

**Московский Государственный Университет имени
М.В. Ломоносова
Факультет вычислительной математики и
кибернетики
Введение в численные методы
Отчёт по практическому заданию**

Студент Кибизов Кирилл, группа 207

2024

Оглавление

Оглавление	1
1 Постановка задачи	2
2 Описание используемых числовых методов	3
3 Анализ применимости используемых числовых методов	4
4 Реализация используемых числовых методов	6
Заключение	8
Приложения	9
Литература	10

Постановка задачи

Дано:

1. Уравнение в частных производных с граничными условиями:

$$\begin{cases} k_x \frac{\partial^2 u}{\partial x^2} + k_y \frac{\partial^2 u}{\partial y^2} = 0, & (x, y) \in [0, 1] \times [0, 1], \\ u(x, 0) = 0, & x \in [0, 1] \\ u(0, y) = 0, & y \in [0, 1] \\ u(x, 1) = \sin(\pi x), & x \in [0, 1] \\ u(1, y) = 0, & y \in [0, 1] \end{cases}$$

2. Разностная схема:

$$\begin{cases} k_x \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + k_y \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} = 0, & i = \overline{1, N-1}, j = \overline{1, N-1}, \\ u_{i,0} = 0, & i = \overline{0, N}, \\ u_{0,j} = 0, & j = \overline{0, N}, \\ u_{i,N} = 0, & i = \overline{0, N}, \\ u_{N,j} = 0, & j = \overline{0, N}. \end{cases}$$

где

$$u_{i,j} \approx u(x_i, y_j), \quad x_i = \frac{i}{N}, \quad y_j = \frac{j}{N}, \quad h = \frac{1}{N}.$$

3. Аналитическое решение данной задачи:

$$u(x, y) = \frac{\sinh(\pi y / \sqrt{k_y})}{\sinh(\pi / \sqrt{k_y})} \sin(\pi x)$$

Задача:

Требуется решить данную СЛАУ с помощью итерационного метода Якоби (где он применим) для $N = 100$, рассматривая следующие случаи:

1. $k_x = k_y = 1$,
2. $k_x = 1, k_y = 10^6$.

В случае неприменимости итерационного метода якоби предложить рабочий альтернативный метод.

Описание используемых числовых методов

Итерационные алгоритмы

При применении итерационных методов решения СЛАУ $Ax = f$ ответ получается в процессе построения последовательных приближений (итераций) $x_k = \{x_1^k, x_2^k, \dots, x_n^k\}$, сходящихся к решению исходной системы в пространстве E_n с евклидовой нормой $\|x\|$: $\lim_{k \rightarrow \infty} x_k = x$, где i - номер компоненты, а k - номер итерации.

Сходимость обеспечивает принципиальную возможность получить в процессе итераций ответ с любой наперед заданной степенью точности.

Если очередной член последовательности x_{k+1} может выражаться только через предыдущий $x_{k+1} = F(x_k)$. Такие итерационные алгоритмы называют одношаговыми. Обычно линейно одношаговые алгоритмы записывают в стандартной канонической форме: $B_{k+1} \frac{x_{k+1} - x_k}{\tau_{k+1}} + Ax_k = f$ и $\det B_{k+1} \neq 0$ и $\tau_{k+1} > 0$. В такой записи процесс характеризуется последовательностью матриц B_{k+1} и числовых параметров τ_{k+1} , которые называют итерационными параметрами. Если матрицы B_{k+1} и параметры τ_{k+1} не меняются в процессе итераций, т.е. не зависят от индекса k , то итерационный процесс называется стационарным.

Анализ применимости используемых числовых методов

Перед тем как применять итерационные методы для решения системы линейных алгебраических уравнений (СЛАУ), необходимо убедиться, что они сходятся в рассматриваемом случае. Это включает в себя проверку структуры и свойств матрицы системы, а также оценку выполнения достаточных условий сходимости итерационных методов.

