



Московский государственный университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Отчёт СКМИТ

Ковч Н. С.

609 гр.

Москва, 2025

Contents

Математическая постановка задачи	3
Метод фиктивных областей	3
Разностная схема решения задачи	4
Метод решения СЛАУ	5
Последовательный код и результаты	6
OpenMP версия	8
MPI версия	9

Математическая постановка задачи

В области D , ограниченной треугольником γ с вершинами в точках $(-3, 0), (3, 0), (0, 2)$, рассматривается дифференциальное уравнение Пуассона:

$$-\Delta u = 1, \quad (x, y) \in \text{int } D. \quad (1)$$

Решение уравнения дополняется граничным условием Дирихле:

$$u(x, y) = 0, \quad (x, y) \in \gamma. \quad (2)$$

Требуется найти функцию $u(x, y)$, удовлетворяющую уравнению (1) в области D и краевому условию (2) на её границе.

Метод фиктивных областей

Для приближенного решения задачи (1), (2) был использован метод фиктивных областей. Область D принадлежит прямоугольнику

$$\Pi = \{(x, y) : -3 < x < 3, 0 < y < 2\}$$

Область $\hat{D} = \Pi \setminus \overline{D}$ называется фиктивной областью. Выберем и зафиксируем малое $\epsilon > 0$. В прямоугольнике Π рассматривается задача Дирихле:

$$-\frac{\partial}{\partial x}(k(x, y)\frac{\partial v}{\partial x}) - \frac{\partial}{\partial y}(k(x, y)\frac{\partial v}{\partial y}) = F(x, y), \quad (x, y) \in \Pi \setminus \gamma \quad (3)$$

$$v(x, y) = 0, \quad (x, y) \in \Gamma. \quad (4)$$

с кусочно-постоянным коэффициентом

$$k(x, y) = \begin{cases} 1, & (x, y) \in D, \\ \frac{1}{\epsilon}, & (x, y) \in \hat{D} \end{cases} \quad (5)$$

и правой частью

$$F(x, y) = \begin{cases} 1, & (x, y) \in D, \\ 0, & (x, y) \in \hat{D} \end{cases} \quad (6)$$

Здесь Γ – граница прямоугольника Π . Требуется найти непрерывную в $\bar{\Pi}$ функцию $v(x, y)$, удовлетворяющую дифференциальному уравнению задачи (3) всюду в $\Pi \setminus \gamma$, равную нулю на границе Γ прямоугольника, и такую, чтобы вектор потока

$$W(x, y) = -k(x, y)\left(\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y}\right)$$

имел непрерывную нормальную компоненту на общей части криволинейной границы области D и прямоугольника Π .

Разностная схема решения задачи

Краевая задача решается численно методом конечных разностей. В $\bar{\Pi}$ определяется равномерная прямоугольная сетка $\bar{\omega}_h = \bar{\omega}_1 \times \bar{\omega}_2$:

$$\bar{\omega}_1 = \{x_i = -3 + ih_x, i = \overline{0, M}\}, \quad \bar{\omega}_2 = \{y_j = jh_y, j = \overline{0, N}\}, \quad (7)$$

где $h_x = \frac{6}{M}$, $h_y = \frac{2}{N}$. Через ω_h обозначим множество внутренних узлов сетки $\bar{\omega}_h$, т.е. множество узлов сетки прямоугольника, не лежащих на границе Γ . Рассматривается линейное пространство H функций, заданных на сетке ω_h . Обозначим через w_{ij} значение сеточной функции $w \in H$ в узле сетки $(x_i, y_j) \in \omega_h$. В пространстве H задано скалярное произведение и евклидова норма

$$(u, v) = \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} h_x h_y u_{ij} v_{ij}, \quad \|u\|_E = \sqrt{(u, u)}. \quad (8)$$

