - \tilde{T}_j)^2 \right) / JMAX \right]^{1/2}, \qquad (6.5)$$

а для m в числе членов разложения для точного решения (6.4) достаточно принять значение $m = 20$.

Типичные результаты расчета на основе заданных выше исходных параметров по программе GDIFF1 приведены в листинге 6.1.

Для всех данных в листинге 6.1, точное значение вычислялось по формуле (4.20) при $m = 20$, а погрешность решения определялась по формуле (6.5) для последнего слоя.

Листинг 6.1

```
      ЧИСЛЕННОЕ ЗНАЧЕНИЕ ТЕМПЕРАТУРЫ T(x, t) В СТЕРЖНЕ
                    ПО СХЕМЕ ВВЦП
 t=500 : 50.00 25.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 25.00 50.00
 t=1000:100.00 50.00 12.50 0.00 0.00 0.00 0.00 0.00 12.50 50.00 100.00
 t=1500:100.00 56.25 25.00 6.25 0.00 0.00 0.00 6.25 25.00 56.25 100.00
 t=2000:100.00 62.50 31.25 12.50 3.13 0.00 3.13 12.50 31.25 62.50 100.00
 t=2500:100.00 65.63 37.50 17.19 6.25 3.13 6.25 17.19 37.50 65.63 100.00
 t=3000:100.00 68.75 41.41 21.88 10.16 6.25 10.16 21.88 41.41 68.75 100.00
      ТОЧНОЕ ЗНАЧЕНИЕ ТЕМПЕРАТУРЫ В ПОСЛЕДНЕМ СЛОЕ:
 t=3000:100.00 68.33 41.53 22.49 11.68 8.25 11.68 22.49 41.53 68.33 100.00
    x=:  0.00  0.10  0.20  0.30  0.40 0.50  0.60  0.70  0.80  0.90   1.00
  JMAX=11,DELX=0.100,DELT=0.500E+03,S=0.500,ALPHA=0.100E-05, ↳
RM=0.942E+00
      ЧИСЛЕННОЕ ЗНАЧЕНИЕ ТЕМПЕРАТУРЫ T(x,t) В СТЕРЖНЕ
                    ПО СХЕМЕ ВВЦП
 t=500: 50.00 25.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 25.00  50.00
```

```
t=1000:100.00 50.00 12.50  0.00  0.00  0.00  0.00  0.00 12.50 50.00 100.00
t=1500:100.00 56.25 25.00  6.25  0.00  0.00  0.00  6.25 25.00 56.25 100.00
t=2000:100.00 62.50 31.25 12.50  3.13  0.00  3.13 12.50 31.25 62.50 100.00
t=2500:100.00 65.63 37.50 17.19  6.25  3.13  6.25 17.19 37.50 65.63 100.00
t=3000:100.00 68.75 41.41 21.88 10.16  6.25 10.16 21.88 41.41 68.75 100.00
      ТОЧНОЕ ЗНАЧЕНИЕ ТЕМПЕРАТУРЫ В ПОСЛЕДНЕМ СЛОЕ:
t=3000:100.00 68.33 41.53 22.49 11.68 8.25 11.68 22.49 41.53 68.33 100.00
   x=:  0.00  0.10  0.20  0.30  0.40  0.50  0.60  0.70  0.80  0.90   1.00
  JMAX=11,DELX=0.100,DELT=0.500E+03,S=0.500,ALPHA=0.100E-05,↳
RM=0.942E+00
```

Погрешность численного решения по данным листинга 6.1 составит $r_m = 0.942$.

6.1. Понятия согласованности, устойчивости и сходимости разностной схемы

Очевидная формулировка условия *сходимости* численного решения к точному для схемы ВВЦП – это стремление Δx, $\Delta t \to 0$ по мере измельчения сетки. Однако для формулировки более полного условия сходимости необходимо рассмотреть понятия *согласованности* и *устойчивости*.

Согласованной разностной называется схема, процесс дискретизации которой может быть восстановлен до исходного уравнения.

Под *устойчивостью* будем понимать устойчивость алгоритма решения алгебраических уравнений для получения численного решения.

Таким образом, условие сходимости может быть сформулировано в следующем виде

СОГЛАСОВАННОСТЬ + УСТОЙЧИВОСТЬ = СХОДИМОСТЬ

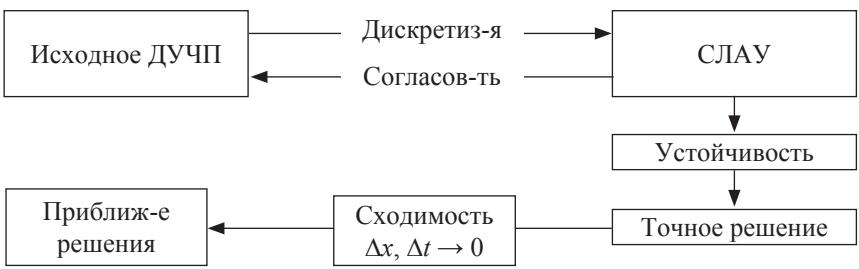


Рис. 6.2

Решение системы алгебраических уравнений (СЛАУ), аппроксимирующее данное ДУЧП, называется *сходящимся*, если выполнено следующее условие:

$$\tilde{T}_j^n \to T\left(x_j, t_n\right) \text{ при } \Delta x \to 0,\ \Delta t \to 0.$$

Разность между точным решением ДУЧП и точным решением СЛАУ называется *ошибкой решения*, которая обозначается так:

$$e_j^n = T\left(x_j, t_n\right) - \tilde{T}_j^n. \qquad (6.1.1)$$

Решение СЛАУ *точное*, если оно не содержит характерных вычислительных ошибок, связанных с представлением данных в компьютере (например, округление).

Теорема Лакса. Если имеется корректно поставленная линейная задача с начальными условиями и конечно-разностная аппроксимация к этой задаче, удовлетворяющая условию согласованности, то устойчивость является необходимым и достаточным условием сходимости.

Рассмотрим изменение среднеквадратичной ошибки решения при измельчении сетки для уравнения теплопроводности, рассчитанное по программе GDIFF1.

Таблица 6.1

S	Среднеквадратичная ошибка r_m			
	$\Delta x = 0.2$	$\Delta x = 0.1$	$\Delta x = 0.05$	$\Delta x = 0.25$
0.6	4.070	1.720	120.0	$5.56\text{x}10^{11}$
0.5	3.930	0.942	0.230	0.058
0.3	0.127	0.362	0.091	0.023
1/6	6.28×10^{-2}	5.77×10^{-3}	3.03×10^{-4}	3.49×10^{-5}

Решения были построены для сеток с числом узлов по $x = 5, 10, 20, 40$ соответственно. Из *табл. 6.1* видно, что среднеквадратичная ошибка уменьшается примерно пропорционально Δx^2, за исключением случаев $s = 0.6$, $s = 1/6$, которые будут рассмотрены ниже. Измельчение сетки, согласно таблице, должно приводить к уменьшению среднеквадратичной ошибки. Таким образом, анализируя данные *табл. 6.1*, можно сделать вывод о том, что при $\Delta x \to 0$ решение алгебраических уравнений должно сходится к точному.

Согласованность СЛАУ с исходным дифференциальным уравнением является необходимым условием для сходимости численного решения к точному. Является ли оно достаточным? При измельчении сетки, когда размеры ее ячеек стремятся к нулю, СЛАУ должна быть эквивалентна исходному ДУЧП.

Однако, если для схемы ВВЦП положить $s > 0.5$, то при измельчении сетки решение будет быстро расходиться, что подтверждается данными табл. 6.1.

Таким образом, для проверки на согласованность необходимо подставить точное решение в СЛАУ и разложить узловые значения искомой функции в ряд Тейлора, в окрестности фиксированного узла (j, n). В результате получается выражение, состоящее из двух частей: исходное ДУЧП и остаточный член. При этом остаточный должен стремиться к нулю при измельчении сетки. Рассмотрим, как это можно сделать, применительно к схеме ВВЦП.

Подставим точное решение в виде T_j^n в рекуррентное соотношение (6.2) для численного решения уравнения теплопроводности:

$$T_j^{n+1} = sT_{j-1}^n + (1-2s)T_j^n + sT_{j+1}^n. \tag{6.1.2}$$

Необходимо определить, насколько близко выражение (6.1.2) соответствует уравнению диффузии (5.1). Разложив каждый член выражения (6.1.2) в точке (j, n), в итоге получим:

$$\left[\frac{\partial T}{\partial t}\right]_j^n - \alpha\left[\frac{\partial^2 T}{\partial x^2}\right]_j^n + E_j^n = 0, \tag{6.1.3}$$

где

$$E_j^n = 0.5\Delta t\left[\frac{\partial^2 T}{\partial t^2}\right]_j^n - \alpha\left(\frac{\Delta x^2}{12}\right)\left[\frac{\partial^4 T}{\partial x^4}\right]_j^n + O\left(\Delta t^2, \Delta x^4\right). \tag{6.1.4}$$

В уравнении (6.1.3) ошибка аппроксимации E_j^n есть величина, отличающая ДУЧП (6.1.3) от исходного ДУЧП (5.1) и связана с дискретизацией производных $\partial T / \partial t$, $\partial^2 T / \partial x^2$ (где аппроксимация на порядок выше).

При стремлении $\Delta x \to 0$, $\Delta t \to 0$ величина $E_j^n - > 0$ в узле (j, n). В результате алгоритм ВВЦП стремится к уравнению диффузии, что и определяет согласованность.

Заметим, что общая ошибка, которую мы допускаем при отбрасывании членов тейлоровского разложения в алгоритме ВВЦП составляет $O(\Delta t, \Delta x^2)$. Таким образом, ее стремление к нулю медленнее, чем в (6.1.4), т.е. сходимость медленнее, чем согласованность. Покажем, как можно получить другую оценку, переписав уравнение диффузии иным образом. Если в ДУЧП (5.1) слева и справа взять производные по t, то функция T, удовлетворяющая уравнению диффузии (5.1), удовлетворяет и такому уравнению

$$\frac{\partial^2 T}{\partial t^2} = \alpha \frac{\partial}{\partial t}\frac{\partial^2 T}{\partial x^2} = \alpha \frac{\partial^2}{\partial x^2}\frac{\partial T}{\partial t} = \alpha^2 \frac{\partial^4 T}{\partial x^4}. \qquad (6.1.5)$$

Следовательно, схема ВВЦП (6.1.2) также согласуется с ДУЧП (6.1.5), с общей ошибкой аппроксимации порядка $O(\Delta t^2, \Delta x^4)$. При этом остаточный член можно переписать так

$$E_j^n = 0.5\alpha\Delta x^2 \left(s - \frac{1}{6} \right) \left[\frac{\partial^4 T}{\partial x^4} \right]_j^n + O\left(\Delta t^2, \Delta x^4 \right). \qquad (6.1.6)$$

Если $s = 1/6$, первый член в (6.1.6) обращается в нуль, ошибка аппроксимации имеет порядок $O(\Delta t^2, \Delta x^4)$. Если же s – фиксировать, то порядок ошибки $O(\Delta x^4)$. В этом случае при $\Delta x \to 0$, $\Delta t \to 0$ ошибка аппроксимации стремится к нулю быстрее, чем для любого другого значения s, что хорошо иллюстрируется данными *табл. 6.2*.

Таким образом, схема ВВЦП обладает замечательным свойством — она может обеспечить четвертый порядок точности только лишь путем выбора специального значения параметра дискретизации s.

Устойчивость или неустойчивость разностной схемы иногда можно определить непосредственно из ее результатов. Другими словами, физически нереальных результатов. Например, рассмотрим схему ВВЦП для уравнения теплопроводности на следующей сетке (*рис. 6.3*).

Пусть в момент времени t $\tilde{T}_{j+1}^n = \tilde{T}_{j-1}^n = 100C^0$, а $\tilde{T}_j^n = 0$. Если $s > 1/2$, то температура на следующем временном слое окажется выше, чем в соседних узлах, что нереально $s = 1$:

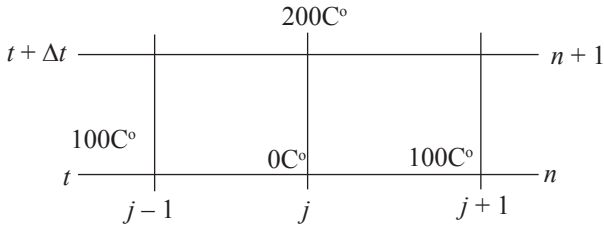


Рис. 6.3

$$\tilde{T}_j^{n+1} = s\tilde{T}_{j-1}^n + (1-2s)\tilde{T}_j^n + s\tilde{T}_{j+1}^n = 1.0\times100.0–0.0+1.0\times100.0 = 200C^0.$$

В итоге получаем противоречащее законам физики изменение температуры.

Типично неустойчивое решение уравнения теплопроводности можно получить уже при $s = 0.6$. Данные *табл. 6.2* это подтверждают.

Понятие устойчивости связано, прежде всего, с действием ошибок, вносимых в расчет в любой узловой точке. Ошибки эти связаны с ограниченными возможностями компьютера. Компьютерные данные представляются с конечным числом цифр после запятой, что неизбежно приводит к ошибке округления. Поэтому для алгоритма ВВЦП целесообразно рассматривать величину $\dot{T}_j^n$, связанную с численным решением СЛАУ, вместо значения $\tilde{T}_j^n$, связанного с дискретизацией. Говорят, что конкретный метод считается устойчивым, если суммарная доля ошибок, вносимых в расчет при использовании этого метода, пренебрежимо мала.

Рассмотрим ошибки, вносимые в узловых точках (j, n):

$$\xi_j^n = \tilde{T}_j^n - \dot{T}_j^n. \tag{6.1.7}$$

Величину (6.1.7) на практике рассчитать достаточно трудно, однако существует возможность получить ее оценку.

