Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики

Отчёт по заданию в рамках курса «Суперкомпьютерное моделирование и технологии»
Численное решение задачи Дирихле для уравнения Пуассона в криволинейной области

Выполнил: Анжиганов Д.А.
608 группа
Вариант 6

Москва 2023

Введение

Требуется приближенно решить задачу Дирихле для уравнения Пуассона в криволинейной области. Задание необходимо выполнить на ПВС Московского университета IBM Polus.
Исследуемая область $D = |x| + |y| < 2,\ y < 1$

Математическая постановка задачи

В области $D \subset R^2$, ограниченной контуромγ, рассматривается дифференциальное уравнение Пуассона

$$-\Delta u = f(x, y)$$

в котором оператор Лапласа

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

функция $f(x, y)$ = 1. Для выделения единственного решения уравнение дополняется граничными условием Дирихле:

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

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

Численный метод решения уравнения

Для решения был выбран предложенный метод наименьших невязок. Этот метод позволяет получить последовательность сеточных функций $\omega^{(k)} \in H$, k = 1, 2, ..., сходящуюся по норме пространства H к решению разностной схемы, т.е.

$$\left|\left|\omega - \omega^{(k)}\right|\right|_E \to 0,\ k \to \infty$$

Начальное приближение $\omega^{(0)}$ можно выбрать любым способом, например, равным нулю во всех точках расчетной сетки. Метод является одношаговым. Итерация $\omega^{(k+1)}$ вычисляется по итерации $\omega^{(k)}$ согласно равенствам:

$$\omega_{ij}^{(k+1)} = \omega_{ij}^{(k)} - \tau_{k+1} r_{ij}^{(k)}$$

где невязка $r^k = A\omega^{(k)} - B$, итерационный параметр

$$\tau_{k+1} = \frac{(Ar^{(k)}, r^{(k)})}{\left\|Ar^{(k)}\right\|_E^2}$$

В качестве условия остановки итерационного процесса следует использовать неравенство

$$\left\|\omega^{(k+1)} - \omega^{(k)}\right\|_E < \delta$$

где δ – положительное число, определяющее точность итерационного метода.

Краткое описание проделанной работы по созданию OpenMP-программы

Для реализации поставленной задачи была использована технология OpenMP.

При подсчете площадей пересечения данной в 6 варианте фигуры с областью $П_{ij}$ были использованы два способа. Первый заключался в разбиении сложной фигуры на сумму более простых. Под простыми фигурами, как правило понимаются трапеции и треугольники. Такой путь оказался весьма трудоемким поскольку при подсчете было необходимо разобрать большое множество всевозможных вариантов пересечения. В связи с этим был придуман более простой в реализации способ - пересечение области

$П_{ij} = \{(x, y): x_{i-1/2} \le x \le x_{i+1/2}, y_{j-1/2} \le y \le y_{j+1/2}\}$ с фигурой определяется разбиением этой области на ещё 100*100 узлов. Таким образом появляется возможность определить количество точек принадлежащих пересечению. После этого отношение найденного количества и общего количества узлов в области $П_{ij}$ умножается на площадь области $П_{ij}$.

Для реализации распараллеливания использовались директивы:

#pragma omp parallel default(shared) private(i) для арифметических операций

#pragma omp parallel default(shared) private(i) reduction(+:sum) для скалярного произведения

Краткое описание проделанной работы по созданию MPI-программы

Для реализации поставленной задачи была использована технология OpenMP.

При подсчете площадей пересечения данной в 6 варианте фигуры с областью $П_{ij}$ был использован метод Монте-Карло - численный метод для изучения случайных процессов. После чего, прямоугольник был разделён на 4 квадранта.

Сделано это для того чтобы суметь выполнить оптимизационную задачу параллельно. Данные передаются между процессами. Каждый процесс содержит определенное число квадрантов. Основным потоком является нулевой поток. Таким образом выполняется ускорение работы.

Вместе с этим удается распараллелить процессы на потоки благодаря OpenMP/

Результаты расчетов для разных размеров задач и на разном числе процессов.

Число OpenMP-нитей	Число точек сетки M×N	Время решения (с)	Ускорение
2	80*80	29.1113	1.73
4	80*80	18.6243	3.04
8	80*80	12.9444	3.91
16	80*80	10.1774	4.96
2	160*160	500.652	1.77
4	160*160	283.374	3.12
8	160*160	181.295	4.88
16	160*160	142.657	6.21

Таблица 1: Таблица с результатами расчетов на ПВС IBM Polus (OpenMP).

Число процессов MPI	Число точек сетки M×N	Время решения (с)	Ускорение
2	80*80	25.8906	1.95
4	80*80	16.6074	3.04
2	160*160	337.0098	2.63
4	160*160	208.5496	4.25

Таблица 2: Таблица с результатами расчетов на ПВС IBM Polus (MPI).

Число процессов MPI	Число OpenMP-нитей	Число точек сетки M×N	Время решения (с)	Ускорение
2	1	80*80	25.8906	1.95
2	2	80*80	17.0563	2.96
2	4	80*80	13.2511	3.81
2	8	80*80	11.9920	4.21
4	1	160*160	208.5496	4.25
4	2	160*160	170.1220	5.21
4	4	160*160	152.5535	5.81
4	8	160*160	130.5354	6.79

Таблица 3: Таблица с результатами расчетов на ПВС IBM Polus (MPI и OpenMP).

Ускорение считалось как отношение времени выполнения последовательной программы к времени выполнения программы на определённой конфигурации программы для заданного числа точек сетки M*N, числа нитей OpenMP и процессов MPI.
Таким образом, было разобрано следующее число процессов: 2, 4, число нитей: 2, 4, 8, 16 и числа точек в сетке: 80*80 и 160*160. Время выполнения

последовательной программы для 80*80: 50.4867 сек. Время выполнения последовательной программы для 160*160: 886.336 сек.

Рисунок приближенного решения, полученного на сетке с наибольшим количеством узлов, графики ускорений.

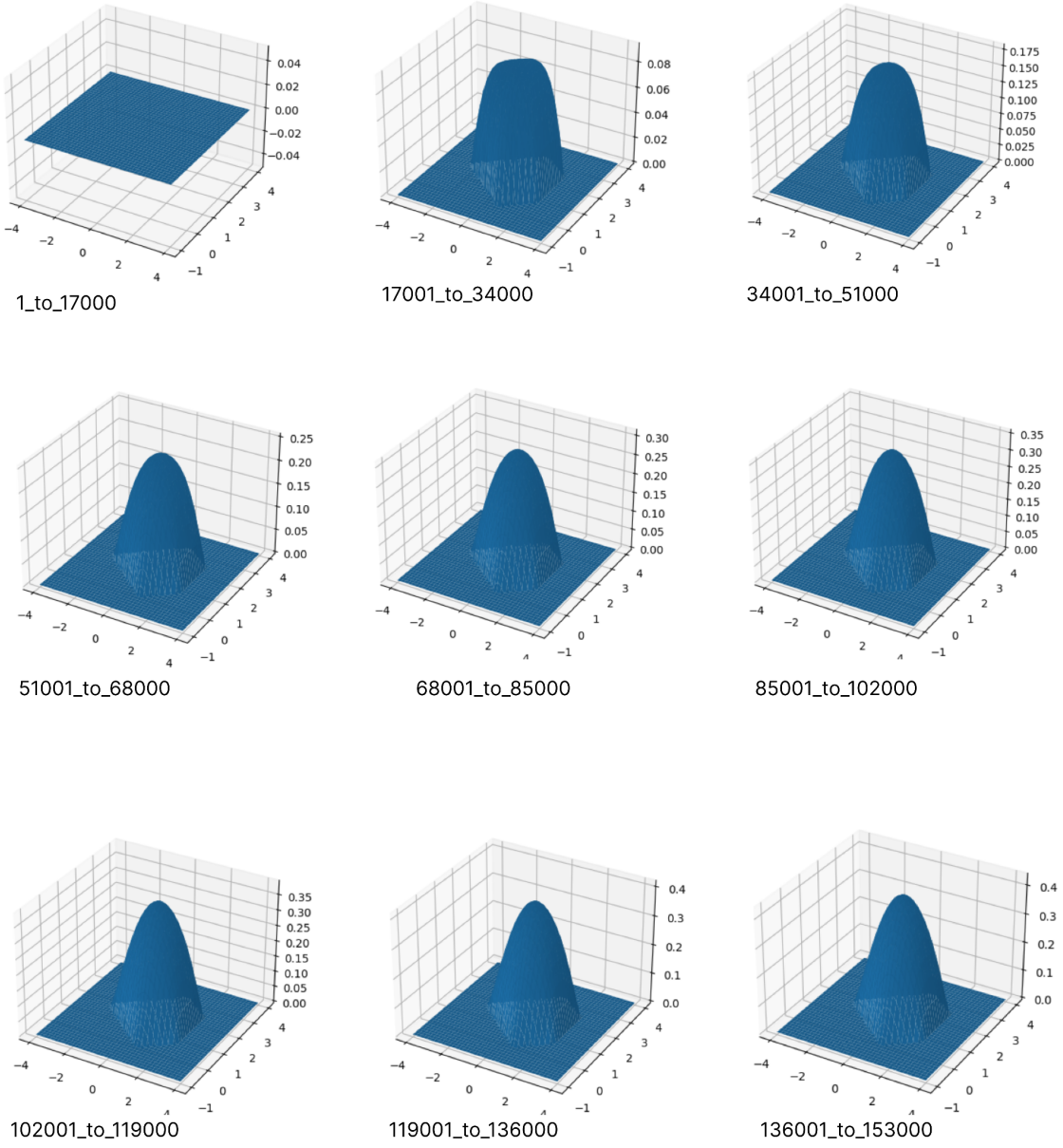


Рис 1. Получение решения.