Дифференциальное уравнение задачи (3) во всех внутренних точках сетки $\bar{\omega}_h$ аппроксимируется разностным уравнением

$$-\frac{1}{h_x} \left(a_{i+1j} \frac{w_{i+1j} - w_{ij}}{h_x} - a_{ij} \frac{w_{ij} - w_{i-1j}}{h_x} \right) - \frac{1}{h_y} \left(b_{ij+1} \frac{w_{ij+1} - w_{ij}}{h_y} - b_{ij} \frac{w_{ij} - w_{ij-1}}{h_y} \right) = F_{ij}, \\ i = \overline{1, M-1}, j = \overline{1, N-1}, \quad (9)$$

в котором коэффициенты

$$a_{ij} = \frac{1}{h_y} \int_{y_{j-1/2}}^{y_{j+1/2}} k(x_{i-1/2}, t) dt, \quad b_{ij} = \frac{1}{h_x} \int_{x_{i-1/2}}^{x_{i+1/2}} k(t, y_{j-1/2}) dt \quad (10)$$

при всех $i = \overline{1, M}$, $j = \overline{1, N}$. Здесь полуцелые узлы

$$x_{i\pm 1/2} = x_i \pm h_x/2, \quad y_{j\pm 1/2} = y_j \pm h_y/2. \quad (11)$$

Правая часть разностного уравнения

$$F_{ij} = \frac{1}{h_x h_y} \int_{\Pi_{ij}} F(x, y) dx dy, \quad \Pi_{ij} = \{(x, y) : x_{i-1/2} \leq x \leq x_{i+1/2}, y_{j-1/2} \leq y \leq y_{j+1/2}\} \quad (12)$$

при всех $i = \overline{1, M-1}$, $j = \overline{1, N-1}$.

Краевые условия Дирихле аппроксимируются точно равенством

$$w_{ij} = w(x_i, y_j) = 0, \quad (x_i, y_j) \in \Gamma. \quad (13)$$

Введя стандартные обозначения для разностной производной вперед и назад, разностная схема (9) записывается в следующем виде

$$-(aw_{\bar{x}})_{x,ij} - (bw_{\bar{y}})_{y,ij} = F_{ij}, \quad i = \overline{1, M-1}, j = \overline{1, N-1}. \quad (14)$$

Получаем систему линейный уравнений

$$Aw = -(aw_{\bar{x}})_x - (bw_{\bar{y}})_y = F. \quad (15)$$

Здесь A – самосопряженный и положительно определенный оператор.

Метод решения СЛАУ

Приближенное решение разностной схемы (9) получено итерационным методом сопряженных градиентов. Для ускорения сходимости метода применяется диагональное предобуславливание. Оператор $D : H \rightarrow H$ действует на сеточные функции $w \in H$ по правилу

$$(Dw)_{ij} = [(a_{i+1j} + a_{ij})/h_x^2 + (b_{ij+1} + b_{ij})/h_y^2] w_{ij}, \quad i = \overline{1, M-1}, j = \overline{1, N-1}. \quad (16)$$

Начальное приближение $w^{(0)}$ выбирается нулевым. Первая итерация совершаются по формулам скорейшего спуска. Пусть $r^{(0)} = B - Aw^{(0)} = B$ – невязка начального приближения, а функция $z^{(0)} \in H$ удовлетворяет уравнению $Dz^{(0)} = r^{(0)}$. Направление спуска $p^{(1)} = z^{(0)}$, а шаг вдоль направления спуска определяется параметром

$$\alpha_1 = \frac{(z^{(0)}, r^{(0)})}{(Ap^{(1)}, p^{(1)})}. \quad (17)$$

Следующее приближение $w^{(1)}$ вычисляется согласно равенству

$$w^{(1)} = w^{(0)} + \alpha_1 p^{(1)}. \quad (18)$$

Дальнейшие вычисления производятся по следующим формулам. Пусть выполнено k итераций метода и функции $r^{(k-1)}, z^{(k-1)}, p^{(k)}, w^{(k)} \in H$, а также коэффициент α_k являются известными.