Для СЛАУ, полученных в результате дискретизации, можно показать, что ошибка, связанная с компоненты этих уравнений, определяется из тех же алгебраических уравнений. Например, для схемы ВВЦП можем записать:

$$\dot{T}_j^{n+1} = s\dot{T}_{j-1}^n + (1-2s)\dot{T}_j^n + s\dot{T}_{j+1}^n. \tag{6.1.8}$$

Заметив, что точное решение СЛАУ удовлетворяет $\tilde{T}_j^n$, вычислим ошибки по формуле (6.1.7), учитывая выражение (6.1.8):

$$\xi_j^{n+1} = s\xi_{j-1}^n + (1-2s)\xi_j^n + s\xi_{j+1}^n. \tag{6.1.9}$$

Если заданы начальные (по *j)* и граничные (по *n*) условия, то все начальные ошибки $\xi_j^0, j = 2,3..,JMAX-1$ и граничные ошибки $\xi_1^n, \xi_{JMAX}^n;\quad n = 0,1,..,NMAX$, будут равны нулю.

Если в результате расчета функции *T* в каком-либо внутреннем узле (*j*, *n*) не будут внесены ошибки округления, то и результирующие ошибки будут нулевыми.

Для анализа устойчивсости разностных схем, применяются различные методы. Мы рассмотрим два основных — матричный метод и метод Фурье.

Суть *матричного метода* заключается в том, что система уравнений, описывающая изменение ошибки, представляется в матричной форме, после чего проводится анализ собственных значений полученной матрицы.

Подставим в (6.1.9) значения *j* и, учтя, что на границах ошибки равны нулю $\left(\xi_1^n = \xi_j^n = 0\,\forall n\right)$, получим:

$$\begin{aligned}
\xi_2^{n+1} &= (1-2s)\xi_2^n + s\xi_3^n,\\
\xi_3^{n+1} &= s\xi_2^n + (1-2s)\xi_3^n + s\xi_4^n,\\
\xi_j^{n+1} &= s\xi_{j-1}^n + (1-2s)\xi_j^n + s\xi_{j+1}^n,\\
\xi_{JMAX-1}^{n+1} &= s\xi_{JMAX-2}^n + (1-2s)\xi_{JMAX-1}^n.
\end{aligned} \tag{6.1.10}$$

В матричном виде

$$\{\bar{\xi}^{n+1}\} = [A]\{\bar{\xi}^n\}, \tag{6.1.11}$$

$$A = \begin{bmatrix}
(1-2s) & s & & & \\
s & (1-2s) & s & & \\
 & s & . & s & \\
 & & . & . & . \\
 & & & s & (1-2s)
\end{bmatrix}, \quad \bar{\xi}^n = \begin{Bmatrix}
\xi_2^n \\ . \\ . \\ \xi_j^n \\ . \\ . \\ \xi_{JMAX-1}^n
\end{Bmatrix}. \tag{6.1.12}$$

Известно, что при увеличении n вектор ошибок $\overline{\xi}^n$ имеет ограниченную норму, если все собственные значения λ_m матрицы A различны и ограничены 1

$$\left|\lambda_m\right| \le 1, \forall \lambda. \tag{6.1.13}$$

Собственные значения трехдиагональной матрицы вычисляются по формуле

$$\lambda_m = 1 - 4s\sin^2\left(\frac{m\pi}{2\left(JMAX-1\right)}\right),\ m = 1,2,...,JMAX-2. \tag{6.1.14}$$

Используя условие устойчивости (7.13) можем записать

$$-1 \le 1 - 4s\sin^2\left(\frac{m\pi}{2\left(JMAX-1\right)}\right) \le 1. \tag{6.1.15}$$

Правая часть (6.1.15) автоматически удовлетворяется для всех m и s.

Для выполнения неравенства по левой части, необходимо выполнение следующего условия

$$s\sin^2\left(\frac{m\pi}{2\left(JMAX-1\right)}\right) \le \frac{1}{2}, \tag{6.1.16}$$

что выполняется для всех m при $s \le \frac{1}{2}$. Из этого можно сделать вывод, что алгоритм ВВЦП является устойчивым, если $s \le \frac{1}{2}$.

Данные *табл. 6.1*, показывают, что решение сходится быстрее всего именно при этих значения s. Таким образом, условие (6.1.16) подтверждается на практике.

Метод Неймана анализа устойчивости состоит в том, что ошибки, в одном временном слое (в узлах сетки) разлагаются в конечный ряд Фурье. Распределение ошибок на интервале $0 < x < L$ считается периодическим, при этом для удобства выкладок примем $L = 1$. Устойчивость алгоритма в этом случае определяется убыванием (устойчивый алгоритм) или возрастанием (неустойчивый алгоритм) компонент ряда Фурье. Погрешность ξ можно представить в виде суммы ряда Фурье следующим образом:

$$\xi(x,t) = \sum_m a_m(t) e^{ik_m x}. \tag{6.1.17}$$

Предположив, что начальные данные заданы неточно, выразим вектор начальной ошибки в узлах сетки с помощью комплексного ряда Фурье в точке x_j в момент времени $t = 0$ ($n = 0$) из формулы (6.1.17):

$$\xi_j^0 = \sum_{m=1}^{JMAX-2} a_m e^{i\theta_m j},\ j = 1,...,JMAX-1, \tag{6.1.18}$$

где волновое число $\theta_m = m\pi\Delta x$. Так как функция погрешности — линейна (удовлетворяет линейному уравнению (6.1.9)), то можно рассматривать поведение каждой гармоники независимо друг от друга.

Поэтому, достаточно изучить влияние только одной составляющей на распространение ошибки — члена $e^{i\theta_m j}$. Индекс m в дальнейшем будем опускать.

С учетом выражения начальной ошибки (6.1.18) решение уравнения (6.1.9) в n-ном временном слое будем искать методом разделения переменных в виде:

$$\xi_j^n = (Z)^n e^{i\theta j}, \tag{6.1.19}$$

где Z – комплексный коэффициент *возведен в степень n*.

Подставив (6.1.19) в уравнение (6.1.9), после несложных преобразований получим:

$$Z = 1 - 4s\sin^2(\theta/2). \tag{6.1.20}$$

Так как Z является функцией от s, θ, то Z зависит как от размеров сетки, так и от конкретной моды Фурье. Используя (6.1.19), можем записать, что $\xi_j^{n+1} = (Z)^{n+1} e^{i\theta j}$ или $\xi_j^{n+1} = Z\xi_j^n$. Таким образом, погрешность не будет возрастать на каждом временном слое, если потребовать выполнения условия $|Z| \le 1$.

Абсолютная величина $|Z|$ называется *избытком* и ограничивает рост ошибок, если $|Z| \le 1\ \forall\theta$ (6.1.21).

Следовательно, условие устойчивости для схемы ВВЦП можно сформулировать следующим образом:

$$-1 \le 1 - 4s\sin^2\left(\theta/2\right) \le 1, \forall\,\theta, \tag{6.1.22}$$

что выполняется при $s \le \frac{1}{2}$. Этот результат подтверждается и матричным методом.

6.2. Повышение точности решения

Как мы видели из данных *табл. 6.1*, среднеквадратичная ошибка решения убывает по мере измельчения сетки так же, как ошибка аппроксимации. Поэтому целесообразно ввести экстраполяцию по Ричардсону для улучшения точности получаемого решения.

Рассмотрим два решения уравнения диффузии $\tilde{T}_a$, $\tilde{T}_b$ с различными длинами шагов Δx_a и Δx_b, при одинаковом значении s. Составное решение $\tilde{T}$ ищется в виде:

$$\tilde{T} = a\tilde{T}_a + b\tilde{T}_b. \tag{6.2.1}$$

Решение (6.2.1) строится таким образом, чтобы выполнялось условие

$$\left(a+b\right) = 1 \left(\text{при } \tilde{T}_a \equiv \tilde{T}_b\right). \tag{*}$$

Разложим решение $\tilde{T}$ в ряд Тейлора в окрестности узла (j, n):

$$\left[\frac{\partial\tilde{T}}{\partial t}\right]_j^n - \alpha\left[\frac{\partial^2\tilde{T}}{\partial x^2}\right]_j^n + a\left[E_a\right]_j^n + b\left[E_b\right]_j^n = 0, \tag{6.2.2}$$

где по формуле (6.1.6) получим выражение остаточных членов

$$\left[E_a\right]_j^n = 0.5\alpha\Delta x_a^2\left(s-\frac{1}{6}\right)\left(\frac{\partial^4 T}{\partial x^4}\right) + O\left(\Delta x^4\right),$$

$$\left[E_b\right]_j^n = 0.5\alpha\Delta x_b^2\left(s-\frac{1}{6}\right)\left(\frac{\partial^4 T}{\partial x^4}\right) + O\left(\Delta x^4\right).$$

Если положить $\Delta x_b = \Delta x_a / 2$, то коэффициент при старшей производной можно сделать равным нулю с помощью формулы $a + 0.25b = 0$. Учитывая условие (*), получим, что $a = -1/3$, $b = -4/3$. Тогда составное решение $\tilde{T} = \frac{4}{3}\tilde{T}_b - \frac{1}{3}\tilde{T}_a$ имеет ошибку решения $O\left(\Delta x^4\right)$. Однако такое поведение решения характерно только при достаточно мелкой сетке.

Таблица 6.2

Схема ВВЦП	$\Delta x = 0.1$	$\Delta x = 0.05$	$\Delta x = 0.025$
$s = 0.5$	0.942	0.230	0.058
Экстраполяция по Ричардсону, $s = 0.5$	0.9744	0.0104	0.00634
$s = 0.167$	0.00577	0.000303	0.0000339

Справедливость этого подтверждается *табл. 6.2*. Из данных таблицы видно, что на более грубой сетке среднеквадратичная ошибка наоборот увеличивается.

6.3. Другие конечно-разностные схемы решения уравнения диффузии

Явные схемы

Ричардсон предложил усовершенствовать схему ВВЦП с аппроксимацией первой производной по времени со вторым порядком точности

$$\frac{\tilde{T}_j^{n+1} - \tilde{T}_j^{n-1}}{2\Delta t} = \frac{\alpha\left(\tilde{T}_{j-1}^n + 2\tilde{T}_j^n - \tilde{T}_{j+1}^n\right)}{\Delta x^2}, \qquad (6.3.1)$$

имеющей порядок аппроксимации $O\left(\Delta t^2, \Delta x^2\right)$. Анализ устойчивости схемы (6.3.1) по Нейману показывает, что она неустойчива при всех $s > 0$. Поэтому применение этой схемы не имеет никакого практического применения.

Дюффорт и *Франкель* модифицировали схему (6.3.1) до получения устойчивого алгоритма, заменив $\tilde{T}_j^n$ на $0.5\left(\tilde{T}_j^{n-1} + \tilde{T}_j^{n+1}\right)$. В результате получается явная схема

$$\tilde{T}_j^{n+1} = \left(\frac{2s}{1+2s}\right)\left(\tilde{T}_{j-1}^{n} + \tilde{T}_{j+1}^{n}\right) + \left(\frac{1-2s}{1+2s}\right)\tilde{T}_j^{n-1}. \qquad (6.3.2)$$

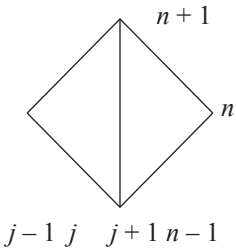


Рис. 6.4

Схема (6.3.2) с конечно-разностным шаблоном, показанным на *рис. 6.4* является условно трехслойной по времени, и при $s = 0.5$ совпадает со схемой ВВЦП. Конечно-разностный шаблон схемы требует хранения в памяти данных на двух временных слоях. Поэтому появляется необходимость в применении двухслойной схемы для первого шага по времени. Анализ устойчивости схемы по Нейману приводит к следующему выражению для коэффициента усиления Z.

$$|Z| = \frac{2s\cos\theta + \left(1 - 4s^2\sin^2\theta\right)^{\frac{1}{2}}}{1+2s}. \qquad (6.3.3)$$

Условие $|Z| < 1$ выполняется для любого значения θ при $s > 0$. Таким образом, схема (6.3.2) устойчива при любом значении Δt.
Проверка на согласованность позволяет получить следующее выражение:

$$\left[\frac{\partial T}{\partial t}\right]_j^n - \alpha\left[\frac{\partial^2 T}{\partial x^2}\right]_j^n + \alpha\left(\frac{\Delta t}{\Delta x}\right)^2\left[\frac{\partial^2 T}{\partial t^2}\right]_j^n + O\left(\Delta t^2, \Delta x^2\right) = 0. \qquad (6.3.4)$$

Для получения согласованной схемы, необходимо положить $\Delta t / \Delta x \to 0$ при $\Delta t \to 0, \Delta x \to 0$. Таким образом, требуется выполнение условия $\Delta t << \Delta x$. Множитель при третьем члене уравнения (6.3.4) равен $\alpha(\Delta t / \Delta x)^2 = s\Delta t$. При решении уравнения диффузии мы предполагали, что s всегда имеет порядок $O(1)$. Следовательно, схема Дюффорта-Франкеля согласованна с уравнением диффузии, но теряет точность, если слишком велика величина $s\Delta t$.