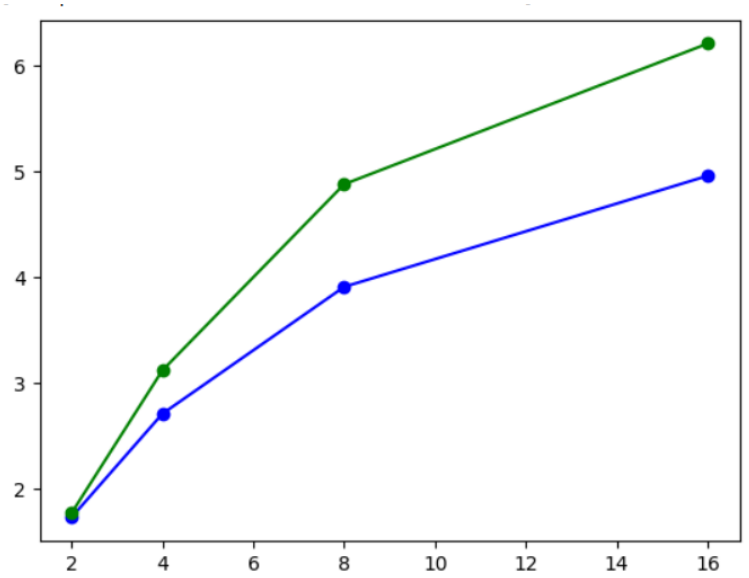
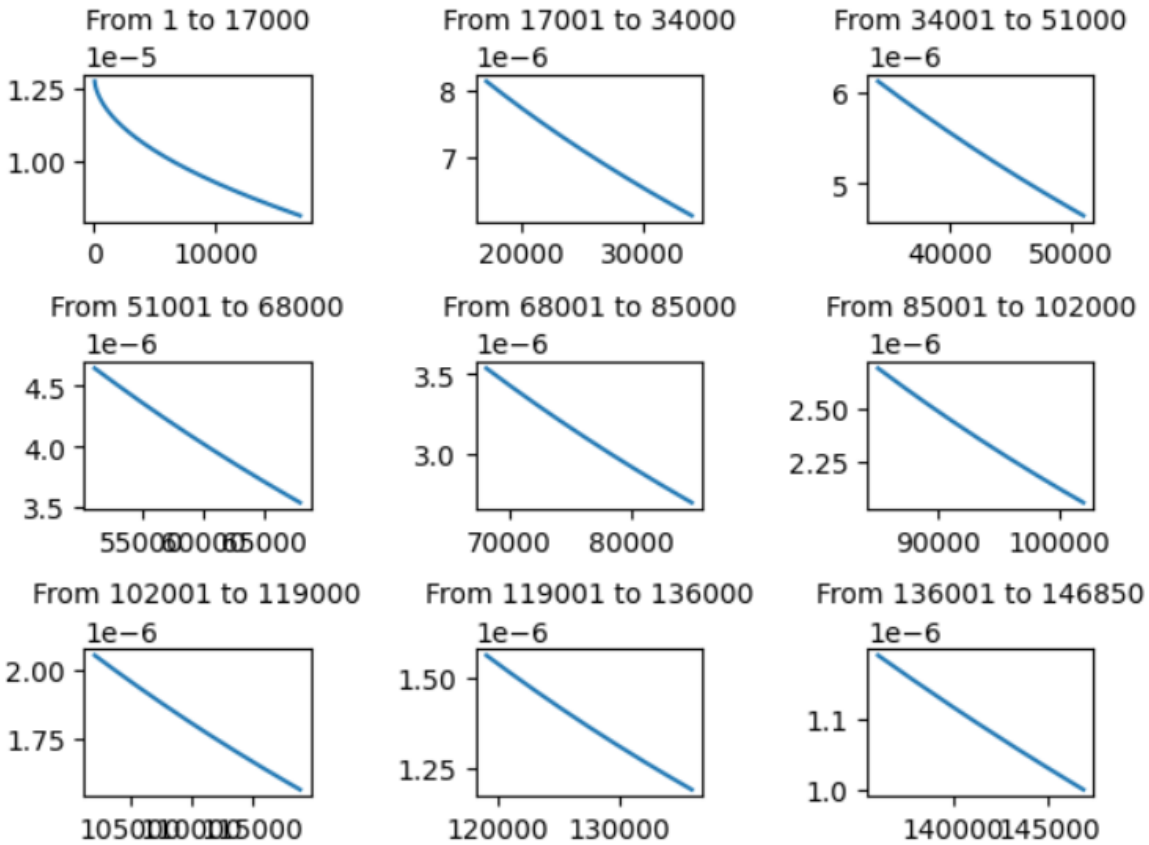


Рис 2. Зависимость ускорения по оси ординат и числа OpenMP-нитей по оси абсцисс для параметров M, N = 80 (синий) и M, N (зелёный) = 160.



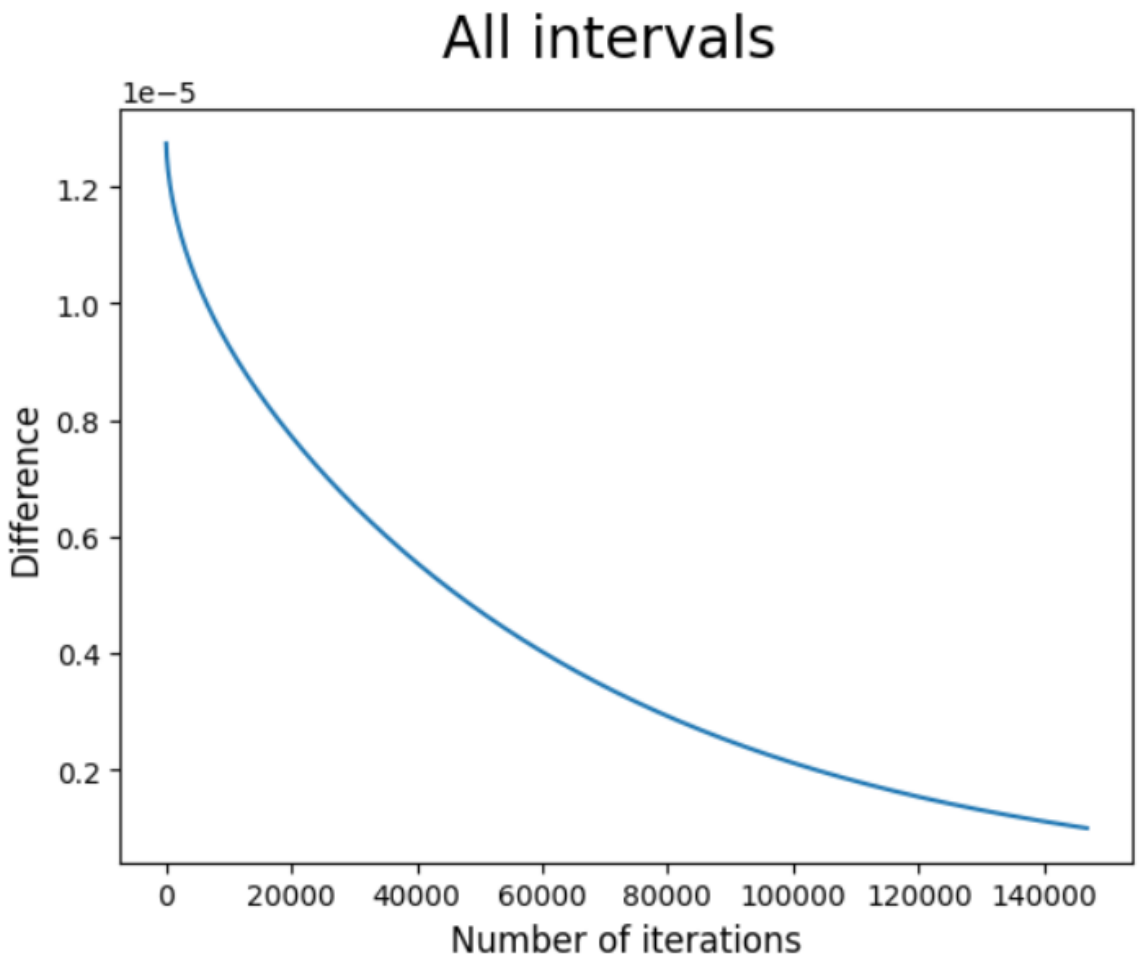


Рис 3. Графики сходимости за определённые интервалы итераций и общая сходимость за все интервалы

Список литературы

[1] IBM Polus. —http://hpc.cs.msu.su/polus