$$r^{(k)} = r^{(k-1)} - \alpha_k Ap^{(k)}, \quad Dz^{(k)} = r^{(k)}. \quad (19)$$

$$p^{(k+1)} = z^{(k)} + \beta_{k+1} p^{(k)}, \quad \beta_{k+1} = \frac{(z^{(k)}, r^{(k)})}{(z^{(k-1)}, r^{(k-1)})}. \quad (20)$$

$$w^{(k+1)} = w^{(k)} + \alpha_{k+1} p^{(k+1)}, \quad \alpha_{k+1} = \frac{(z^{(k)}, r^{(k)})}{(Ap^{(k+1)}, p^{(k+1)})}. \quad (21)$$

В качестве условия остановки итерационного процесса использовано неравенство

$$||w^{(k+1)} - w^{(k)}||_E < \delta \quad (22)$$

Метод сопряженных градиентов является методом вариационного типа, в основе которого находится задача минимизации квадратичного функционала

$$J(w) = 0.5(Aw, w) - (B, w), \quad (23)$$

который должен монотонно убывать на итерационной последовательности $w^{(k)}$, $k = 0, 1, \dots$

Последовательный код и результаты

Для получения коэффициентов F_{ij} было использовано упрощенное правило: если клетка Π_{ij} целиком содержится в области \bar{D} , то $F_{ij} = 1$, иначе $F_{ij} = 0$. До этого использовалась формула шнурования и Sutherland-Hodgman algorithm для расчета площади пересечения. В качестве констант δ и ϵ взяты $1e - 10$ и $1/\max(h_x^2, h_y^2)$ соответственно.

Ниже представлены графики приближенных решений, а также соответствующие им графики энергетических функционалов $J(w^{(k)})$ и норм $\|w^{(k+1)} - w^{(k)}\|_E$ на сгущающихся сетках. Для построения графиков были использованы библиотеки matplotlib и plotly языка Python.