Достаточные условия сходимости итерационного процесса

Самосопряжённость матрицы

В одномерном случае в направлении x вторая производная $\frac{\partial^2 u}{\partial x^2}$ аппроксимируется по формуле:

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h_x^2}.$$

Важно, что коэффициенты при $u_{i+1,j}$ и $u_{i-1,j}$ одинаковы (оба равны $\frac{1}{h_x^2}$). Аналогично производится аппроксимация второй производной по оси y . Благодаря симметрии разностной схемы все связи между узлами в матрице A получаются парными и зеркальными. Если уравнение для узла (i, j) ссылается на узел $(i+1, j)$ с некоторым коэффициентом, то и уравнение для узла $(i+1, j)$ будет иметь аналогичный коэффициент при $u_{i,j}$. Это обеспечивает симметричность матрицы. Известные значения на границах области не делают матрицу несимметричной, так как они просто выносятся в вектор правой части f . Таким образом, можно сделать вывод, что матрица A — самосопряжённая. В случае вещественной матрицы (все элементы матрицы A — вещественные) понятия самосопряжённости и симметричности совпадают.

Положительно определённая матрица

Чтобы доказать, что матрица A положительно определённая, нужно показать, что для любого ненулевого вектора v выполняется неравенство: $v^T A v > 0$. Рассмотрим выражение $v^T A v$. Это скаляр, который можно записать как:

$$v^T A v = \sum_{i,j} v_{i,j} (A v)_{i,j}.$$

Для матрицы A , полученной из разностной аппроксимации второго порядка, можно записать, что A действует на вектор v следующим образом:

$$(A v)_{i,j} = \frac{k_x}{h^2} (v_{i+1,j} - 2v_{i,j} + v_{i-1,j}) + \frac{k_y}{h^2} (v_{i,j+1} - 2v_{i,j} + v_{i,j-1}).$$

Подставим это в $v^T A v$ и раскроем сумму:

$$v^T A v = \sum_{i,j} v_{i,j} \left(\frac{k_x}{h^2} (v_{i+1,j} - 2v_{i,j} + v_{i-1,j}) + \frac{k_y}{h^2} (v_{i,j+1} - 2v_{i,j} + v_{i,j-1}) \right).$$

При раскрытии суммы оказывается, что многие члены взаимно сокращаются. Итоговый результат можно выразить через квадраты разностей значений v в соседних узлах. После упрощений получим:

$$v^T A v = \sum_{i,j} \frac{k_x}{h^2} (v_{i+1,j} - v_{i,j})^2 + \frac{k_y}{h^2} (v_{i,j+1} - v_{i,j})^2.$$

В выражении $v^T A v$ остались только суммы квадратов разностей значений v в соседних узлах. Так как $k_x > 0$, $k_y > 0$ и $h > 0$, каждый член суммы неотрицателен. Более того, если вектор v ненулевой, то хотя бы одно из слагаемых будет строго больше нуля. Поскольку $v^T A v$ является суммой строго неотрицательных слагаемых, и каждое из них положительно, если $v \neq 0$, то $v^T A v > 0$. Это доказывает, что матрица A положительно определённая.

Теорема Самарского

Пусть A — самосопряжённая положительно определённая матрица: $A = A^T$, $A > 0$, и $B = A - \frac{\tau}{2} A$ — положительно определённая матрица, τ — положительное число: $B = A - \frac{\tau}{2} A > 0$.

Можно утверждать, что для матрицы A , которая:

- симметрична ($A^T = A$),
- положительно определённая ($v^T A v > 0$ для любого $v \neq 0$),

выполняются достаточные условия сходимости итерационных методов, таких как метод Якоби, метод Зейделя и метод релаксации (SOR).

Сходимость методов

В итоге методы Якоби, Зейделя, верхней релаксации (SOR) применимы, к данной задаче, однако важно также учитывать:

- **Точность решения:** Точность определяется выбранным критерием остановки (например, достижением малого значения невязки или изменения решения между итерациями).
- **Быстрота сходимости:** Для улучшения быстроты сходимости можно:
 - уменьшить шаг h ,
 - использовать "ускоряющие" параметры, такие как ω в методе верхней релаксации,

Реализация используемых числовых методов

```
1 int solve_slae_via_jacobi(double u[N + 1][N + 1], double kx, double ky) {
2     double u_new[N + 1][N + 1] = {{0.0}};
3     for (int i = 0; i <= N; ++i) {
4         double x = i * h;
5         u_new[i][N] = u[i][N] = sin(M_PI * x);
6     }
7
8     int iter = 0;
9     double max_dif;
10    do {
11        max_dif = 0.0;
12        for (int i = 1; i < N; ++i) {
13            for (int j = 1; j < N; ++j) {
14                u_new[i][j] = (kx*(u[i+1][j] + u[i-1][j]) + ky*(u[i][j+1] + u[i][j-1])) /
15                    (2*(kx+ky));
16                double dif = fabs(u[i][j] - u_new[i][j]);
17                if (max_dif < dif) {
18                    max_dif = dif;
19                }
20            }
21        }
22        for (int i = 1; i < N; ++i) {
23            for (int j = 1; j < N; ++j) {
24                u[i][j] = u_new[i][j];
25            }
26        }
27    } while ((++iter < MAX_ITERS) && (max_dif > EPS));
28
29    if (iter == MAX_ITERS) {
30        return -1;
31    }
32
33    return iter;
34 }
35 }
```

Реализация метода Якоби

```
1 ./a.out
2 Please, input amount of tests (max 10): 1
3 Leave 3rd argument as 0 (for Jacobi) or as w (w = 1 for Gauss-Seidel; 1 < w < 2 for SOR)
4 Input kx and ky and w; for test #1: 1 1 0
5 ...
6 u(0.250000, 0.500000) | 0.139489 | 0.140904
7 u(0.250000, 0.750000) | 0.319105 | 0.320099
8 u(0.250000, 1.000000) | 0.707107 | 0.707107
9 u(0.500000, 0.000000) | 0.000000 | 0.000000
10 ...
11 u(0.500000, 0.750000) | 0.451283 | 0.452688
12 u(0.500000, 1.000000) | 1.000000 | 1.000000
13 u(0.750000, 0.000000) | 0.000000 | 0.000000
14 u(0.750000, 0.250000) | 0.052184 | 0.053187
15 ...
16 -----
17 Test #1:
18 Iterations = 10247
```

Вывод (для некоторых точек, по координатам кратных 0.25)

```

1  int solve_slac_via_w(double u[N + 1][N + 1], double kx, double ky, double w) {
2      for (int i = 0; i <= N; ++i) {
3          double x = i * h;
4          u[i][N] = sin(M_PI * x);
5      }
6
7      int iter = 0;
8      double max_dif;
9      do {
10         max_dif = 0.0;
11         for (int i = 1; i < N; ++i) {
12             for (int j = 1; j < N; ++j) {
13                 double old_val = u[i][j];
14                 double tmp = (kx*(u[i+1][j] + u[i-1][j]) + ky*(u[i][j+1] + u[i][j-1])) /
15                     (2*(kx+ky));
16                 u[i][j] = (1 - w)*old_val + w*tmp;
17
18                 double dif = fabs(u[i][j] - old_val);
19                 if (max_dif < dif) {
20                     max_dif = dif;
21                 }
22             }
23         }
24         while ((++iter < MAX_ITERS) && (max_dif > EPS));
25
26         if (iter == MAX_ITERS) {
27             return -1;
28         }
29
30         return iter;
31     }

```

Реализация метода верхней релаксации

```

1  guest@host:/media/sf_Shared/jacobi$ ./a.out
2  Please, input amount of tests (max 10): 1
3  Leave 3rd argument as 0 (for Jacobi) or as w (w = 1 for Gauss-Seidel; 1 < w < 2 for SOR)
4  Input kx and ky and w; for test #1: 1 1000000 1.0
5  ...
6  u(0.250000, 0.500000) | 0.140196 | 0.140904
7  u(0.250000, 0.750000) | 0.319612 | 0.320099
8  u(0.250000, 1.000000) | 0.707107 | 0.707107
9  u(0.500000, 0.000000) | 0.000000 | 0.000000
10 ...
11 u(0.500000, 1.000000) | 1.000000 | 1.000000
12 u(0.750000, 0.000000) | 0.000000 | 0.000000
13 u(0.750000, 0.250000) | 0.052690 | 0.053187
14 u(0.750000, 0.500000) | 0.140214 | 0.140904
15 ...
16 -----
17 Test #1:
18 Iterations = 5851

```

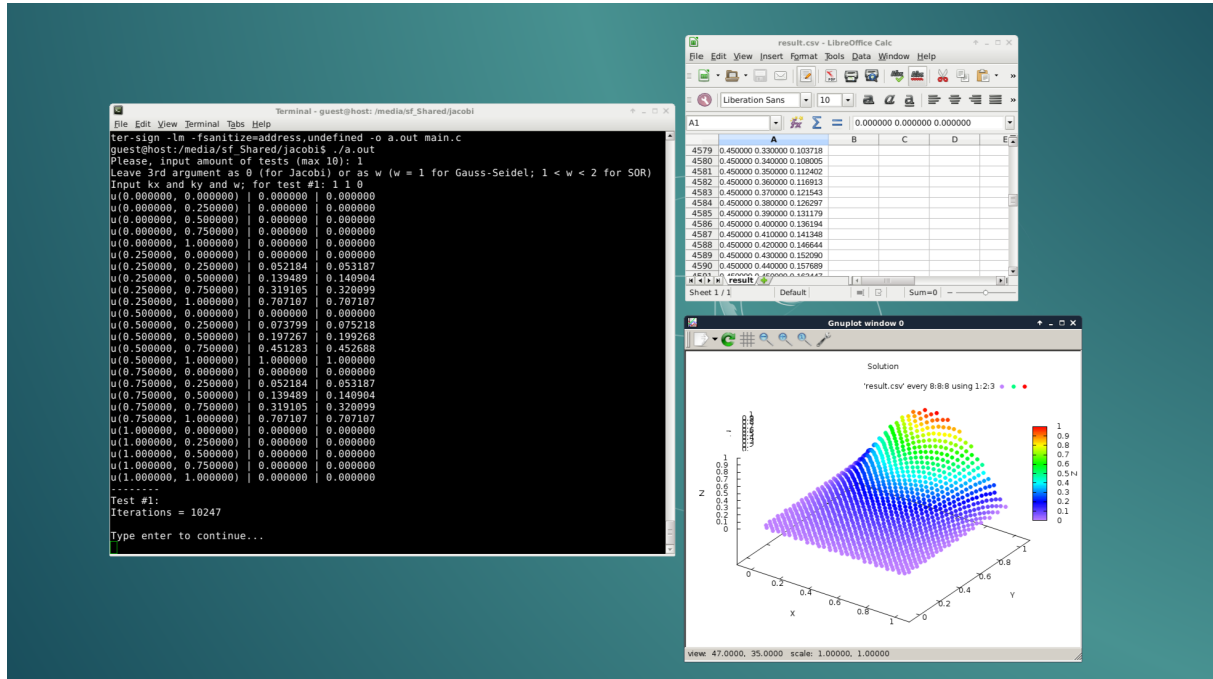
Вывод (для некоторых точек, по координатам кратных 0.25)

Заключение

В данном отчёте была рассмотрена задача решения уравнения в частных производных с помощью итерационных методов, таких как метод Якоби и метод верхней релаксации. Реализация методов была выполнена на языке программирования С. Были получены численные результаты, которые были сравнены с аналитическим решением, что продемонстрировало высокую точность использованных численных методов.

Приложения

<https://github.com/kibizoffs/jacobi>



Литература

- [1] Костомаров Д. П., Фаворский А. П. *Вводные лекции по численным методам*. — М.: Логос, 2004. — 184 с.
- [2] Самарский А. А. *Введение в численные методы*. — М.: Наука, 1989. — 416 с.