Неявные схемы

Неявное представление конечно-разностного аналога уравнения диффузии можно записать следующим образом:

$$\frac{\tilde{T}_j^{n+1}-\tilde{T}_j^n}{\Delta t}-\frac{\alpha\left(\tilde{T}_{j-1}^{n+1}-2\tilde{T}_j^{n+1}+\tilde{T}_{j+1}^{n+1}\right)}{\Delta x^2}=0. \tag{6.3.5}$$

Конечно-разностное представление уравнения диффузии (6.3.5) имеет первый порядок точности по времени, и второй — по пространству. Из уравнения (6.3.5) следует неявная схема

$$\tilde{T}_j^n=-s\tilde{T}_{j-1}^{n+1}+(1-2s)\tilde{T}_j^{n+1}-s\tilde{T}_{j+1}^{n+1}. \tag{6.3.6}$$

Схема (6.3.6) имеет ошибку аппроксимации относительно узла (j, n)

$$E_j^n=-\frac{\Delta t}{2}\left(1+\frac{1}{6s}\right)\left[\frac{\partial^2 T}{\partial x^2}\right]_j^n+O\left(\Delta t^2,\Delta x^4\right). \tag{6.3.7}$$

Анализ устойчивости по Нейману схемы (6.3.6) приводит к выражению

$$Z=\frac{1}{1+2s(1-\cos\theta)}. \tag{6.3.8}$$

Оценка коэффициента усиления Z в формуле (6.3.8) позволяет получить интересный результат: если $s > 0$, то $|Z| < 1$ для любых значений θ. Такой результат позволяет говорить о *безусловной устойчивости* неявной разностной схемы, по сравнению с условно устойчивыми явными схемами. Очевидно, что применение неявных разностных схем является более предпочтительным. Однако в этом случае приходится рассматривать все узлы по пространству с формированием в итоге матрицы коэффициентов неизвестной функции в каждом последующем временном слое. Это неизбежно приведет к повышению требований к объему компьютерной памяти.

После дискретизации уравнения диффузии по схеме (6.3.6) получается система линейных алгебраических уравнений вида $[A]\{\tilde{T}\}=\{b\}$, где матрица A, вектор T и вектор b имеют вид:

$$\begin{bmatrix} (1+2S) & -S & & & 0 \\ -S & (1+2S) & -S & & \\ & \cdot & \cdot & \cdot & \\ & \cdot & \cdot & \cdot & \\ & -S & (1+2S) & -S & \\ & \cdot & \cdot & \cdot & \\ & \cdot & \cdot & \cdot & \\ 0 & & -S & (1+2S) & \end{bmatrix} \begin{Bmatrix} \tilde{T}_2^{n+1} \\ \tilde{T}_3^{n+1} \\ \cdot \\ \cdot \\ \tilde{T}_j^{n+1} \\ \cdot \\ \cdot \\ \tilde{T}_{JMAX-1}^{n+1} \end{Bmatrix} = \begin{Bmatrix} b_2 \\ b_3 \\ \cdot \\ \cdot \\ b_j \\ \cdot \\ \cdot \\ b_{JMAX-1} \end{Bmatrix}. \qquad (6.3.9)$$

Компоненты вектора b в (6.3.9) определяются следующим образом $b_2 = \tilde{T}_2^n + s\tilde{T}_1^{n+1}$, $b_j = \tilde{T}_j^n$, $b_{JMAX-1} = \tilde{T}_{JMAX-1}^n + s\tilde{T}_{JMAX}^{n+1}$. Значения $\tilde{T}_1^{n+1}, \tilde{T}_{JMAX}^{n+1}$ берутся из граничных условий. Матрица A в (6.3.9) является трехдиагональной, поэтому для решения системы уравнений (6.3.9) лучше использовать методы, учитывающие ленточную структуру матрицы.

Для компьютерного решения неявной системы (6.3.9) требуется примерно вдвое больше времени, чем, например, для явной схемы ВВЦП.

Схема Кранка-Николсона аппроксимирует пространственную производную на промежуточном временном слое $n = (n + 1)/2$. Схема имеет второй порядок точности по времени и второй порядок по пространству, т.е. ошибку усечения порядка $O(\Delta t^2, \Delta x^2)$, что значительно точнее неявной схемы, а также схемы ВВЦП, имеющих ошибку $O(\Delta t, \Delta x^2)$. Шаблон схемы Кранка-Николсона представлен на *рис. 6.5*, а ее выражение имеет вид:

$$\frac{\tilde{T}_j^{n+1} + \tilde{T}_j^n}{\Delta t} - \alpha \left[(1-\beta) \frac{\tilde{T}_{j-1}^n - 2\tilde{T}_j^n + \tilde{T}_{j+1}^n}{\Delta x^2} + \beta \frac{\tilde{T}_{j-1}^{n+1} - 2\tilde{T}_j^{n+1} + \tilde{T}_{j+1}^{n+1}}{\Delta x^2} \right] = 0, \quad (6.3.10)$$

где $0 \le \beta \le 1$. Форма записи (6.3.10) удобна тем, что для различных значений β можно получить различные конечно-разностные схемы:

$\beta = 0$ — схема ВВЦП,

$\beta = 0.5$ — схема Кранка-Николсона,

$\beta = 0.5$ — чисто неявная схема.

Анализ устойчивости схемы (6.3.10) по Нейману показывает, что она располагается на предельной границе безусловной устойчивости, т.е. устойчивое решение получается при следующих значениях:

$\Delta t \le \dfrac{0.5\Delta x^2}{\alpha\left(1-2\beta\right)}$, если $0 \le \beta \le \dfrac{1}{2}$; Δt — без огрнаничений, если $\dfrac{1}{2} \le \beta \le 1$.

7. Разностные схемы для решения уравнения конвекции

Задачи, связанные с проникновением тепла в среду (параболическое уравнение диффузии), могут быть успешно решены путем изучении поведения членов вида $\dfrac{\partial^2 T}{\partial x^2}$ на основе описанных выше схем. Для задач связанных с выделением тепла средой (гиперболическое уравнение конвекции) аналогичные методы применимы также и для изучения поведения членов вида $\dfrac{\partial T}{\partial x}$. Рассмотрим особенности построения вычислительных схем для гиперболических уравнений на примере одномерного уравнения конвекции.

Запишем уравнение конвекции (7.1) с начальным условием (7.2)

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} = 0; \tag{7.1}$$

$$T\left(x,0\right) = F\left(x\right), \tag{7.2}$$

где u — скорость распространения теплового фронта, T — температура, F — некоторая известная функция. Для случая постоянной и положительной величины u запишем общее решение уравнения (7.1)

$$T\left(x,t\right) = F\left(x-ut\right). \tag{7.3}$$

Если $\exists F, \forall\ x \in \left(-\infty,\infty\right)$, то решение T в точке (x_1, t_1) совпадает с решением T в точке ($x_1 - ut_1$, 0), в момент времени $t = 0$. Другими словами, решение T остается постоянным вдоль линии характеристик уравнения (7.1), поэтому можно записать

$$T\left(x_1,t_1\right) = F\left(x_1-ut_1\right) = T\left(x_1-ut_1,0\right). \tag{7.4}$$

Для решения уравнения (7.1) используем алгоритм ВВЦП, а соответствующая схема ВВЦП для уравнения конвекции будет иметь вид

$$\frac{\tilde{T}_j^{n+1} - \tilde{T}_j^n}{\Delta t} + \frac{u\left(\tilde{T}_{j+1}^n - \tilde{T}_{j-1}^n\right)}{2\Delta x} = 0. \tag{7.5}$$

Выражение (7.4) можно переписать в виде удобного алгоритма

$$\tilde{T}_j^{n+1} = \tilde{T}_j^n - \frac{1}{2} C\left(\tilde{T}_{j+1}^n - \tilde{T}_{j-1}^n\right), C = u\frac{\Delta t}{\Delta x}. \tag{7.6}$$

Величина C в (7.6) называется *числом Куранта*. Точность представления выражением (7.5) исходного уравнения конвекции (7.1) имеет ошибку порядка $O(\Delta t, \Delta x^2)$. Анализ устойчивости по Нейману алгоритма (7.6) приводит к выражению $Z = 1 - \imath C \sin\theta$. Очевидно, что $|Z| \geq 1$ для всех θ. Поэтому схема (7.6) будет, безусловно, неустойчивой. Следует заметить также, что свойство неустойчивости схемы ВВЦП имеет место и применительно к уравнению диффузии, где она является лишь условно устойчивой.

Для получения устойчивой схемы, вместо центральных разностей для аппроксимации члена $\frac{\partial T}{\partial x}$ можно использовать разности назад, предполагая, что величина u — положительна. Полученная схема имеет вид

$$\frac{\tilde{T}_j^{n+1} - \tilde{T}_j^n}{\Delta t} + u\frac{\tilde{T}_j^n - \tilde{T}_{j-1}^n}{\Delta x} = 0. \tag{7.7}$$

Схему (7.7) удобнее представить в следующем виде

$$\tilde{T}_j^{n+1} = \left(1 - C\right)\tilde{T}_j^n + C\tilde{T}_{j-1}^n. \tag{7.8}$$

Положив $C = 1$ в формуле (7.7), можем записать $\tilde{T}_j^{n+1} = \tilde{T}_{j-1}^n$. (7.9)

Равенство (7.9) совпадает с точным решением (7.4) уравнения (7.1). Так как значение T в узле j в последующем слое получается из значения T для узла $j - 1$ в предыдущем слое (в направлении, вверх по потоку), то схема (7.8) получила название *разностей против потока*.

Если величина u — отрицательна, то для аппроксимации производной $\frac{\partial T}{\partial x}$ необходимо использовать разности вперед, что приведет к схеме

$$\tilde{T}_j^{n+1} = \left(1 - |C|\right)\tilde{T}_j^n + |C|\,\tilde{T}_{j+1}^n. \qquad (7.10)$$

Анализ устойчивости схемы (7.7) по Нейману показывает, что она устойчива, если выполнено условие $C \le 1$. (7.11)

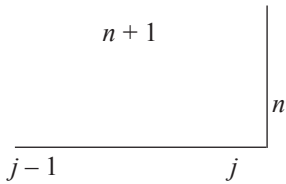


Рис. 7.1

Конечно — разностный шаблон схемы разностей против потока показан на *рис. 7.1*. Алгоритм (7.8) начинает работать с узла $j = 0$ до $j = j$ MAX – 1, по двуслойной схеме. Информация для слоя $n = 0$ берется из начальных данных (7.2). Для проверки на согласованность разностной схемы (7.8) с исходным уравнением конвекции (7.1) подставим точное решение в виде T_j^n в формулу (7.8) и разложим в ряд Тейлора в окрестности узла (j, n) каждый член полученного выражения до второй производной включительно. Тогда окончательно запишем

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} + 0.5\Delta t\frac{\partial^2 T}{\partial t^2} - 0.5u\Delta x\frac{\partial^2 T}{\partial x^2} + O\left(\Delta t^2, \Delta x^2\right). \qquad (7.12)$$

Выражение (7.12) показывает, что схема (7.8) согласуется с исходным уравнением конвекции с ошибкой аппроксимации $O(\Delta t, \Delta x)$. Для исключения в выражении (7.12) члена со второй производной по времени, перепишем уравнение (7.1):

$$\frac{\partial T}{\partial t} = -u\frac{\partial T}{\partial x},\ \frac{\partial^2 T}{\partial t^2} = u^2\frac{\partial^2 T}{\partial x^2}. \qquad (7.13)$$

Тогда, уравнение (7.12) с учетом (7.13) можно переписать так:

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} - 0.5u\Delta x\left(1 - C\right)\frac{\partial^2 T}{\partial x^2} + O\left(\Delta t^2, \Delta x^2\right) = 0. \qquad (7.14)$$

Если схема (7.8) удовлетворяет уравнению конвекции (7.1) с погрешностью аппроксимации $O(\Delta t, \Delta x)$, то, как следует из (7.14), схема (7.8) удовлетворяет с погрешностью $O(\Delta t^2, \Delta x^2)$ следующему уравнеию:

$$\frac{\partial T}{\partial t} + u\frac{\partial T}{\partial x} - 0.5u\Delta x\left(1 - C\right)\frac{\partial^2 T}{\partial x^2} = 0. \qquad (7.15)$$

Полученный результат означает, что схема с разностями против потока вносит *искусственную диффузию* или просто *численную диффузию* в уравнение конвекции. Очевидно, что выбор $C = 1$ позволяет исключить это явление, тем более что при таком выборе мы получаем алгоритм (7.9), соответствующий точному решению. Однако такое поведение решения будет сохраняться только в случае линейного уравнения (7.1).

Схема, альтернативная разностям против потока может быть получена методом, аналогичным схеме Дюффорта-Франкеля для уравнения диффузии, т.е. заменой в схеме ВВЦП (7.6) члена $\tilde{T}_j^n$ на $\frac{1}{2}\left(\tilde{T}_{j+1}^n + \tilde{T}_{j-1}^n\right)$, что в итоге приводит к схеме Лакса:

$$\tilde{T}_j^{n+1} = \frac{1}{2}\left(\tilde{T}_{j-1}^n + \tilde{T}_{j+1}^n\right) - \frac{1}{2}C\left(\tilde{T}_{j+1}^n - \tilde{T}_{j-1}^n\right). \qquad (7.16)$$

Ошибка аппроксимации схемой (7.16) исходного уравнения конвекции (7.1) имеет вид $O(\Delta t, \Delta x^2 / \Delta t)$. Схема Лакса не всегда удовлетворяет условию согласованности, так как величина $\Delta x^2 / \Delta t$ не обязательно стремится к нулю при $\Delta t, \Delta x \to 0$. Однако, если C постоянно при $\Delta t, \Delta x \to 0$, то условие согласованности выполняется.

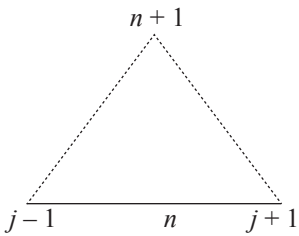


Рис. 7.2

Конечно — разностный шаблон схемы Лакса показан на *рис. 7.2*. Первая часть разности в правой части схемы (7.16) является средним значением температуры на предыдущем шаге по времени, а вторая — центрально-разностное представление первой производной по пространству. Применив к схеме Лакса анализ устойчивости по Нейману, получим коэффициент усиления $Z = \cos\theta - \imath C \sin\theta$. Очевидно, что для выполнения условия $|Z| \le 1$ необходимо положить $C \le 1$. (7.17)

Условие (7.11) и (7.17) являются важным критерием устойчивости *Куранта-Фридрихса-Леви (КФЛ)* или просто *условием Куранта*. Уравнение конвекции (7.1) можно рассматривать, как задачу с начальным условием (7.2). Для ДУЧП с начальным условием можно показать, что аналитическое решение в последующем временном слое зависит от информации внутри некоторой области зависимости на предыдущих слоях.

Численное решение имеет собственную область зависимости, определяемую выбором узлов на одном временном слое, значения в которых используются для вычислений значения функции в другом слое. Условие устойчивости Куранта в этом случае соблюдается, если область зависимости аналитического решения лежит в области зависимости численного решения, что подробно иллюстрирует *рис. 7.3*.

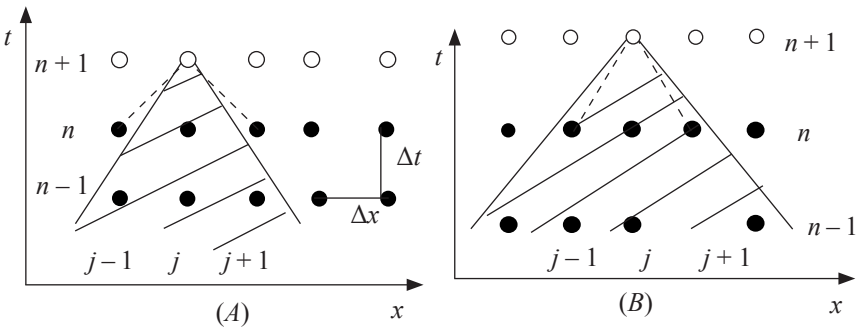


Рис. 7.3

Ситуация, показанная на *рис. 7.3* (*A*) характеризует типично устойчивое решение по Куранту. Заштрихованная часть показывает область зависимости для аналитического решения, линии, изображенные пунктиром, характеризуют область зависимости для численного решения (например, для схемы Лакса). Если область зависимости аналитического решения шире области зависимости численного решения, то решение является неустойчивым по Куранту, как показано на *рис. 7.3* (*B*).

Величина C в формуле (7.6) как раз определяет тангенс угла наклона образующей области численного решения. Тангенс угла наклона образующей области аналитического решения может быть определен из характеристик уравнения (7.1).

Для тестирования описанных выше методов проведем вычисления на примере задачи линейной конвекции усеченной синусоидальной волны. В этой задаче уравнение (7.1) должно решаться при следующих начальном

$$T(x,0)=\begin{cases}\sin 10\pi x, & 0\le x\le 0.1\\ 0, & 0.1<x\le 1\end{cases}, \qquad (7.18)$$

и граничных условиях

$$T(0,t)=0, T(1,t)=0. \tag{7.19}$$

Задача (7.1)–(7.19) имеет точное решение вплоть до значений $t = 0.9 / u$:

$$T=\begin{cases}0, & 0\le x\le ut,\\ \sin\left[10\pi\left(x-ut\right)\right], & ut<x\le ut+0.1\,.\\ 0, & ut+0.1<x\le 1.0\end{cases} \tag{7.20}$$

Листинг компьютерной программы GDIFF2, реализующей разности против потока в виде функции DAFproc(), приведен ниже

Листинг 6.1. Программа GDIFF2

```
1  subroutine DAFproc(id)
2  use task
3  integer(4)                  :: nstep=1, id, tmp
4  real(8)                     :: d(3),var
5  real(8)                     :: aa, bb, cc
6  tmp=id
7  cvs='f6.2'
8  write(cvn,'(i3)')jmax
9  fmt='('//'a3'//','//'f6.2'//','//'a4'//','//cvn//cvs//')'
10 fmt2= '('//'1x'//','//'a3'//','//'10x'//','//cvn//cvs//')'
11 allocate(td(jmax),x(jmax),te(jmax))
12 write(3,'(/,a90)')"THE DIFFERENCE AGAINST FLOW ↳
SCHEME USING"
13 jmap=jmax-1; ajm=DBLE(jmap)
14 delx=L/ajm; delt=Curant*delx/Velocity;Pi=datan(1.D0)*4.D0
15 do j=1, jmax
16   aj=j-1
17   x(j)=aj*delx
18   if(0.0D0<=x(j).and.x(j)<=1.D-01)td(j)=DSIN(1.0D+01*Pi*x(j))
19   if(1.D-01<x(j).and.x(j)<=L)     td(j)=0.D0
```

```
20 enddo
21 t=0.D0; n=0
22 write(3,fmt)"t=",t,"TD=",(td(i),i=1,jmax,nstep)
23 aa=1.0D0;bb=-Curant;cc=Curant
24 do while(t<tmax)
25   t=t+delt;n=n+1
26   td(1)=0.0D0
27   td(jmax)=0.0D0
28   d(2)=td(1)
29   do j=2, jmap
30          d(3)= d(2)
31          d(2)=td(j)
32          d(1)=td(j)
33    td(j)=aa*d(1)+bb*d(2)+cc*d(3)
34   enddo
35   write(3,fmt)"t=",t,"TD=",(td(i),i=1,jmax,nstep)
36   if(n>nmax)exit
37 enddo
38 sum=0.0D0
39 call CONVextra()
40 do j=1, jmax
41    dmp=te(j)-td(j)
42    sum=sum+dmp*dmp
43 enddo
44 write(3,fmt2)"X=",(x(i),i=1,jmax,nstep)
45 write(3,'(/,1x,a5,i3,a7,i3,a6,f6.3,a6,e10.3,a4,f6.3)')
46#"jmax=",jmax,"ntime=",n,"delx=",delx,"delt=",delt," C=",Curant
47 write(3,'(/,a80)')"Compare solution at the last time layer:"
48 write(3,fmt)"t=",t,"TD=",(te(i),i=1,jmax,nstep)
49 avs=sum/(1.D0+ajm); rms=dsqrt(avs)
50 write(3,'(30x,a15,e11.3,//)')"ERROR ESTIMATE=",rms
51 deallocate(td,x,te)
52 end subroutine DAFproc
```

В программе GDIFF2 в строке 14 вычисляется Δx, затем через заданное число Куранта определяется Δt. В строках 15–22 задается начальное усло-

вие и определяется решение в нулевом слое. Граничные условия задаются в строках 26–27.

В строках 24–37 реализуется собственно алгоритм (7.8). Для контроля погрешности в строках 38–43 вычисляется точное решение в последнем слое по формуле (7.20), записанное в виде функции CONVextra(), а затем в строке 49 вычисляется среднеквадратичная погрешность по формуле (6.5).

В задаче (7.1)–(7.19) примем $t = 0.8$, $u = 0.1$, $L = 1$. Вначале используем равенство (7.9) для проверки соответствия численного решения точному, приняв число Куранта $C = 1$. Уже при числе узлов jMAX = 40 схема разностей против потока дает следующий результат:

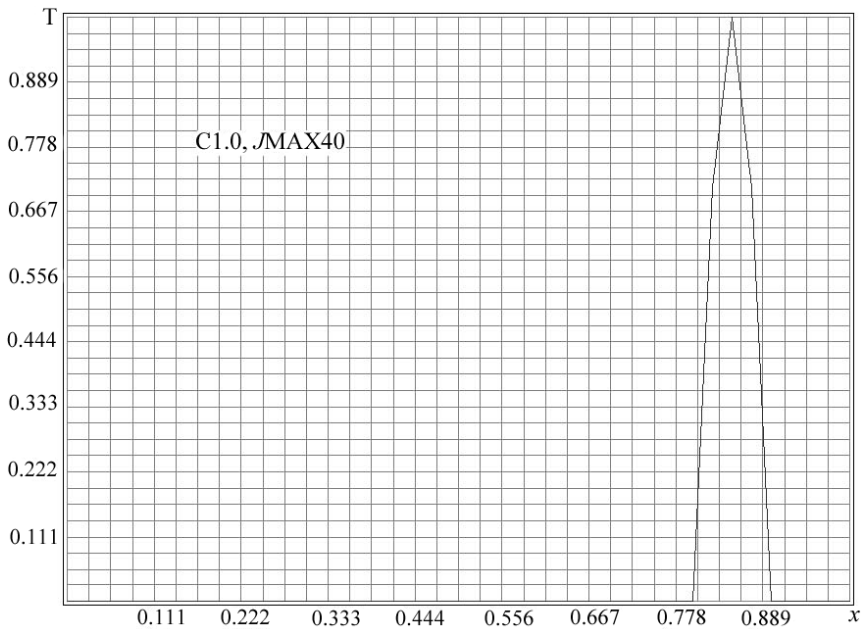
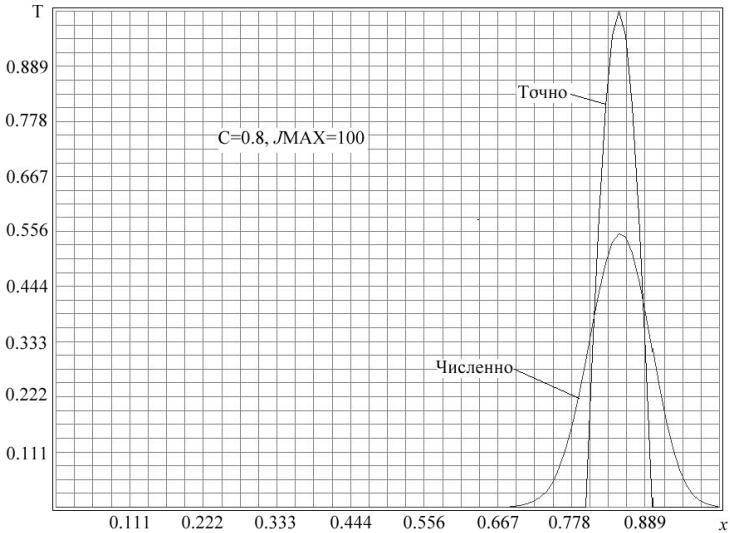


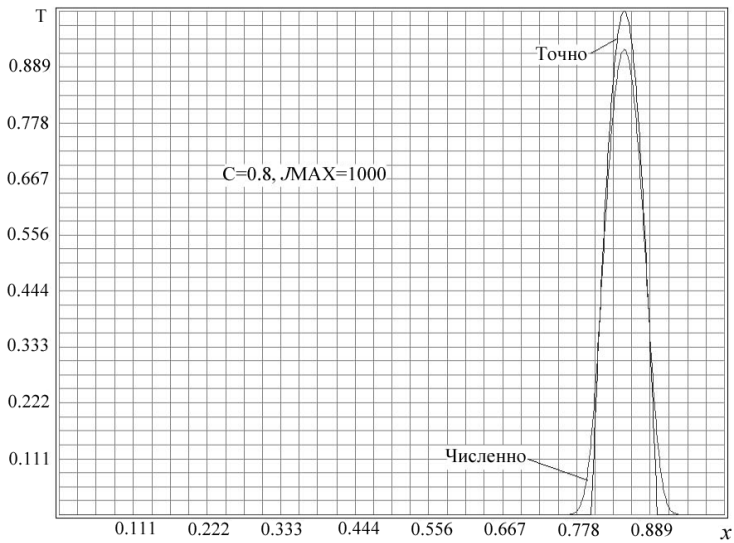
Рис. 7.4

Численное решение во всех точках соответствует точному, как видно из *рис. 7.4*. Такая картина остается в силе и в случае схемы Лакса. При этом среднеквадратичная погрешность составляет $r_m = 2.36 \cdot 10^{-16}$.

Результаты расчета при $C = 0.8$ для разностей против потока (*рис. 7.5*) и схемы Лакса (*рис. 7.6*) показывают, что метод разностей против потока оказывается несколько точнее, чем метод Лакса.

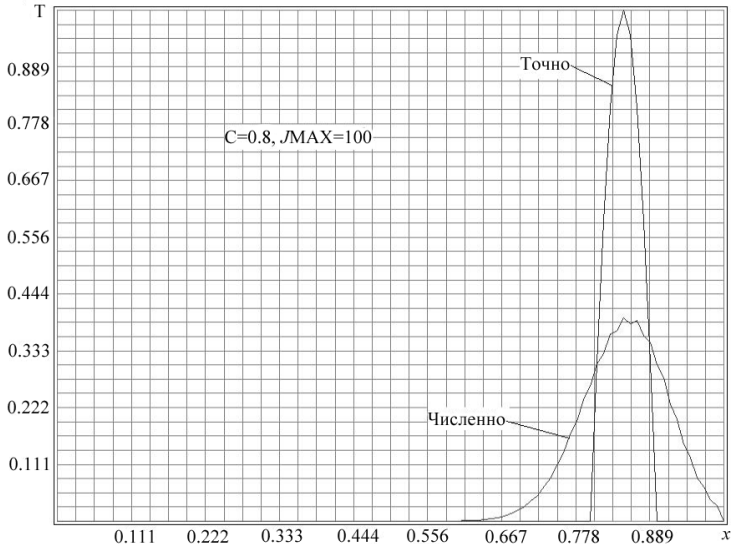

T
0.889
0.778
0.667
0.556
0.444
0.333
0.222
0.111
C=0.8, JMAX=100
Точно
Численно
0.111
0.222
0.333
0.444
0.556
0.667
0.778
0.889
x

(*A*)

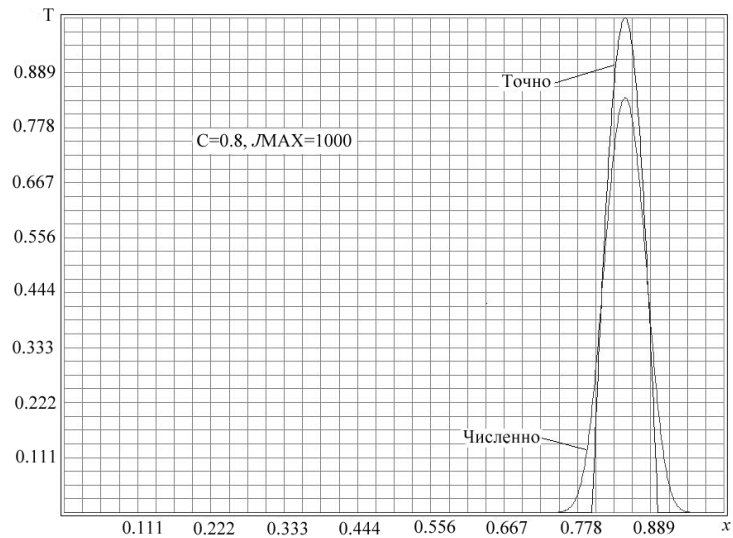

T
0.889
0.778
0.667
0.556
0.444
0.333
0.222
0.111
C=0.8, JMAX=1000
Точно
Численно
0.111
0.222
0.333
0.444
0.556
0.667
0.778
0.889
x

(*B*)

Рис. 7.5


T
0.889
0.778
0.667
0.556
0.444
0.333
0.222
0.111
C=0.8, JMAX=100
Точно
Численно
0.111
0.222
0.333
0.444
0.556
0.667
0.778
0.889
x

(*A*)


T
0.889
0.778
0.667
0.556
0.444
0.333
0.222
0.111
C=0.8, JMAX=1000
Точно
Численно
0.111
0.222
0.333
0.444
0.556
0.667
0.778
0.889
x

(*B*)

Рис. 7.6

Среднеквадратичные ошибки для разностной схемы против потока составят $r_m = 0.108,\ \ r_m = 2.48 \cdot 10^{-2}$ при числе узлов JMAX = 100, JMAX = 1000 соответственно, а для схемы Лакса их величины соответственно равны $r_m = 0.143,\ \ r_m = 4.52 \cdot 10^{-2}$. Сходимость метода разностей против потока и метода Лакса соответствует сходимости методов первого порядка точности.

Численные решения, полученные по схемам первого порядка точности, имеют гладкую форму, однако относительно точного решения они сглажены, что является следствием численной диффузии, вносимой, например, при применении схемы разностей против потока.

Для получения методов более высокого порядка точности Лакс и Вендрофф предложили схему второго порядка, разложив точное решении вблизи узла (j, n) по времени до второго члена включительно:

$$\tilde{T}_j^{n+1} \approx \tilde{T}_j^n + \Delta t \frac{\partial T}{\partial t} + \frac{1}{2}\Delta t^2 \frac{\partial^2 T}{\partial t^2}. \tag{7.21}$$

Заменив в разложение (7.21) член со второй производной по времени членом с производной по пространству из выражения (7.13), можно записать

$$\frac{\partial T}{\partial t} \approx \frac{\tilde{T}_j^{n+1} - \tilde{T}_j^n}{\Delta t} - \frac{1}{2}\Delta t u^2 \frac{\partial^2 T}{\partial x^2}. \tag{7.22}$$

Теперь, заменив член со второй производной в (7.22) его центрально-разностной аппроксимацией, получим *схему Лакса-Вендроффа*

$$\tilde{T}_j^{n+1} = \tilde{T}_j^n - 0.5C\left(\tilde{T}_{j+1}^n - \tilde{T}_{j-1}^n\right) + 0.5C^2\left(\tilde{T}_{j-1}^n - 2\tilde{T}_j^n + \tilde{T}_{j+1}^n\right). \tag{7.23}$$

Схема (7.23) согласуется с уравнением конвекции с ошибкой аппроксимации, равной $O\left(\Delta t^2, \Delta x^2\right)$. Условие устойчивости схемы Лакса-Вендроффа остается прежним — $C \le 1$. Нетрудно показать, что структура схемы Лакса-Вендроффа аналогична неустойчивой схеме ВВЦП (7.6) для уравнения конвекции с добавочным диффузионным членом и коэффициенте при нем, равным $0.5Cu\Delta x$.

В решении по схеме Лакса-Вендроффа для $C = 0.8$ (*рис. 7.7*) отчетливо прослеживаются начальная волна (основное решение) с небольшой ампли-

тудой, которая появляется раньше точного решения, и за ней вторичные волны с возрастающей амплитудой.

Решение, вызывающее появление вторичных волн, следующее за основным решением, принято называть *дисперсионным следом*. При значении $C = 1.0$ результаты расчета по схеме Лакса-Вендроффа соответствуют точному решению, так же как и схемы с разностями против потока и схемы Лакса.

Следует отметить, что при переходе к более мелким сеткам численные решения по описанным выше схемам сглаживаются и решения стремятся к точному, даже при достаточно трудном с вычислительной точки зрения, начальном условии (7.18).

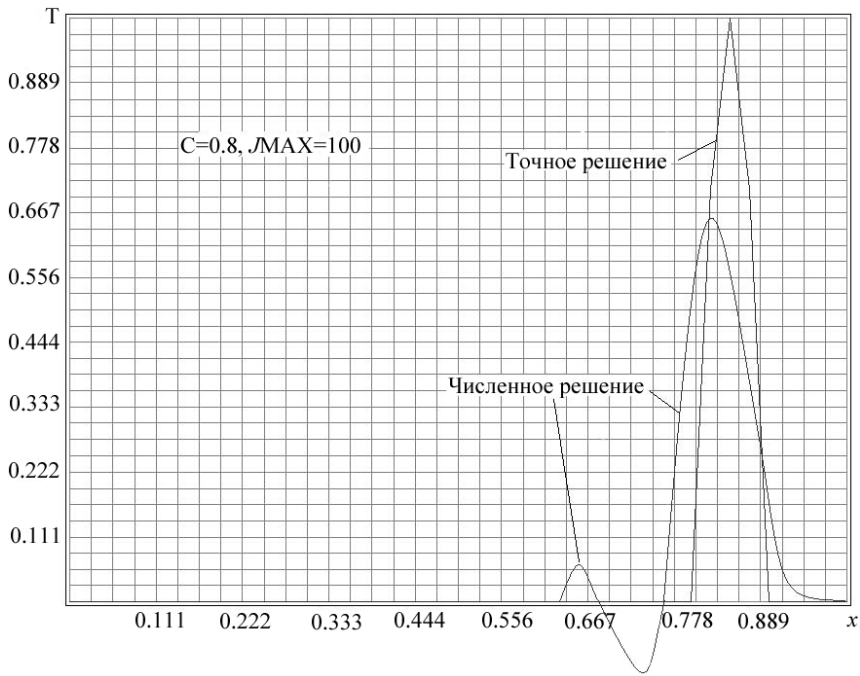


Рис. 7.7

Сравнивая методы получения численных решений для уравнения диффузии (5.1) и уравнения конвекции (7.1), можно заключить, что построение численных решений для уравнения диффузии значительно проще, чем для уравнения конвекции.

8. Разностные схемы для решения уравнения Лапласа

Примеры, рассмотренные выше, содержали время в качестве независимой переменной, и это учитывалось при построении алгоритмов. При этом можно было ограничиться применением в основном явных схем. Однако при решении стационарных задач приходится сталкиваться с дифференциальными уравнениями в частных производных эллиптического типа и неявными конечно-разностными схемами. Одним из таких примеров может послужить двумерное уравнение Лапласа (4.5).

Методы решения уравнения Лапласа отличаются не так сильно, как различаются методы решения получающейся в результате дискретизации уравнения Лапласа системы линейных алгебраических уравнений.

Рассмотрим задачу о нахождении функции u, удовлетворяющей уравнению Лапласа (4.5) в квадратной области $0 \le x \le 1, 0 \le y \le 1$. Наиболее простой функцией, являющейся решением уравнения Лапласа в этой области, может послужить функция

$$u = x^2 - y^2. \tag{8.1}$$

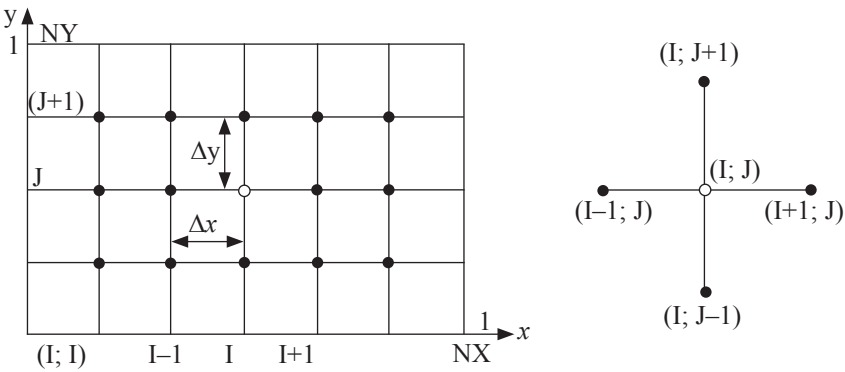


Рис. 8.1

Уравнение (4.5) требует 4 граничных условий, которые получаются подстановкой в решение (8.1) граничных точек:

$$\begin{aligned} u(0,y) &= y^2, \qquad u(x,0) = x^2, \\ u(1,y) &= 1 - y^2. \quad u(x,1) = x^2 - 1. \end{aligned} \tag{8.2}$$

Для решения краевой задачи (4.5)–(8.2) воспользуемся пятиточечной схемой Рунге, шаблон которой показан на *рис. 8.1*, и имеющей погрешность аппроксимации $O\left((\Delta x)^2, (\Delta y)^2\right)$

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{(\Delta x)^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{(\Delta y)^2} = 0, \tag{8.3}$$

или в более удобной форме

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} = 0.$$

Схема (8.3) неявная и начинает работать по шаблону, обозначенному на *рис. 8.1*, с минимальным числом узлов по x и по y равным 4. Обозначим число узлов по x – NX, число узлов по y – NY, тогда общее число узлов составит N = NX * NY, а число внутренних узлов NIN = N–2*(NX + NY) + 4. Если используется неравномерная сетка, то можно определить также и число внутренних узлов по x и y как NINX = NX – 2, NINY = NY – 2 соответственно.

Уравнения по схеме (8.3) записываются, начиная с индексов $i = 2, j = 2$ по всем внутренним узлам по x от $i = 2$ до $i = \mathrm{NX} - 1$ и всем внутренним узлам по y от $j = 1$ до $j = \mathrm{NY} - 1$:

$$\begin{aligned}
&i = 2,\ j = 2 &&: u_{3,2} + u_{1,2} + u_{2,3} + u_{2,1} - 4u_{2,2} = 0;\\
&i = 3 &&: u_{4,2} + u_{2,2} + u_{3,3} + u_{3,1} - 4u_{3,2} = 0;\\
&\ldots\ldots\ldots &&: \ldots\ldots\ldots\ldots\ldots\ldots = 0;\\
&i = \mathrm{NX} - 1 &&: u_{\mathrm{NX},2} + u_{\mathrm{NX}-2,2} + u_{\mathrm{NX}-1,3} + u_{\mathrm{NX}-1,1} - 4u_{\mathrm{NX}-1,2} = 0;\\
&i = 2,\ J = 3 &&: u_{3,3} + u_{1,3} + u_{2,4} + u_{2,2} - 4u_{2,3} = 0;\\
&i = 3 &&: u_{4,3} + u_{2,3} + u_{3,4} + u_{3,2} - 4u_{3,3} = 0;\\
&. &&: \ldots\ldots\ldots\ldots\ldots\ldots = 0;\\
&. &&: \ldots\ldots\ldots\ldots\ldots\ldots = 0;\\
&i = \mathrm{NX} - 1,\ J = \mathrm{NY\text{-}1}: &&\\
&u_{\mathrm{NX},\,\mathrm{NY}-1} + u_{\mathrm{NX}-2,\,\mathrm{NY}-1} + u_{\mathrm{NX}-1,\,\mathrm{NY}} + u_{\mathrm{NX}-1,\,\mathrm{NY}-2} - 4u_{\mathrm{NX}-1,\,\mathrm{NY}-1} = 0. &&
\end{aligned} \tag{8.4}$$

Коэффициенты в системе уравнений (8.4) с индексом «1» берутся из граничных условий (8.2).

Структура матрицы системы линейных уравнений сильно разрежена. Разреженность матрицы системы уравнений и является основной трудностью при решении уравнения Лапласа. При одинаковом числе узлов по x и по y структура матрицы коэффициентов $[A]$ системы уравнений $[A]\{u\}=\{d\}$ имеет блочную трехдиагональную форму, причем сами блоки имеют диагональную форму: трехдиагональные матрицы по главной диагонали и единичные матрицы в под-и наддиагоналях. Схематически это выглядит так

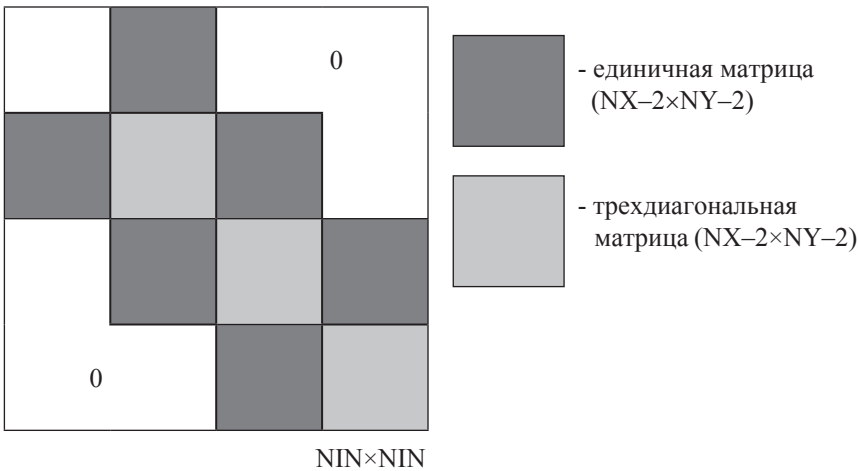


Рис. 8.2

Элементы трехдиагональной матрицы для различных значений NX×NY при различных размерах равномерной сетки выглядят следующим образом:

–4 1	–4 1 0	–4 1 0 0
1 –4	1 –4 1	1 –4 1 0
	0 1 –4	0 1 –4 1
		0 0 1 –4
4×4	5×5	6×6

Таким образом, матрица коэффициентов [*A*], например, для сетки с числом узлов NX = NY = 6 будет иметь вид:

```
-4.0  1.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0
 1.0 -4.0  1.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0  0.0  0.0
 0.0  1.0 -4.0  1.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0  0.0
 0.0  0.0  1.0 -4.0  0.0  0.0  0.0  1.0  0.0  0.0  0.0  0.0
 1.0  0.0  0.0  0.0 -4.0  1.0  0.0  0.0  1.0  0.0  0.0  0.0
 0.0  1.0  0.0  0.0  1.0 -4.0  1.0  0.0  0.0  1.0  0.0  0.0
 0.0  0.0  1.0  0.0  0.0  1.0 -4.0  1.0  0.0  0.0  1.0  0.0
 0.0  0.0  0.0  1.0  0.0  0.0  1.0 -4.0  0.0  0.0  0.0  1.0
 0.0  0.0  0.0  0.0  1.0  0.0  0.0  0.0 -4.0  1.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0  1.0 -4.0  1.0  0.0
 0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0  1.0 -4.0  1.0
 0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0  1.0 -4.0
 0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0  0.0
 0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0  0.0
 0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  0.0  1.0

 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
 1.0  0.0  0.0  0.0
 0.0  1.0  0.0  0.0
 0.0  0.0  1.0  0.0
 0.0  0.0  0.0  1.0
-4.0  1.0  0.0  0.0
 1.0 -4.0  1.0  0.0
 0.0  1.0 -4.0  1.0
 0.0  0.0  1.0 -4.0
```

Для решения систем уравнений с такой матрицей существуют специальные алгоритмы, реализованные в различных математических библиотеках. В частности, алгоритм Томаса также может быть обобщен на случай блочно-трехдиагональной матрицы системы уравнений

$$\begin{bmatrix} B_1\,C_1 & & & & & \\ A_2\,B_2\,C_2 & & & & & \\ \ldots & & & & & \\ A_i\,B_i\,C_i & & & & & \\ \ldots & & & & & \\ A_{N-1}\,B_{N-1}\,C_{N-1} & & & & & \\ A_N\;B_N & & & & & \end{bmatrix} \begin{Bmatrix} V_1 \\ V_2 \\ . \\ V_i \\ . \\ V_{N-1} \\ V_N \end{Bmatrix} = \begin{Bmatrix} D_1 \\ D_2 \\ . \\ D_i \\ . \\ D_{N-1} \\ D_N \end{Bmatrix}, \qquad (8.5)$$

где A, B, C — подматрицы размерностью M×M, а подвекторы V, D соответственно имеют M компонент. Число M соответствует NX–2 или NY–2 в случае равномерной сетки. Система (8.5), состоящая из N блоков уравнений, может быть решена по алгоритму, очень близкому к алгоритму Томаса (2.11)–(2.14). Исходная матрица (*рис. 8.2*) преобразуется к верхнетреугольной форме путем исключения подматриц A_i.

Так, для первого блока уравнений в прямом ходе имеем

$$\left[C_1'\right] = \left[B_1\right]^{-1}\left[C_1\right], \left\{D_1'\right\} = \left[B_1\right]^{-1}\left\{D_1\right\}, \qquad (8.6)$$

для i-го блока

$$\left[B_i'\right] = \left[B_i\right] - \left[A_i\right]\left[C_{i-1}'\right], \left[C_i'\right] = \left[B_i'\right]^{-1}\left[C_i\right], \left\{D_i'\right\} = \left[B_i'\right]^{-1}\left(\left\{D_i\right\} - \left[A_i\right]\left\{D_{i-1}'\right\}\right). \qquad (8.7)$$

Далее решается M — компонентная подсистема, например, относительно $\left[C_i'\right]$ решается система $\left[B_i'\right]\left[C_i'\right] = \left[C_i\right]$. После выполнения этапов (8.6)–(8.7) из уравнения (8.5) получается верхне-треугольная система при замене $\left[C_i\right]$ и $\left\{D_i\right\}$ на $\left[C_i'\right]$ и $\left\{D_i'\right\}$, а также при замене $\left[B_i\right]$ на единичную матрицу $[E]$.

В обратном ходе осуществляется подстановка по формулам

$$\left\{V_N\right\} = \left\{D_N'\right\}, \quad \left\{V_i\right\} = \left\{D_i'\right\} - \left[C_i'\right]\left\{V_{i+1}\right\}. \qquad (8.8)$$

Подходящими являются и итерационные методы, например метод последовательной верхней релаксации (ПВР). В частности, метод решения систем с блочными трехдиагональными матрицами можно найти в библи-

отеке Numerical Recipes (подпрограмма SOR). При небольшом числе узлов для решения системы линейных уравнений с разреженной матрицей можно использовать и процедуру Краута, которая при выборе главного элемента использует еще и перестановку столбцов (модули LUDCMP и LUBKSB из библиотеки Numerical Recipes).

Компьютерная программа, реализующая неявную пятиточечную схему Рунге (8.3) на языке FORTRAN, может быть записана следующим образом

Листинг 4. Программа GDIFF3

```
0  program GDIFF3
1  USE TASK
2  integer(4) ::       INDX(NIN), I,   J,    k,    l, count=0
3  double precision :: x,y,U(NIN,NIN),D(NIN),UL(NY),UR(NY),↳
UB(NX),UU(NX)
4  character fmt*20, fmt2*40, rep*5, spc*4
5  open(1,file=”laplas.dat”)
6  write(rep,’(i4)’)NIN
7  x0=0.0; y0=0.0; xN=1.0; yN=1.0
8  delX=(xN-x0)/(NX-1);delY=(yN-y0)/(NY-1)
9  fmt= ‘(‘//rep//’f4.1’//’)’
10 write(*,’(/10x,a40)’)”MATRIX OF COEFFICIENTS:”
11 !Forming the matrix of coefficients
12 U=0.0
13 do I=1, NIN
14   do J=1, NIN
15     if(I==2)then
16           k=I
17           l=1
18           do while(l<=NIN-1)
19             if(count/=NINX)then
20               U(k,l)=b
21                     else
22                       U(k,l)=0.0
23                       count=0
24                     endif
25                     count=count+1
```

```
26                      U(l,k)=U(k,l)
27                      k=k+1; l=l+1
28            enddo
29          endif
30          if(I==NINX+1)then
31            k = I
32            l = 1
33            do while(l<=NIN-NINX)
34             U(k,l)=a
35                    U(l,k)=U(k,l)
36                    k = k+1; l=l+1
37       enddo
38     endif
39          if(I==J)U(I,J)=c
40  enddo
41 enddo
42 !Forming the boundary conditions
43 x0=0.0;y0=0.0;xN=1.0;yN=1.0
44 CALL BOUNDS(UL,UR,UB,UU)
45 CALL RightHand(UL,UR,UB,UU,D)
46 !Solve SLAE
47 CALL LUDCMP(U,NIN,NIN,INDX,p)
48 CALL LUBKSB(U,NIN,NIN,INDX,D)
49 write(*,'(/20x,a45)')"CHECKING SOLUTION OF THE LAPLASE ↳
EQUATION:"
50 write(rep,'(i4)')NINX
51 fmt= '('//rep//'E11.3'//')'
52 write(*,*)"NUMERICAL SOLUTION"
53 do i=1,NINY
54   write(*,fmt)(D((i-1)*NINX+j), j=1, NINX)
55 enddo
56 write(*,*)"EXACTLY SOLUTION"
57 do i=1, NINY
58   write(*,fmt)(UE(delX*j,delY*i),j=1,NINX
59 enddo
60 write(1,*)"NUMERICAL SOLUTION"
61 do i=1,NINY
62   write(1,fmt)(D((i-1)*NINX+j), j=1, NINX)
```

```
63 enddo
64 write(1,*)"EXACTLY SOLUTION"
65 do i=1, NINY
66   write(1,fmt)(UE(delX*j,delY*i),j=1,NINX)
67 enddo
68 close(1)
69 END PROGRAM
```

Общие параметры задачи задаются в строке 1 (модуль task). Значения неизвестной функции u задаются в массиве *U*, вектор правой части — в массиве *D*. Граничные условия по левой и правой границе задаются в массивах *UL*, *UR* соответственно, а условия по верхней и нижней границе — в массивах *UU*, *UB*.

В строках 12–41 формируются значения разреженной матрицы [*A*], граничные условия формируются подпрограммой BOUNDS (вызываемой в строке 44). Подпрограмма RightHand (вызов в строке 45) формирует вектор правой части *D* из граничных условий. В строках 47–48 вызывается процедура Краута.

Типичные результаты расчета во внутренних узлах по программе GDIFF3 на расчетной сетке 9×9

Листинг 8.1.

```
                    ЧИСЛЕННОЕ РЕШЕНИЕ
 0.173E-17  0.469E-01  0.125E+00  0.234E+00  0.375E+00  0.547E+00 ↳
0.750E+00
-0.469E-01  0.408E-17  0.781E-01  0.187E+00  0.328E+00  0.500E+00 ↳
0.703E+00
-0.125E+00 -0.781E-01  0.253E-16  0.109E+00  0.250E+00  0.422E+00 ↳
0.625E+00
-0.234E+00 -0.187E+00 -0.109E+00  0.596E-16  0.141E+00  0.313E+00 ↳
0.516E+00
 0.375E+00 -0.328E+00 -0.250E+00 -0.141E+00  0.855E-16  0.172E+00 ↳
0.375E+00
-0.547E+00 -0.500E+00 -0.422E+00 -0.312E+00 -0.172E+00  0.597E-16 ↳
0.203E+00
-0.750E+00 -0.703E+00 -0.625E+00 -0.516E+00 -0.375E+00 ↳
-0.203E+00 0.168E-16
```

ТОЧНОЕ РЕШЕНИЕ

0.000E+00 0.469E-01 0.125E+00 0.234E+00 0.375E+00 0.547E+00 ↳
0.750E+00
-0.469E-01 0.000E+00 0.781E-01 0.188E+00 0.328E+00 0.500E+00 ↳
0.703E+00
-0.125E+00 -0.781E-01 0.000E+00 0.109E+00 0.250E+00 0.422E+00 ↳
0.625E+00
-0.234E+00 -0.188E+00 -0.109E+00 0.000E+00 0.141E+00 0.312E+00 ↳
0.516E+00
-0.375E+00 -0.328E+00 -0.250E+00 -0.141E+00 0.000E+00 0.172E+00 ↳
0.375E+00
-0.547E+00 -0.500E+00 -0.422E+00 -0.312E+00 -0.172E+00 0.000E+00 ↳
0.203E+00
-0.750E+00 -0.703E+00 -0.625E+00 -0.516E+00 -0.375E+00 -0.203E+00 ↳
0.000E+00

ЧИСЛЕННОЕ РЕШЕНИЕ

0.173E-17 0.469E-01 0.125E+00 0.234E+00 0.375E+00 0.547E+00 ↳
0.750E+00
-0.469E-01 0.408E-17 0.781E-01 0.187E+00 0.328E+00 0.500E+00 ↳
0.703E+00
-0.125E+00 -0.781E-01 0.253E-16 0.109E+00 0.250E+00 0.422E+00 ↳
0.625E+00
-0.234E+00 -0.187E+00 -0.109E+00 0.596E-16 0.141E+00 0.313E+00 ↳
0.516E+00
0.375E+00 -0.328E+00 -0.250E+00 -0.141E+00 0.855E-16 0.172E+00 ↳
0.375E+00
-0.547E+00 -0.500E+00 -0.422E+00 -0.312E+00 -0.172E+00 0.597E-16 ↳
0.203E+00
-0.750E+00 -0.703E+00 -0.625E+00 -0.516E+00 -0.375E+00 -0.203E+00 ↳
0.168E-16

ТОЧНОЕ РЕШЕНИЕ

0.000E+00 0.469E-01 0.125E+00 0.234E+00 0.375E+00 0.547E+00 ↳
0.750E+00
-0.469E-01 0.000E+00 0.781E-01 0.188E+00 0.328E+00 0.500E+00 ↳
0.703E+00
-0.125E+00 -0.781E-01 0.000E+00 0.109E+00 0.250E+00 0.422E+00 ↳
0.625E+00
-0.234E+00 -0.188E+00 -0.109E+00 0.000E+00 0.141E+00 0.312E+00 ↳
0.516E+00

-0.375E+00 -0.328E+00 -0.250E+00 -0.141E+00 0.000E+00 0.172E+00 ↳
0.375E+00
-0.547E+00 -0.500E+00 -0.422E+00 -0.312E+00 -0.172E+00 0.000E+00 ↳
0.203E+00
-0.750E+00 -0.703E+00 -0.625E+00 -0.516E+00 -0.375E+00 -0.203E+00 ↳
0.000E+00

Совпадение результатов численного решения с точным наблюдается во всех узловых точках, во всех значащих цифрах.

Графически, результаты расчета представлены на *рис. 8.3*.

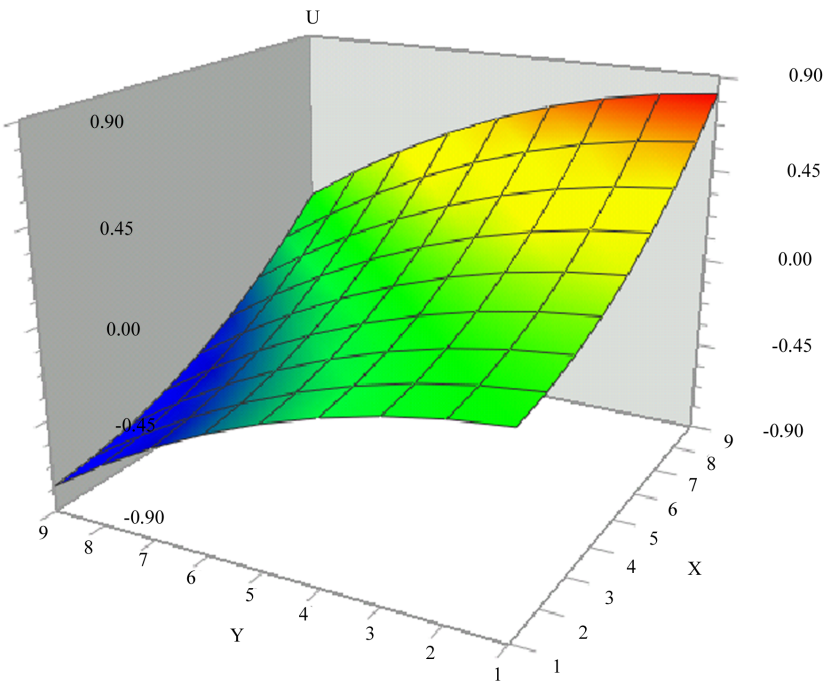


Рис. 8.3

Расчеты на неравномерной сетке дают худшие результаты. Типичные результаты расчета на неравномерной сетке 9×18 узлов представлены в листинге 8.2.

Листинг 8.2.

ЧИСЛЕННОЕ РЕШЕНИЕ

0.424E-01 0.107E+00 0.195E+00 0.307E+00 0.445E+00 0.607E+00 ↳
0.792E+00
0.505E-01 0.128E+00 0.224E+00 0.339E+00 0.474E+00 0.628E+00 ↳
0.800E+00
0.450E-01 0.132E+00 0.233E+00 0.350E+00 0.483E+00 0.632E+00 ↳
0.795E+00
0.285E-01 0.122E+00 0.227E+00 0.346E+00 0.477E+00 0.622E+00 ↳
0.779E+00
0.246E-02 0.100E+00 0.208E+00 0.327E+00 0.458E+00 0.600E+00 ↳
0.752E+00
-0.324E-01 0.681E-01 0.178E+00 0.298E+00 0.428E+00 0.568E+00 ↳
0.718E+00
-0.755E-01 0.266E-01 0.137E+00 0.258E+00 0.387E+00 0.527E+00 ↳
0.675E+00
-0.126E+00 -0.237E-01 0.877E-01 0.208E+00 0.338E+00 0.476E+00 ↳
0.624E+00
-0.185E+00 -0.825E-01 0.288E-01 0.149E+00 0.279E+00 0.417E+00 ↳
0.565E+00
-0.252E+00 -0.150E+00 -0.390E-01 0.812E-01 0.211E+00 0.350E+00 ↳
0.498E+00
-0.326E+00 -0.226E+00 -0.116E+00 0.356E-02 0.134E+00 0.274E+00 ↳
0.424E+00
-0.409E+00 -0.312E+00 -0.204E+00 -0.844E-01 0.464E-01 0.188E+00 ↳
0.341E+00
-0.501E+00 -0.407E+00 -0.302E+00 -0.184E+00 -0.521E-01 0.926E-01 ↳
0.249E+00
-0.602E+00 -0.515E+00 -0.414E+00 -0.297E+00 -0.164E+00 -0.149E-01 ↳
0.148E+00
-0.714E+00 -0.636E+00 -0.541E+00 -0.426E+00 -0.291E+00 -0.136E+00 ↳
0.358E-01
-0.840E+00 -0.775E+00 -0.688E+00 -0.575E+00 -0.438E+00 ↳
-0.275E+00-0.900E-01

ТОЧНОЕ РЕШЕНИЕ

0.122E-01 0.590E-01 0.137E+00 0.247E+00 0.387E+00 0.559E+00 ↳
0.762E+00

0.178E-02 0.487E-01 0.127E+00 0.236E+00 0.377E+00 0.549E+00 ↳
0.752E+00
-0.155E-01 0.314E-01 0.109E+00 0.219E+00 0.359E+00 0.531E+00 ↳
0.734E+00
-0.397E-01 0.714E-02 0.853E-01 0.195E+00 0.335E+00 0.507E+00 ↳
0.710E+00
-0.709E-01 -0.240E-01 0.541E-01 0.163E+00 0.304E+00 0.476E+00 ↳
0.679E+00
-0.109E+00 -0.621E-01 0.161E-01 0.125E+00 0.266E+00 0.438E+00 ↳
0.641E+00
-0.154E+00 -0.107E+00 -0.289E-01 0.804E-01 0.221E+00 0.393E+00 ↳
0.596E+00
-0.206E+00 -0.159E+00 -0.808E-01 0.285E-01 0.169E+00 0.341E+00 ↳
0.544E+00
-0.265E+00 -0.218E+00 -0.140E+00 -0.303E-01 0.110E+00 0.282E+00 ↳
0.485E+00
-0.330E+00 -0.284E+00 -0.205E+00 -0.960E-01 0.446E-01 0.216E+00 ↳
0.420E+00
-0.403E+00 -0.356E+00 -0.278E+00 -0.169E+00 -0.281E-01 0.144E+00 ↳
0.347E+00
-0.483E+00 -0.436E+00 -0.358E+00 -0.248E+00 -0.108E+00 0.642E-01 ↳
0.267E+00
-0.569E+00 -0.522E+00 -0.444E+00 -0.335E+00 -0.194E+00 -0.223E-01 ↳
0.181E+00
-0.663E+00 -0.616E+00 -0.538E+00 -0.428E+00 -0.288E+00 -0.116E+00 ↳
0.874E-01
-0.763E+00 -0.716E+00 -0.638E+00 -0.529E+00 -0.388E+00 -0.216E+00-↳
0.129E-01
-0.870E+00 -0.823E+00 -0.745E+00 -0.636E+00 -0.495E+00 -0.323E+00-↳
0.120E+00

Графически численное решение имеет несколько другую форму, представленную на *рис. 8.4*.

Схема Рунге дает хорошие результаты на равномерных сетках, однако применение ее на неравномерных сетках дает худшие результаты.

Другой схемой, имеющей более высокий порядок точности, применяемой для решения уравнения Лапласа, является девятиточечная схема. Девятиточечная схема наиболее эффективна на неравномерных сетках и имеет порядок аппроксимации $O\left(h^4, g^4\right)$, где $\Delta x = h, \Delta y = g$:

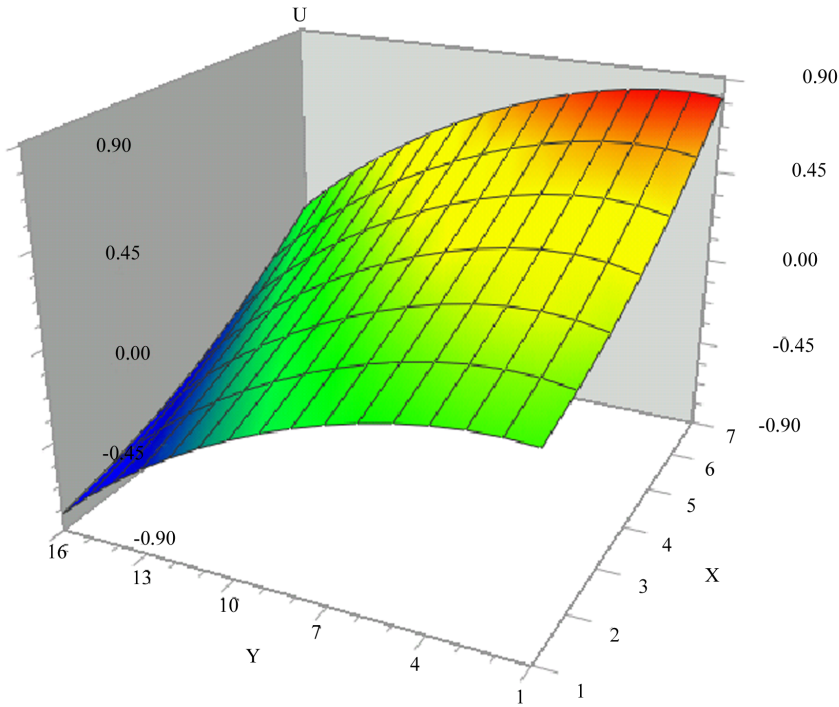


Рис. 8.4

$$u_{i+1,j+1}+u_{i-1,j+1}+u_{i+1,j-1}+u_{i-1,j-1}-2\frac{h^2-5g^2}{h^2+g^2}\left(u_{i+1,j}+u_{i-1,j}\right)+$$
$$+2\frac{5h^2-g^2}{h^2+g^2}\left(u_{i,j+1}+u_{i,j-1}\right)-20u_{i,j}=0 \qquad (8.9)$$

Шаблон девятиточечной схемы показан на *рис. 8.5* слева. На равномерной сетке порядок аппроксимации увеличивается вдвое, поэтому девятиточечная схема кажется довольно привлекательной для решения уравнений Лапласа, однако для уравнений более общего вида погрешность аппроксимации даже ниже.

Исходя из инвариантности оператора Лапласа, можно построить другие конечно-разностные схемы, например, вместо четырех точек (исключая

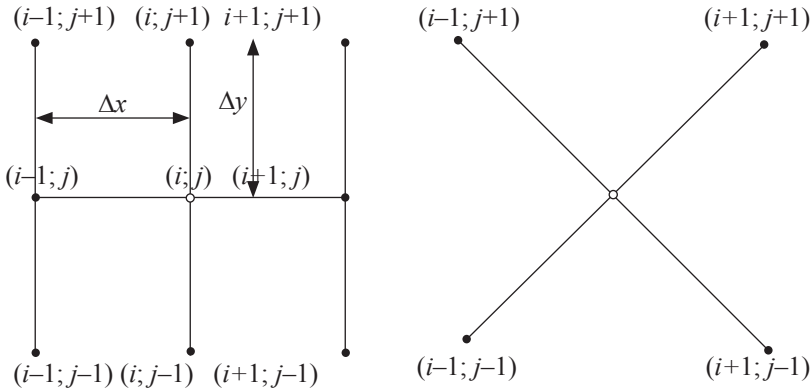


Рис. 8.5

центральную) пятиточечной схемы Рунге использовать точки, в которые они переходят при повороте на 45 градусов относительно центральной точки (узел (i, j). При этом шаг увеличивается до $\sqrt{2}h$, и в результате получается схема, шаблон которой показан на *рис. 8.5* справа:

$$u_{i+1,j+1} + u_{i+1,j-1} + u_{i-1,j-1} + u_{i-1,j+1} - 4u_{i,j} = 0. \qquad (8.10)$$

Схема (8.10) получила название диагональной пятиточечной схемы.

ФАКТОРИЗАЦИЯ ЛЕНТОЧНЫХ МАТРИЦ

Компактная форма записи матрицы

Пусть требуется получить решение системы уравнений с ленточной матрицей. В качестве элементов матрицы можно взять коэффициенты конечно — разностного аналога дифференциального уравнения 4-го порядка. Например, рассмотрим матрицу A[npxnp]:

$$\begin{vmatrix} x & g & h & 0 & 0 & 0 \\ f & x & g & h & 0 & 0 \\ e & f & x & g & h & 0 \\ 0 & e & f & x & g & h \\ 0 & 0 & e & f & x & g \\ 0 & 0 & 0 & e & f & x \end{vmatrix}$$

m1 — число наддиагоналей $(g, h) = 2$
m2 — число поддиагоналей $(f, e) = 2$
np — размерность матрицы A $= 6$
X — элементы главной диагонали
Исходная матрица A записывается в компактной форме хранения (без нулей) следующим образом:

$$\begin{matrix} 0 & 0 \\ & 0 \\ & \\ & \\ & \\ & \end{matrix} \begin{vmatrix} x & g & h & 0 & 0 & 0 \\ f & x & g & h & 0 & 0 \\ e & f & x & g & h & 0 \\ 0 & e & f & x & g & h \\ 0 & 0 & e & f & x & g \\ 0 & 0 & 0 & e & f & x \end{vmatrix} \begin{matrix} \\ \\ \\ \\ 0 \\ 0 & 0 \end{matrix}$$

К диагональным элементам дописываются нули, так чтобы из исходной квадратной матрицы A получилась прямоугольная матрица размерностью npXmp, где $\mathrm{mp} = \mathrm{m1} + \mathrm{m2} + 1$. Таким образом вместо исходной матрицы A записывается следующая матрица:

МАТРИЦА [A] В КОМПАКТНОЙ ФОРМЕ ХРАНЕНИЯ

$$\begin{matrix} 0 & 0 & x & g & h \\ 0 & f & x & g & h \\ e & f & x & g & h \\ e & f & x & g & h \\ e & f & x & g & 0 \\ e & f & x & 0 & 0 \end{matrix}$$

[6×5]

или с учетом роста np:

$$\begin{matrix} 0 & 0 & x & g & h \\ 0 & f & x & g & h \\ e & f & x & g & h \\ . & . & . & . & . \\ . & . & . & . & . \\ e & f & x & g & h \\ e & f & x & g & 0 \\ e & f & x & 0 & 0 \end{matrix}$$

С помощью пакета Numerical Recipes можно получить решение этой пятидиагональной системы, например с коэффициентами $X = 6$, $e = h = 1,0$ $f = g = -4.0$. Сначала нужно сконструировать вектор правой части и вектор решения при помощи модуля BANMUL (перемножение ленточных матриц). Например, можно задать вектор решения так: $X(i) = float(i)$, $i = 1,..,$ np; и по нему получить вектор правой части B, вызвав подпрограмму BANMUL. При помощи модуля BANDEC получить LU-разложение ленточной матрицы, а при помощи BANBKS осуществить обратную постановку.

ПРИМЕР

```
 PROGRAM BANDMTEST
 INTEGER(4)            i, j, np
 INTEGER(4),PARAMETER  :: m1=2,m2=2,mp=m1+m2+1
 REAL(4)   ,ALLOCATABLE:: A(:,:),B(:),AL(:,:),X(:)
 INTEGER(4),ALLOCATABLE:: INDX(:)
 REAL(4)               :: BAND(mp)=(/1.0,-4.0,6.0,-4.0,1.0/)
      REAL(4)              D
      CHARACTER            fmt*20, cvn*5, cvs*5, fmt2*20
      write(*,*)"Please, give n:"
      read(*,*)np
      ALLOCATE (A(np,mp), B(np),INDX(np), ↳
AL(np,m1), X(np))
 cvs='4.1'
 write(cvn,'(i3)')np
 fmt='('//cvn//'F'//cvs//')'
 cvs='7.3'
 fmt2='('//cvn//'F'//cvs//')'
      CALL ARRAYGEN(BAND,A,mp,np)
      write(*,*)"Matrix A:"
 DO i=1, np
       WRITE(*,fmt)(A(i,j),j=1, mp)
 ENDDO
      WRITE(*,*)
```

```
DO i=1, np
       X(i)=float(i)
ENDDO
CALL BANMUL(A,np,m1,m2,np,mp,X,B)
      write(*,*)"Vector B:"
      WRITE(*,fmt2)(B(i),i=1, np)
      write(*,*)"Vector X=[A]{B}:"
      WRITE(*,fmt2)(X(i),i=1, np)
      WRITE(*,*)
      WRITE(*,*)"Calling Numerical Recipes subroutines..."
      WRITE(*,*)
CALL BANDEC(A,np,m1,m2,np,mp,AL,m1,INDX,D)
      CALL BANBKS(A,np,m1,m2,np,mp,AL,m1,INDX,B)
      WRITE(*,*)"Solution Vector is  : (check above!)"
      WRITE(*,fmt2)(B(i),i=1, np)
WRITE(*,*)
      DEALLOCATE(A,B,INDX,AL,X)
END PROGRAM BANDTEST
SUBROUTINE ARRAYGEN(BAND,A,mp,np)
      INTEGER(4) np,mp,i,j
      REAL(4)    A(np,mp),B(np),BAND(mp)
DO i=1, np
       DO j=1, mp
         IF((i==1.AND.j<=2))THEN
                 A(i,j)=0.0
         ELSEIF(i==1.AND.j>2)THEN
                 A(i,j)=BAND(j)
   ELSEIF(i==2.AND.j==1)THEN
           A(i,j)=0.0
   ELSEIF(i==2.AND.J>1)THEN
           A(i,j)=BAND(j)
   ELSEIF(i>2.AND.i<np-1)THEN
```

```
            A(i,j)=BAND(j)
        ELSEIF(i==np-1.AND.j<mp)THEN
            A(i,j)=BAND(j)
        ELSEIF(i==np-1.AND.j==mp)THEN
          A(i,j)=0.0
        ELSEIF(i==np.AND.j<=3)THEN
            A(i,j)=BAND(j)
        ELSEIF(i==np.AND.j>=3)THEN
            A(i,j)=0.0
        ENDIF
      ENDDO
    ENDDO
END SUBROUTINE
```

РАЗРЕЖЕННЫЕ МАТРИЦЫ

Упаковка разреженных матриц

Пусть, требуется компактно сохранить разреженную матрицу, например

$$[A] = \begin{matrix} 3 & 0 & 1 & 0 & 0 \\ 0 & 4 & 0 & 0 & 0 \\ 0 & 7 & 5 & 9 & 0 \\ 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 6 & 5 \end{matrix}$$

N×N, N = 5, содержащую 9 ненулевых элементов. Для сохранения такой матрицы потребуется два одномерных массива — один целый *IJA* и один вещественный *SA*, все — размерностью 11 элементов.

K	1	2	3	4	5	6	7	8	9	10	11
IJA(k)	7	8	8	10	11	12	3	2	4	5	4
SA(k)	3.	4.	5.	0.	5.		1.	7.	9.	2.	6.

Таким образом, диагональные элементы матрицы [*A*] в строке *i* это *SA*(*i*), и недиагональные элементы, располагающиеся в соответствующей строке, строке матрицы [*A*] есть элементы массива *SA*(*k*), где *k* проходит цикл от *IJA*(*i*) до *IJA*(*i*+1)–1, если верхняя граница больше или равна единице.

ПРИМЕР 1

Использование библиотеки Numerical Recipes для упаковки разреженной матрицы (модуль sprsin.for).

```
PROGRAM xsprsin
INTEGER(4), PARAMETER :: NP=5,NMAX=2*NP*NP+1
INTEGER(4)  i,j,msize,ija(NMAX)
REAL a(NP,NP),aa(NP,NP),sa(NMAX)
REAL(4) :: a(NP,NP)= (/3.0,0.0,0.0,0.0,0.0,
*                      0.0,4.0,7.0,0.0,0.0,
*                      1.0,0.0,5.0,0.0,0.0,
```

```
*                0.0,0.0,9.0,0.0,6.0,
*                0.0,0.0,0.0,2.0,5.0/)
 call sprsin(a,NP,NP,0.5,NMAX,sa,ija)
 msize=ija(ija(1)-1)-1
 sa(NP+1)=0.0
 write(*,'(t4,a,t18,a,t24,a)') 'index','ija','sa'
 do i=1,msize
   write(*,'(t2,i4,t16,i4,t20,f12.6)') i,ija(i),sa(i)
 enddo
 aa=0.0
 do i=1,NP
   aa(i,i)=sa(i)
   do j=ija(i),ija(i+1)-1
     aa(i,ija(j))=sa(j)
   enddo
 enddo
 write(*,*) 'Original matrix:'
 write(*,'(5f7.1)') ((a(i,j),j=1,NP),i=1,NP)
 write(*,*) 'Reconstructed matrix:'
 write(*,'(5f7.1)') ((aa(i,j),j=1,NP),i=1,NP)
 END PROGRAM
```

ПРИМЕР 2

Решение системы уравнений $[A]\{x\} = \{b\}$, где $[A]$ — разреженная матрица, вектор $\{b\} = 0$. Матрица A должна быть упакована по формату процедуры **sprsin**. В примере 2 используется матрица, размерностью 20×20.

Для решения систем с разреженной матрицей, вызывается модуль **linbcg**, реализующий итерационную процедуру решения систем линейных алгебраических уравнений методом сопряженных градиентов. Условие окончания итераций $(\text{ITOL} = 1)\left|[A]\{x\} - \{b\}\right| / \left|\{b\}\right| < \text{TOL}$, где TOL — точность задаваемая пользователем. Максимальное число итераций задается параметром *IT*MAX, максимальная размерность матрицы задается параметром *N*MAX, текущая размерность матрицы задается параметром *NP*. Вектор решения возвращается в массив *X*.

Полученное решение можно проверить, умножив разреженную матрицу на вектор решения, путем вызовам процедуры **dsprsax.** Все подпрограммы в примере рассчитаны на работу с данными двойной точности.

```
PROGRAM xlinbcg
INTEGER(4),PARAMETER :: NP=20,NMAX=1000,ITOL=1,ITMAX=75
REAL(8),   PARAMETER :: TOL=1.d-9
INTEGER(4)             i,iter
REAL(8),DIMENSION(NP):: b,x,bcmp
REAL(8)                err
REAL(8)             :: sa(NMAX)=
   (/3.d0,3.d0,3.d0,3.d0,3.d0,3.d0,3.d0,3.d0,
*   3.d0,3.d0,3.d0,3.d0,3.d0,3.d0,3.d0,3.d0,
*   3.d0,3.d0,3.d0,3.d0,0.d0,2.d0,-2.d0,2.d0,
*  -2.d0,2.d0,-2.d0,2.d0,-2.d0,2.d0,-2.d0,
*   2.d0,-2.d0,2.d0,-2.d0,2.d0,-2.d0,2.d0,-2.d0,
*   2.d0,-2.d0,2.d0,-2.d0,2.d0,-2.d0,2.d0,
*   -2.d0,2.d0,-2.d0,2.d0,-2.d0,2.d0,-2.d0,
*   2.d0,-2.d0,2.d0,-2.d0,2.d0,-2.d0,941*0.d0/)
INTEGER(4)          :: ija(NMAX)=
   (/22,23,25,27,29,31,33,35,37,39,41,43,45,47,49,51,53,
*   55,57,59,60,2,1,3,2,4,3,5,4,6,5,7,6,8,7,9,8,10,9,11,10,
*   12,11,13,12,14,13,15,14,16,15,17,16,18,17,19,18,20,19,941*0/)
x=0.d0
b=1.d0
b(1)=3.d0
b(NP)=-1.d0
call linbcg(NP,b,x,ITOL,TOL,ITMAX,iter,err)
write(*,'(/1x,a,e15.6)') 'Estimated error:',err
write(*,'(/1x,a,i6)') 'Iterations needed:',iter
write(*,'(/1x,a)') 'Solution vector:'
write(*,'(1x,5f12.6)') (x(i),i=1,NP)
call dsprsax(sa,ija,x,bcmp,NP)
write(*,'(/1x,a)') 'press RETURN to continue...'
read(*,*)
write(*,'(1x,a/t8,a,t22,a)')'Test vector:','a*x','b'
do i=1,NP
  write(*,'(1x,2f12.6)') bcmp(i),b(i)
enddo
END
```

ФАКТОРИЗАЦИЯ МАТРИЦ ОБЩЕГО ВИДА

Решение систем алгебраических уравнений методом Краута

ПРИМЕР 1

Решается система уравнений $[A]\{x\} = \{b\}$ методом Краута. Различные варианты матриц и векторов правой части уравнений, заданы в исходном дисковом файле matrx1.dat. Размерность матриц *A* задается параметром NP. Вектор правой части задается в двумерном массиве *b*. После *LU*-разложения, верхняя и нижняя треугольные матрицы записываются на место исходной матрицы *A*, а вектор решения записывается на место исходного вектора правой части процедурой **ludcmp**.

В головной программе должны быть также заданы рабочий массив целого типа INDX и вспомогательный параметр вещественного типа *p*. Получаемые результаты могут быть проверены, путем перемножения исходной матрицы на вектор решения, что реализовано в следующей головной программе xlubksb.for:

```
    PROGRAM xlubksb
    INTEGER(4),PARAMETER    :: NP=20
    REAL(4),DIMENSION(NP,NP):: a,b,c
    REAL(4)                 p,x(NP)
    INTEGER(4)                j,k,l,m,n,indx(NP)
    CHARACTER(3)              txt
    open(7,file='MATRX1.DAT',status='old')
    read(7,*)
10  read(7,*)
    read(7,*) n,m
    read(7,*)
    read(7,*) ((a(k,l), l=1,n), k=1,n)
    read(7,*)
    read(7,*) ((b(k,l), k=1,n), l=1,m)
    do l=1,n
     do k=1,n
      c(k,l)=a(k,l)
     enddo
    enddo
```

```
      call ludcmp(c,n,NP,indx,p)
      do k=1,m
        do l=1,n
          x(l)=b(l,k)
        enddo
        call lubksb(c,n,NP,indx,x)
        write(*,*) 'Right-hand side vector:'
        write(*,'(1x,6f12.6)') (b(l,k), l=1,n)
        write(*,*) 'Result of matrix applied to sol''n vector'
        do l=1,n
          b(l,k)=0.0
          do j=1,n
            b(l,k)=b(l,k)+a(l,j)*x(j)
          enddo
        enddo
        write(*,'(1x,6f12.6)') (b(l,k), l=1,n)
        write(*,*) '**********************************'
      enddo
      write(*,*) 'Press RETURN for next problem:'
      read(*,*)
      read(7,'(a3)') txt
      if (txt.ne.'END') goto 10
      close(7)
      END PROGRAM
```

Содержимое файла matrx1.dat :

```
MATRICES FOR INPUT TO TEST ROUTINES
Size of matrix (NxN), Number of solutions:
5,2
Matrix A:
1.0 2.0 3.0 4.0 5.0
2.0 3.0 4.0 5.0 1.0
3.0 4.0 5.0 1.0 2.0
4.0 5.0 1.0 2.0 3.0
5.0 1.0 2.0 3.0 4.0
Solution vectors:
1.0 1.0 1.0 1.0 1.0
1.0 2.0 3.0 4.0 5.0
NEXT PROBLEM:
```

Size of matrix (N×N), Number of solutions:
5,2
Matrix A:
1.4 2.1 2.1 7.4 9.6
1.6 1.5 1.1 0.7 5.0
3.8 8.0 9.6 5.4 8.8
4.6 8.2 8.4 0.4 8.0
2.6 2.9 0.1 9.6 7.7
Solution vectors:
1.1 1.6 4.7 9.1 0.1
4.0 9.3 8.4 0.4 4.1
END

Список литературы

1. *Д. Андерсон, Р. Плетчер*. Вычислительная гидромеханика и теплообмен. – М.: МИР, 1990.
2. *К. Флетчер*. Вычислительные методы в динамике жидкостей. – М.: МИР, 1991.
3. *W. Press, W. Vetterling*. Numerical Recipes. – Cambridge University Press, 2004.
4. *Д. Каханер, К. Моулер*. Численные методы и программное обеспечение. – М.: МИР, 1998.
5. *Д. Голуб, Ч. ван Лоун*. Матричные вычисления. – М.: МИР, 1998.
6. *Формалев В.Ф., Ревизников Д.Л*. Численные методы. – М.: Физматлит, 2004.
7. *А.А. Амосов, Ю.А. Дубинский, Н.В. Копченова*. Вычислительные методы для инженеров. – М.: Высшая школа, 1994.
8. *Д. Райс*. Матричные вычисления и математическое обеспечение. – М.: МИР, 1984.
9. *Н.С. Бахвалов, Н.В. Жидков, Г.Л. Кобельков*. Численные методы. – М.: изд-во МГУ, 2011.
10. *W. Press, W. Vetterling*. Numerical Recipes in Fortran 90. – Cambridge University Press, 1996.
11. *W. Press, W. Vetterling*. Numerical Recipes Example Book. – Cambridge University Press, 2004.
12. *Д. Деммель*. Вычислительная линейная алгебра. – М.: МИР, 2001.

Содержание

Учебное издание

Малик Сабитович Максютов

Численные методы решения краевых задач

Учебное пособие

Технический редактор
Компьютерная верстка - К.А. Антонов
Дизайн обложки - К.А. Антонов

Подписано в печать 15.10.2014. Формат 60x84 $^{1}/_{16}$.
Бумага офисная. Гарнитура *Times New Roman*. Печ. лист 5,5.
Тираж 25 экз. Заказ № 48.

Московский государственный гуманитарно-экономический университет
107150, Москва, ул. Лосиноостровская, д. 49.
Отпечатано в типографии МГГЭУ по технологии CtP.

ДЛЯ ЗАМЕТОК

ДЛЯ ЗАМЕТОК