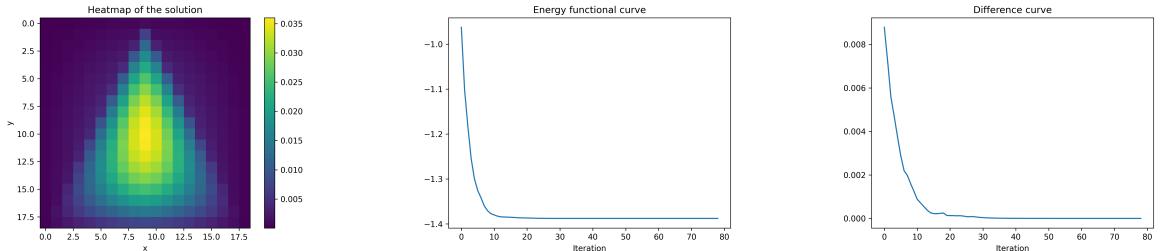


Figure 1: $M = 20, N = 20$

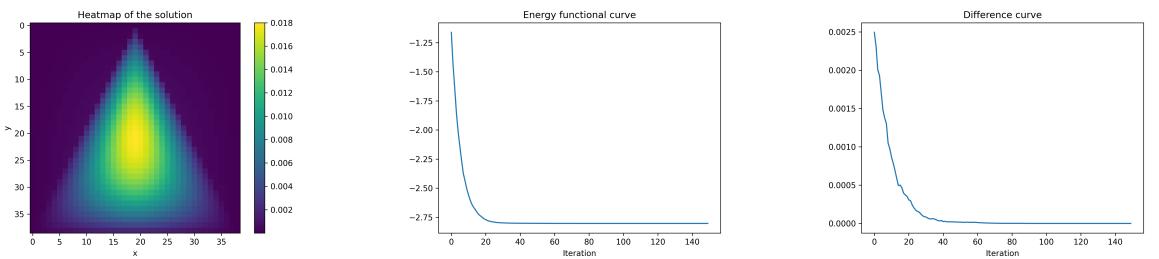


Figure 2: $M = 40, N = 40$

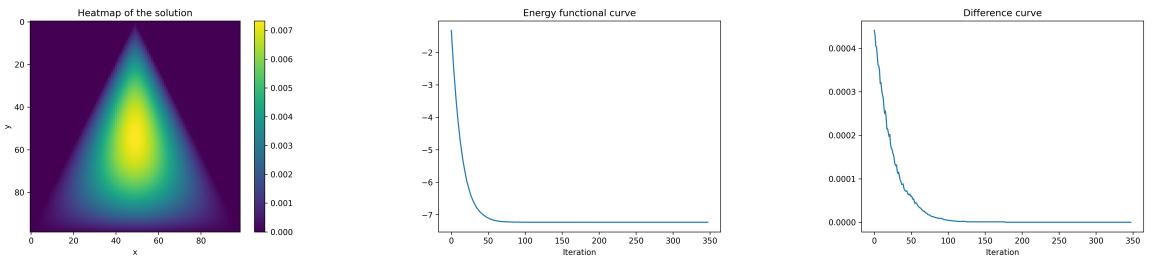


Figure 3: $M = 100, N = 100$

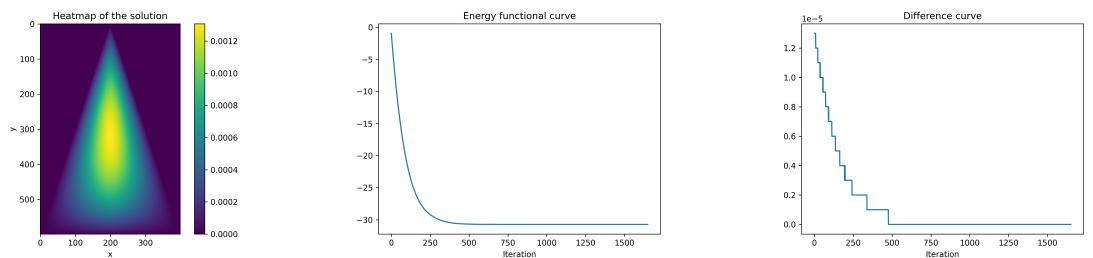


Figure 4: $M = 400, N = 600$

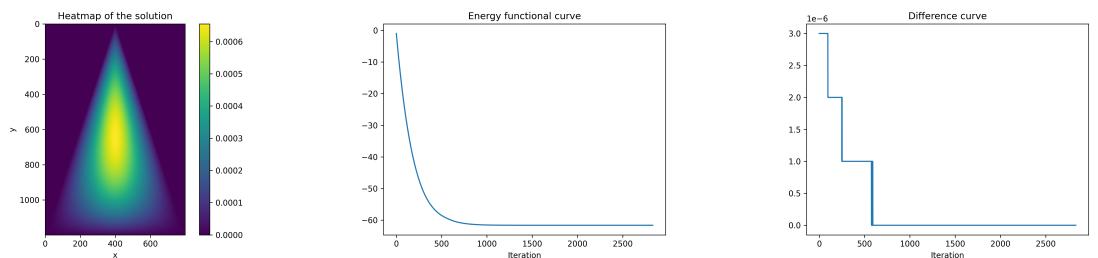


Figure 5: $M = 800, N = 1200$

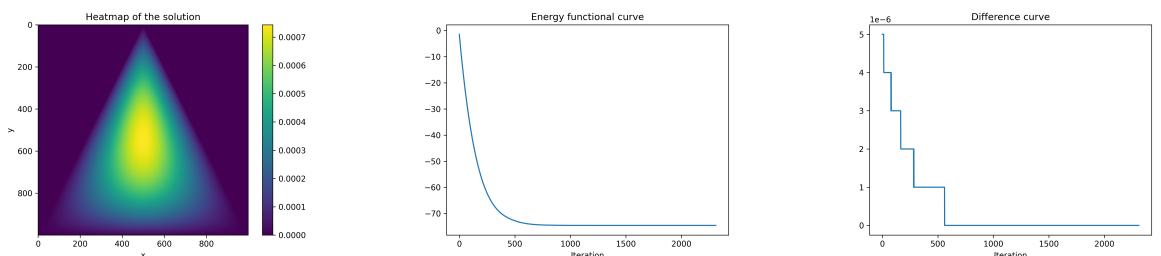


Figure 6: $M = 1000, N = 1000$

OpenMP версия

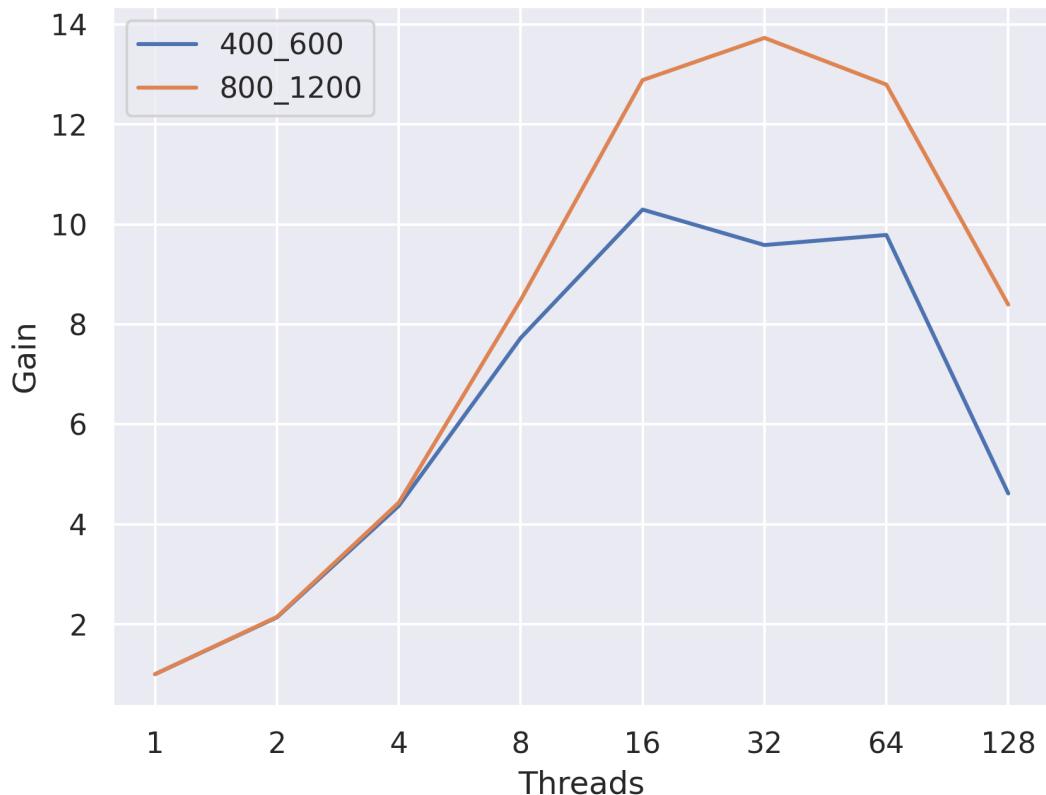
Для распараллеливания кода использовались OpenMP директивы

1. `omp parallel for simd` для простых циклов: вычисление сеточной функции z , вычисление суммы двух векторов.
2. `omp parallel for simd reduction(+:x)` для вычисления скалярного произведения ($x = \text{dot_product}$) и для вычисления функционала энергии ($x = \text{energy}$).
3. `omp parallel for collapse(2)` для вложенных циклов: вычисление параметров a_{ij}, b_{ij}, F_{ij} , для вычисления элементов матрицы D , а также для вычисления величины $Ap^{(k)}$.

