

Всеволод Заостровский, 409 группа

Отчёт по задаче "Решение уравнения типа теплопроводности с коэффициентами в дивергенции".

Содержание

1	Постановка задачи.	2
1.1	Одномерный Лаплас	2
1.2	Двумерный Лаплас	2
2	Алгоритм решения одномерной схемы.	2
2.1	Дискретизация	2
2.2	Общий вид матрицы уравнения.	3
2.3	Решение схемы.	3
3	Алгоритм решения двумерной схемы.	4
3.1	Дискретизация.	4
3.2	Общий вид матрицы уравнения.	4
3.3	Решение схемы.	6
3.4	Алгоритм для программирования.	8
4	Тесты.	9
4.1	Простейший случай ($k = \text{const}$)	9
4.2	Непостоянный k	9

1 Постановка задачи.

1.1 Одномерный Лаплас

Необходимо решить уравнение:

$$u_t(t, x) = \operatorname{div}(k(x) \operatorname{grad} u(t, x)).$$

Будем считать, что $0 \leq t, x \leq 1$. В моём варианте, краевые условия:

$$\begin{aligned} u(t, x)|_{x \in \partial\Omega} &= 0, \quad \Omega = [0, 1]. \\ u(0, x) &= u^0(x), \quad x \in \Omega. \end{aligned}$$

1.2 Двумерный Лаплас

Необходимо решить уравнение:

$$u_t(t, x, y) = \operatorname{div}(k(x, y) \operatorname{grad} u(t, x, y)).$$

Будем считать, что $0 \leq t, x, y \leq 1$. В моём варианте, краевые условия:

$$\begin{aligned} u(t, x, y)|_{(x,y) \in \partial\Omega} &= 0, \quad \Omega = [0, 1] \times [0, 1]. \\ u(0, x, y) &= u^0(x, y), \quad (x, y) \in \Omega. \end{aligned}$$

2 Алгоритм решения одномерной схемы.

2.1 Дискретизация

Уравнение будем приближать посредством следующей схемы:

$$\frac{u_i^{n+1} - u_i^n}{\tau} = \frac{k(x_{i+\frac{1}{2}}) \frac{u_{i+1}^{n+1} - u_i^{n+1}}{h} - k(x_{i-\frac{1}{2}}) \frac{u_i^{n+1} - u_{i-1}^{n+1}}{h}}{h}.$$

Краевые условия:

$$u(t, x)|_{x \in \partial\Omega} = 0, \quad \Omega = [0, 1].$$

$$u(0, x) = u^0(x), \quad x \in \Omega.$$

2.2 Общий вид матрицы уравнения.

Преобразуем схему:

$$\begin{aligned} u_i^{n+1} - u_i^n &= \tau k(x_{i+\frac{1}{2}}) \frac{u_{i+1}^{n+1} - u_i^{n+1}}{h^2} - \tau k(x_{i-\frac{1}{2}}) \frac{u_i^{n+1} - u_{i-1}^{n+1}}{h^2}. \\ u_i^{n+1} - u_i^n &= \tau k(x_{i+\frac{1}{2}}) \frac{u_{i+1}^{n+1}}{h^2} - u_i^{n+1} \frac{\tau}{h^2} (k(x_{i+\frac{1}{2}}) + k(x_{i-\frac{1}{2}})) + \tau k(x_{i-\frac{1}{2}}) \frac{u_{i-1}^{n+1}}{h^2}. \end{aligned}$$

Таким образом, получим:

$$-u_{i+1}^{n+1} k(x_{i+\frac{1}{2}}) \frac{\tau}{h^2} + u_i^{n+1} \left(1 + \frac{\tau}{h^2} (k(x_{i+\frac{1}{2}}) + k(x_{i-\frac{1}{2}})) \right) - u_{i-1}^{n+1} k(x_{i-\frac{1}{2}}) \frac{\tau}{h^2} = u_i^n. \quad (1)$$

Матрица примет вид:

$$A = \begin{pmatrix} c & b_+ & 0 & 0 & 0 & \dots & 0 \\ b_- & c & b_+ & 0 & 0 & \dots & 0 \\ 0 & b_- & c & b_+ & 0 & \dots & 0 \\ 0 & 0 & b_- & c & b_+ & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & b_- & c & b_+ \\ 0 & 0 & 0 & \dots & 0 & b_- & c \end{pmatrix}, \text{ где } \begin{cases} c = 1 + \frac{\tau}{h^2} (k(x_{i+\frac{1}{2}}) + k(x_{i-\frac{1}{2}})), \\ b_+ = -k(x_{i+\frac{1}{2}}) \frac{\tau}{h^2}, \\ b_- = -k(x_{i-\frac{1}{2}}) \frac{\tau}{h^2}. \end{cases}$$

2.3 Решение схемы.

Итоговый вид интересующей нас системы:

$$Au^{n+1} = u^n, \quad u^n := (u_0^n, u_1^n, \dots, u_{N_X}^n).$$

Как видно, эту системы легко решить методом прогонки: двигаясь от 0-го слоя к N -му. Несколько сложнее дела обстоят с двумерной схемой, которая описана ниже.

3 Алгоритм решения двумерной схемы.

3.1 Дискретизация.

Уравнение будем приближать посредством следующей схемы:

$$\begin{aligned} \frac{u_i^{n+1} - u_i^n}{\tau} = & \frac{k(x_{i+\frac{1}{2},j}) \frac{u_{i+1,j}^{n+1} - u_{i,j}^{n+1}}{h_X} - k(x_{i-\frac{1}{2},j}) \frac{u_{i,j}^{n+1} - u_{i-1,j}^{n+1}}{h_X}}{h_X} + \\ & + \frac{k(x_{i,j+\frac{1}{2}}) \frac{u_{i,j+1}^{n+1} - u_{i,j}^{n+1}}{h_Y} - k(x_{i,j-\frac{1}{2}}) \frac{u_{i,j}^{n+1} - u_{i,j-1}^{n+1}}{h_Y}}{h_Y}. \end{aligned}$$

Краевые условия:

$$\begin{aligned} u(t, x, y) \big|_{(x,y) \in \partial\Omega} &= 0, \quad \Omega = [0, 1] \times [0, 1]. \\ u(0, x, y) &= u^0(x, y), \quad x \in \Omega. \end{aligned}$$

3.2 Общий вид матрицы уравнения.

Для придания выкладкам хоть сколько-нибудь приемлемого вида, здесь и далее считаем $h_X = h_Y = h = \frac{1}{N_X-1}$, $N_X = N_Y$. Пользуясь вычислениями из предыдущего раздела, получим:

$$\begin{aligned} u_{i,j}^{n+1} - u_{i,j}^n &= \tau k(x_{i+\frac{1}{2}}, y_j) \frac{u_{i+1,j}^{n+1} - u_{i,j}^{n+1}}{h^2} - \tau k(x_{i-\frac{1}{2}}, y_j) \frac{u_{i,j}^{n+1} - u_{i-1,j}^{n+1}}{h^2} + \\ &+ \tau k(x_{i,j+\frac{1}{2}}) \frac{u_{i,j+1}^{n+1} - u_{i,j}^{n+1}}{h^2} - \tau k(x_{i,j-\frac{1}{2}}) \frac{u_{i,j}^{n+1} - u_{i,j-1}^{n+1}}{h^2}. \\ u_{i,j}^{n+1} - u_{i,j}^n &= -u_{i,j}^{n+1} \frac{\tau}{h^2} \left(k(x_{i+\frac{1}{2}}, y_j) + k(x_{i-\frac{1}{2}}, y_j) + k(x_{i,j+\frac{1}{2}}) + k(x_{i,j-\frac{1}{2}}) \right) + \\ &+ \tau k(x_{i+\frac{1}{2}}, y_j) \frac{u_{i+1,j}^{n+1}}{h^2} - \tau k(x_{i-\frac{1}{2}}, y_j) \frac{u_{i-1,j}^{n+1}}{h^2} + \tau k(x_{i,j+\frac{1}{2}}) \frac{u_{i,j+1}^{n+1}}{h^2} - \tau k(x_{i,j-\frac{1}{2}}) \frac{u_{i,j-1}^{n+1}}{h^2}. \end{aligned}$$

Итоговая схема выглядит следующим образом:

$$\begin{aligned}
& u_{i,j}^{n+1} \left(\frac{1}{\tau} + \frac{1}{h^2} \left(k(x_{i+\frac{1}{2}}, y_j) + k(x_{i-\frac{1}{2}}, y_j) + k(x_i, y_{j+\frac{1}{2}}) + k(x_i, y_{j-\frac{1}{2}}) \right) \right) - \\
& + \frac{1}{h^2} \left[-u_{i+1,j}^{n+1} k(x_{i+\frac{1}{2}}, y_j) - u_{i-1,j}^{n+1} k(x_{i-\frac{1}{2}}, y_j) - u_{i,j+1}^{n+1} k(x_i, y_{j+\frac{1}{2}}) - u_{i,j-1}^{n+1} k(x_i, y_{j-\frac{1}{2}}) \right] \\
& = \frac{u_{i,j}^n}{\tau}.
\end{aligned}$$

Эту схему можно записать в огромную ($\mathbb{R}^{N_x^4}$) разреженную блочную матрицу вида:

$$A = \begin{pmatrix} I & 0 & 0 & 0 & 0 & \dots & 0 \\ D_- & C & D_+ & 0 & 0 & \dots & 0 \\ 0 & D_- & C & D_+ & 0 & \dots & 0 \\ 0 & 0 & D_- & C & D_+ & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & D_- & C & D_+ \\ 0 & 0 & 0 & \dots & 0 & 0 & I \end{pmatrix},$$

описание блоков:

Блок C :

$$C = \begin{pmatrix} c & b_+ & 0 & 0 & 0 & \dots & 0 \\ b_- & c & b_+ & 0 & 0 & \dots & 0 \\ 0 & b_- & c & b_+ & 0 & \dots & 0 \\ 0 & 0 & b_- & c & b_+ & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & b_- & c & b_+ \\ 0 & 0 & 0 & \dots & 0 & b_- & c \end{pmatrix},$$

в матрице C :

$$\begin{cases} c = \frac{1}{\tau} + \frac{1}{h^2} \left(k(x_{i+\frac{1}{2}}, y_j) + k(x_{i-\frac{1}{2}}, y_j) + k(x_i, y_{j+\frac{1}{2}}) + k(x_i, y_{j-\frac{1}{2}}) \right), \\ b_+ = -k(x_{i+\frac{1}{2}}, y_j) \frac{1}{h^2}, \\ b_- = -k(x_{i-\frac{1}{2}}, y_j) \frac{1}{h^2}; \end{cases}$$

Блок I : I — единичная матрица размера $N_X \times N_X$;

Блок D_- : $D_- = -k(x_i, y_{j-\frac{1}{2}}) \frac{\tau}{h^2} I =: d_- I$;

Блок D_+ : $D_+ = -k(x_i, y_{j+\frac{1}{2}}) \frac{\tau}{h^2} I =: d_+ I$.

Итоговый вид уравнения:

$$Au^{n+1} = \frac{u^n}{\tau}.$$

По предыдущему слою мы будем находить следующий, начиная с 0-го слоя, который нам дан, по условию. Отметим, что мы хотим построить трёхмерную матрицу $U = (u_{i,j}^n)_{0 \leq i,j \leq N_X}^{0 \leq n \leq N}$, но форма записи матрицы A предполагаем, что множество $(u_{i,j}^n)_{0 \leq i,j \leq N_X}^{n=\text{const}}$ вытягивается в вектор u^n :

$$u^n = (u_{0,0}^n, u_{1,0}^n, u_{2,0}^n, \dots, u_{0,1}^n, u_{0,2}^n, \dots, u_{0,N_X}^n, u_{1,N_X}^n, \dots, u_{N_X,N_X}^n)^T$$

3.3 Решение схемы.

Для решения этой системы также можно применять прогонку (точнее, её более общую модификацию). Мы применим итеративный алгоритм решения с предобуславливателем, который подробно опишем ниже. Общий вид таких алгоритмов:

$$B \frac{u^{n+1,p+1} - u^{n+1,p}}{\theta} + Au^{n+1,p} = b^n.$$

В нашем случае,

A — матрица, описанная в разделе 3.2. Следует отметить, что эта матрица пятидиагональная, так что, несмотря на то, что формально она принадлежит пространству $\mathbb{R}^{(N_X+1)^4}$, для её хранения требуется лишь $5 * (N_X + 1)^2$ памяти (по массиву для каждой из 5 диагоналей), а умножение матрицы на вектор требует $10 * (N_X + 1)^2$ арифметических операций.

$u^{n,p}$ — результат после p -ой итерации процесса для n слоя ответа. Отметим, что мы считаем $u^{n,0} = u^n$.

θ — итерационный параметр. Наивысшая (в некотором смысле) скорость сходимости достигается при $\theta = \frac{2}{m+M}$, где M и m — соответственно, максимальное и минимальное собственные значения матрицы.

B — предобуславливатель, берётся разным для разных задач. Мы рассмотрим $B = A|_{k(x_i, y_j) = \frac{k(x_0, y_0) + k(x_{N_X}, y_{N_X})}{2}}$.

Итерироваться мы будем следующим образом:

1. Заметим, что:

$$B \frac{u^{n+1,p+1} - u^{n+1,p}}{\theta} + Au^{n+1,p} = \frac{u^n}{\tau} \Leftrightarrow \begin{cases} By^{p+1} = \frac{u^n}{\tau} - Au^{n+1,p}, \\ u^{n+1,p+1} = u^{n+1,p} + \theta y^{p+1}. \end{cases}$$

2. За $O(N^2)$ вычислим вектор $b - Ax^p$.

3. С помощью метода Фурье за $O(N^3)$ решим систему $By^{p+1} = b - Ax^p$. Схема при этом принимает вид:

$$\begin{aligned} & \frac{u_{i,j}^{n+1}}{k\tau} - \frac{u_{i+1,j}^{n+1} - 2u_{i,j}^{n+1} + u_{i-1,j}^{n+1}}{h^2} - \frac{u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}}{h^2} = \\ & = \underbrace{f(x_i, y_j)}_{=(\frac{u^n}{\tau} - Au^{n+1,p})/k|_{i,j}} = \frac{u^n}{k\tau} - \frac{1}{k}Au^{n+1,p}. \end{aligned}$$

4. Следующий вектор в итеративной процедуре вычислим по формуле $x^{k+1} = x^k + \tau y^{k+1}$.

3.4 Алгоритм для программирования.

На шаге n нам известны слои вплоть до u^n (слой — матрица, вытянутая в вектор). u^0 задан изначально.

1. Нужно решить уравнение:

$$Au^{n+1} = \frac{u^n}{\tau}.$$

Для этого:

- (а) Решаем уравнение

$$By^{k+1} = \frac{u^n}{\tau} - Ax^k,$$

где B — матрица Фурье с постоянным k .

- (b) Вычисляем $x^{k+1} = x^k + \theta y^{k+1}$.

- (c) Повторяем процедуру пока не увидим сходимость.

2. Таким образом, от слоя к слою, восстановим всю матрицу.

4 Тесты.

4.1 Простейший случай ($k = \text{const}$)

Для простоты сравнения с предыдущим отчетом было рассмотрено $k = 1$. В целом, всё аналогично предыдущему случаю, поэтому здесь не проводилось более глубоких тестов. На графике 1 по оси y отложено значение функции, по оси x — порядковый номер тройки точек (x, y, z) : вывод задачи представлен в виде

$$(x, y, z, \text{численное решение}, \text{аналитическое решение})$$

и, при заданном разбиении каждой точки равномерной сетки соответствует натуральное число.

Отметим, что сходимость итерационного алгоритма в этом случае происходила за одну итерацию, как и должно быть.

4.2 Непостоянный k

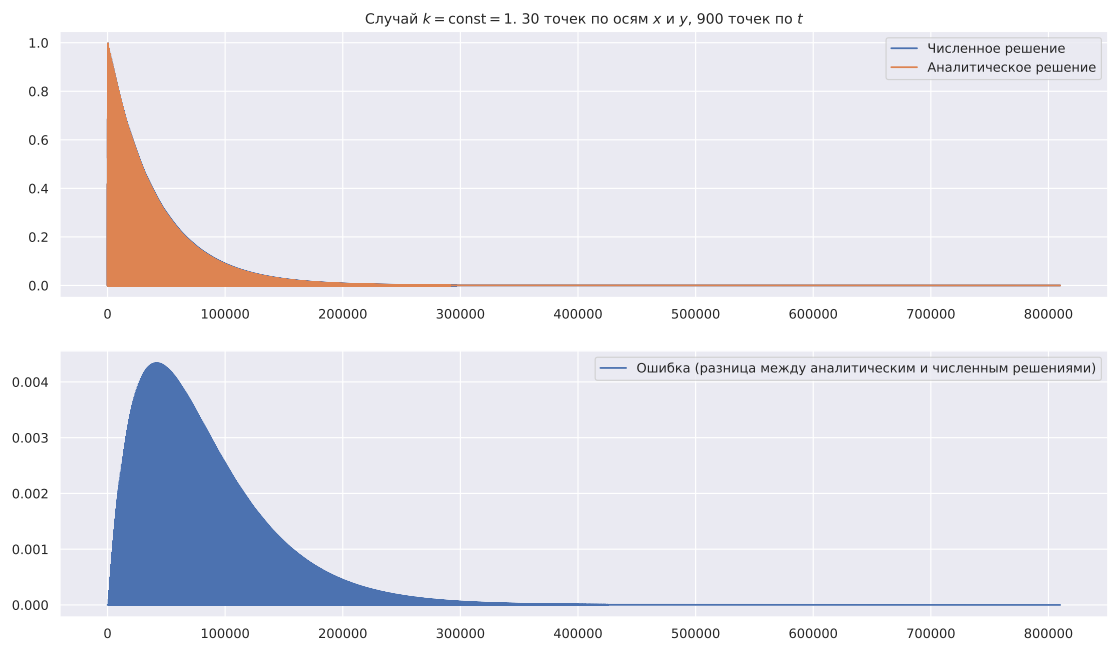


Рис. 1: Замыкание на постоянный k .