Для вычисления времени работы последовательной программы использовался тип `clock_t` и функция `clock()`, а для параллельной – функция `omp_get_wtime()`. Ниже приведена таблица, содержащая информацию о времени вычисления решения на сетках $M = 400, N = 600$ и $M = 800, N = 1200$.

OpenMP-нити	$M \times N$	Число итераций	Время решения	Ускорение
1	400×600	1651	8.58924	1.00
2	400×600	1651	4.01883	2.13
4	400×600	1651	1.96632	4.36
8	400×600	1651	1.11156	7.72
16	400×600	1651	0.83429	10.29
32	400×600	1651	0.89607	9.58
64	400×600	1651	0.87758	9.78
128	400×600	1651	1.86056	4.61
1	800×1200	2833	59.47362	1.00
2	800×1200	2833	27.70785	2.14
4	800×1200	2833	13.41475	4.43
8	800×1200	2833	7.00864	8.48
16	800×1200	2833	4.61571	12.88
32	800×1200	2833	4.33175	13.72
64	800×1200	2833	4.64785	12.79
128	800×1200	2833	7.08512	8.39

Графики ускорения:



MPI версия

Необходимо разработать двумерное разбиение прямоугольника Π на подобласти так, чтобы

1. отношение количества узлов по переменным x и y в каждой подобласти принадлежало диапазону $[0.5, 2]$
2. количество узлов по переменным x и y любых двух подобластей отличалось не более, чем на единицу

В работе были использованы одинаковые прямоугольные подобласти:

1	2	4	8	16	32
400×600	400×300	200×300	200×150	100×150	100×75
800×1200	800×600	400×600	400×300	200×300	200×150

Каждый MPI процесс вычисляет коэффициенты $a_{ij}, b_{ij}, F_{ij}, D_{ij}$ в своей подобласти и производит расчёт необходимых величин для итерации метода сопряженных градиентов. При вычислении произведения вектора p на матрицу A процессы производят обмен граничными узлами (так называемые ghost cells) при помощи неблокирующих MPI функций `MPI_Isend()` и `MPI_Irecv()`. Для корректного сохранения отправленных границ были использованы тэги `TAG_LEFT`, `TAG_RIGHT`, `TAG_TOP`, `TAG_BOTTOM`. Для редукции глобального направления спуска, погрешности решения, а также величины шага использовалась функция `MPI_Allreduce()`.

Сохранение энергетического функционала, глобального решения, а также погрешности решения поручено корневому процессу (процессу с рангом 0). Стоит отметить, что глобальное решение собирается функцией `MPI_Gather()` и сохраняется в разобранном виде (следствие выбранного мной способа нумерации подобластей, а также хранения локальных решений в формате `column-major`). Было решено осуществлять корректную расстановку решений в сопровождающем скрипте на языке Python, при помощи которого строятся графики.

Время выполнения программы производилось при помощи функций `MPI_Barrier()` и `MPI_Wtime()`. Из всех времен выбрано максимальное: `MPI_Reduce` с параметром `MPI_MAX`.

Ниже приведена таблица, содержащая информацию о времени решения:

MPI-процессы	$M \times N$	Число итераций	Время решения	Ускорение
1	400×600	1651	38.2	1.00
2	400×600	1651	18.231362	2.09
4	400×600	1651	9.177342	4.16
8	400×600	1651	4.743801	8.05
16	400×600	1651	2.535652	15.06
32	400×600	1651	1.583696	24.12
1	800×1200	2833	268.6	1.00
2	800×1200	2833	117.475497	2.29
4	800×1200	2833	59.182680	4.54
8	800×1200	2833	29.821165	9.00
16	800×1200	2833	15.684085	17.12
32	800×1200	2833	8.140929	32.99

Графики ускорения:

