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

Суперкомпьютерное моделирование и технологии

Отчет по заданию №2
«Задача Дирихле для уравнения Пуассона в криволинейной области»

ВАРИАНТ 9

Отчет выполнил:
студент 614 группы
Артамонов Георгий

Москва, 2023

Оглавление

Глава 1

Задача Дирихле для уравнения Пуассона

1.1 Математическая постановка задачи

В области $D \subset \mathbb{R}^2$, ограниченной контуром γ, рассматривается дифференциальное уравнение Пуассона (1):

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

в котором оператор Лапласа

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

Для выделения единственного решения уравнение дополняется граничным условием Дирихле (2):

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

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

Рассмотрим задачу в случае, когда правая часть уравнения $f(x, y) = 1$, а область D представляет собой внутренность эллипса: $D = \{(x, y) \mid x^2 + 4y^2 < 1\}$

1.2 Численный метод решения

1.2.1 Метод фиктивных областей

Для решения поставленной задачи предлагается использовать метод фиктивных областей. Идея метода заключается в приближенной замене исходной задачи Дирихле в криволинейной области задачей Дирихле в прямоугольнике с кусочно-постоянным коэффициентом $k(x, y)$. Пусть область D принадлежит прямоугольнику $\Pi = \{(x, y) \mid A_1 < x < B_1, A_2 < y < B_2\}$. Разность множеств Π и $\bar{D}$ обозначим $\hat{D} = \Pi \setminus \bar{D}$, границу прямоугольника Π обозначим Γ.
В прямоугольнике Π рассмотрим задачу Дирихле (3):

$$
\begin{aligned}
&-\frac{\partial}{\partial x}(k(x,y)\frac{\partial v}{\partial x}) - \frac{\partial}{\partial y}(k(x,y)\frac{\partial v}{\partial y}) = F(x,y) \\
&v(x,y) = 0, (x,y) \in \Gamma
\end{aligned}
$$

с кусочно-постоянным коэффициентом:

$$
k(x,y) = \begin{cases} 1, & (x,y) \in D, \\ 1/\varepsilon & (x,y) \in \hat{D} \end{cases}
$$

и правой частью:

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

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

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

имел непрерывную нормальную компоненту на общей части криволинейной границы области D и прямоугольника Π.
Переход к новой задаче позволяет получить решение исходной задачи с любой наперед заданной точностью $\varepsilon > 0$, решая при этом задачу Дирихле в прямо-

угольнике П, содержащем исходную область.

$$\max_{P \in \bar{D}} \|v(x,y) - u(x,y)\| \le C\varepsilon, C > 0$$

Для случая, когда область D представляет собой внутренность эллипса, выберем прямоугольник $\Pi = \{(x,y) \mid -1.0 < x < 1.0, -0.5 < y < 0.5\}$.

1.2.2 Разностная схема решения

В замыкании прямогольника $\bar{\Pi}$ определим равномерную прямоугольную сетку $\bar{\omega}_h = \bar{\omega}_1 \times \bar{\omega}_2$, где

$$\bar{\omega}_1 = \{x_i = A_1 + ih_1, i = 0, \ldots, M\}, \quad h_1 = (B_1 - A_1)/M$$
$$\bar{\omega}_2 = \{y_j = A_2 + jh_2, j = 0, \ldots, N\}, \quad h_2 = (B_2 - A_2)/N$$

Множество внутренних узлов сетки $\bar{\omega}_h$ обозначим ω_h.
Рассмотрим линейное пространство H функций, заданных на сетке ω_h.
Обозначим через w_{ij} значение сеточной функции H в узле сетки $(x_i, y_j) \in \omega_h$.
Определим скалярное произведение и норму в пространстве сеточных функций H:

$$(u,v) = \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} h_1 h_2 u_{ij} v_{ij}$$

$$\|u\| = \sqrt{(u,u)}$$

Будем использовать метод конечных разностей, который заключается в замене дифференциальной задачи математической физики на конечно-разностную операторную задачу вида:

$$A\,\omega = B$$

$$A : H \to H$$

Дифференциальное уравнение задачи (3) во всех внутренних точках сетки аппроксимируется разностным уравнением:

$$-\frac{1}{h_1}(a_{i+1j}\frac{\omega_{i+1j}-\omega_{ij}}{h_1}-a_{ij}\frac{\omega_{ij}-\omega_{i-1j}}{h_1})-\frac{1}{h_2}(b_{ij+1}\frac{\omega_{ij+1}-\omega_{ij}}{h_2}-b_{ij}\frac{\omega_{ij}-\omega_{ij-1}}{h_2})=F_{ij}$$
$i=1,\ldots,M-1,\ j=1,\ldots,N-1$

в котором коэффициенты при $i=1,\ldots,M,\ j=1,\ldots,N$

$$a_{ij}=\frac{1}{h_2}\int_{y_{j-1/2}}^{y_{j+1/2}}k(x_{i-1/2},t)\,dt$$
$$b_{ij}=\frac{1}{h_1}\int_{x_{i-1/2}}^{x_{i+1/2}}k(t,y_{j-1/2})\,dt$$

и правая часть при $i=1,\ldots,M-1,\ j=1,\ldots,N-1$

$$F_{ij}=\frac{1}{h_1h_2}\iint_{\Pi_{ij}}F(x,y)\,dxdy$$
$$\Pi_{ij}=\{(x,y):x_{i-1/2}\le x\le x_{i+1/2},y_{j-1/2}\le y\le y_{j+1/2}\}$$

Краевые условия Дирихле в задаче (3) аппроксимируются точно равенством

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

Полуцелые узлы означают:

$$x_{i\pm1/2}=x_i\pm0.5h_1,\ y_{j\pm1/2}=y_j\pm0.5h_2$$

Полученная система является линейной относительно неизвестных величин и может быть представлена в виде $A\omega=B$ с самосопряженным и положительно определенным оператором A. Построенная разностная схема линейна и имеет единственное решение при любой правой части.

Интегралы a_{ij},b_{ij} будем вычислять аналитически: $a_{ij}=h_2^{-1}l_{ij}+(1-h_2^{-1}l_{ij})/\varepsilon$, где l_{ij} длина части отрезка $[y_{j-1/2},y_{j+1/2}]$, которая принадлежит области D. Для вычисления l_{ij} для заданного $\hat{x}=x_{i-1/2}$ вычислим точки пересечения прямой

$x = \hat{x}$ с границей эллипса γ. Тогда $l_{ij} = \min(y_1, y_{j+1/2}) - \max(y_2, y_{j-1/2})$, где

$$y_{1,2} = \pm\frac{1}{4}\sqrt{1-\hat{x}^2}$$

Правую часть разностной схемы приближенно заменим на значение в центре квадрата Π_{ij}:

$$F_{ij} = F(x_i, y_j) = \begin{cases} 1, & (x_i, y_j) \in D, \\ 0, & (x_i, y_j) \in \hat{D} \end{cases}$$

1.2.3 Метод минимальных невязок

Приближенное решение разностной схемы предлагается вычислять методом наименьших невязок. Метод позволяет получить последовательность сеточных функций $\omega^{(k)} \in H$, $k = 1, 2, \ldots$, сходящуюся по норме пространства H к решению разностной схемы.

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

Начальное приближение $\omega^{(0)}$ выберем равным нулю во всех точках сетки. Итерация $\omega^{(k+1)}$ вычисляется по итерации $\omega^{(k)}$ по формуле:

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

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

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

В качестве критерия останова можно использовать условие:

$$\|r^{(k)}\|_E < \delta$$

с некоторой положительной константой $\delta > 0$, задающей точность приближенного решения.

1.3 Описание программной реализации

Для выполнения задания был разработан последовательный код, представляющий собой программу на языке C sequantial.c, реализующую описанный численный метод. Были выполнены расчеты на сгущающихся сетках $(M, N) = (10, 10), (20, 20), (40, 40)$ и построены графики полученных приближенных решений. Для написания parallel.c вложенные циклы в функциях, вызывающихся на каждой итерации метода минимальных невязок (scalar_product, apply_diff_operator, linear_comb), были размечены с помощью директивы OpenMP: pragma omp parallel for collapse(2). Были проведены расчеты на сетках (40, 40), (80, 80), (160, 160) на разном числе потоков. Полученные приближенные решения совпали с соответствующими решениями при последовательных вычислениях, но время их вычисления удалось уменьшить за счет использования параллелизма.

В рамках второй части задания была разработана MPI программа mpi-parallel.c, реализующая аналогичную разностную схему. Принципиальным отличием от OpenMP версии программы стало построчное хранение матриц по указателю на double для упрощения обменов данными между процессами, в OpenMP версии использовался массив указателей на double.

Разбиение прямоугольника П на домены выполняет функция init_local_grid_sizes, в зависимости от числа процессов и размеров исходной сетки данная функция определяет размеры сеток для доменов. Каждому домену соответствует свой номер. Для определения идентификатора домена, в который попадает узел (i, j) исходной сетки используется макрос DOMAIN_ID. Каждый MPI процесс работает в своем домене и проводит в нем локальные расчеты. Для обмена данными на границах областей функция get_grid_border запаковывает значения в граничных точках сетки домена в массив, после чего при помощи MPIAllgather происходит обмен граничными значениями. Также процессы на каждой итерации обмениваются вычисленными скалярными произведениями: $(Ar^{(k)}, Ar^{(k)})$, $(Ar^{(k)}, r^{(k)})$, $(r^{(k)}, r^{(k)})$.

Для написания гибридной программы mpi-omp-parallel.c в MPI версию программы были добавлены директивы OpenMP аналогично parallel.c.

Результаты расчетов приведены в таблицах далее.

1.3.1 Результаты расчетов для OpenMP программы

Число OpenMP-нитей	Число точек сетки $M \times N$	Время решения	Ускорение
2	40×40	91.549	1.298
4	40×40	57.722	2.059
6	40×40	48.929	2.430
8	40×40	45.768	2.597
16	40×40	46.381	2.563
2	80×80	337.001	1.599
4	80×80	183.397	2.938
6	80×80	130.966	4.106
8	80×80	111.136	4.848
16	80×80	90.581	5.948
2	160×160	321.049	1.689
4	160×160	164.734	3.292
8	160×160	86.978	6.235
16	160×160	53.271	10.180
32	160×160	51.108	10.611

Таблица 1.1: Зависимость времени решения от числа нитей для разных сеток

Был проведен дополнительный эксперимент для сетки $(40, 40)$. Последовательный и параллельный варианты программы были скомпилированы с флагом жесткой оптимизации -O3. При любом числе потоков получили проигрыш по сравнению с последовательным вариантом алгоритма. Компилятор GCC довольно хорошо оптимизирует последовательный код.

Число OpenMP-нитей	Число точек сетки $M \times N$	Время решения	Ускорение
2	40×40	52.303	0.676
4	40×40	39.793	0.889
6	40×40	36.659	0.965
8	40×40	36.883	0.959
16	40×40	36.525	0.969

Таблица 1.2: Зависимость времени решения от числа нитей с флагом -O3

1.3.2 Результаты расчетов для MPI программы

Число процессов MPI	Число точек сетки $M \times N$	Время решения	Ускорение
1	40×40	120.125	0.989
2	40×40	65.090	1.826
4	40×40	41.076	2.894
8	40×40	30.485	3.899
1	80×80	590.532	0.912
2	80×80	296.036	1.819
4	80×80	166.240	3.240
8	80×80	101.185	5.324
1	160×160	610.347	0.888
2	160×160	313.990	1.727
4	160×160	166.844	3.250
8	160×160	90.240	6.009

Таблица 1.3: Результаты для программы MPI

Число процессов MPI	Число OpenMP-нитей	Сетка	Время	Ускорение
2	1	80×80	381.436	1.412
2	2	80×80	299.115	1.801
2	4	80×80	187.923	2.867
2	8	80×80	159.146	3.385
4	1	160×160	237.281	2.285
4	2	160×160	153.893	3.524
4	4	160×160	84.873	6.390
4	8	160×160	51.348	10.562

Таблица 1.4: Результаты для гибридной программы MPI + OpenMP

1.3.3 Графики ускорений

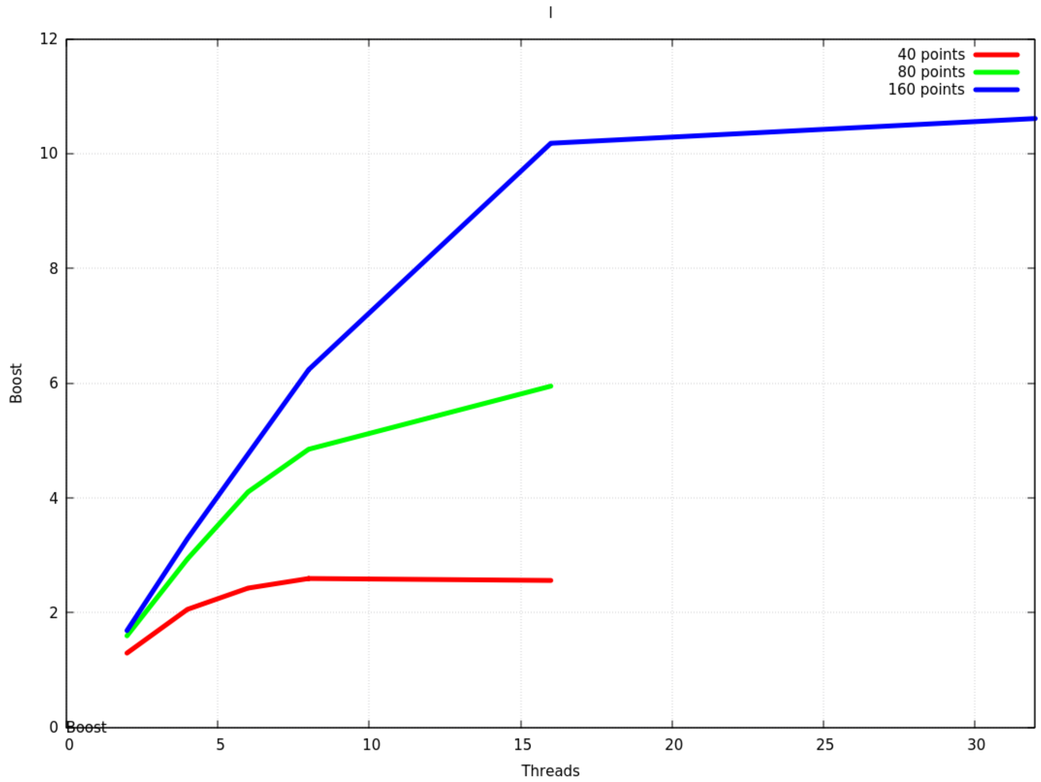

Рис. 1.1: Зависимость ускорения от числа OpenMP потоков для разных сеток

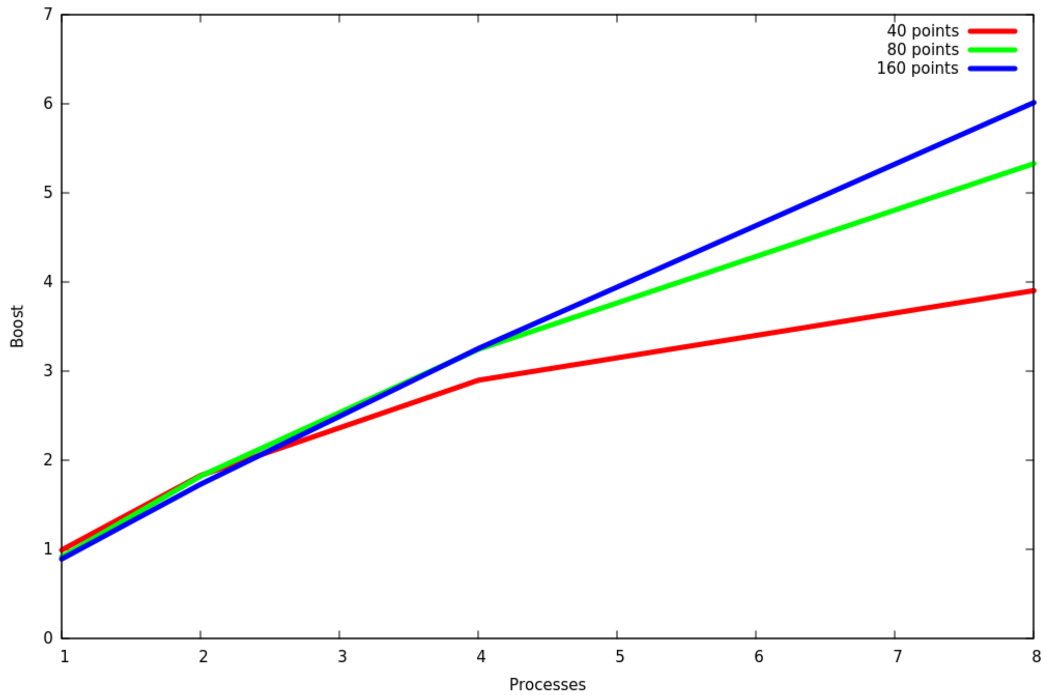

Рис. 1.2: Зависимость ускорения от числа MPI процессов для разных сеток

1.3.4 Графики нормы невязки

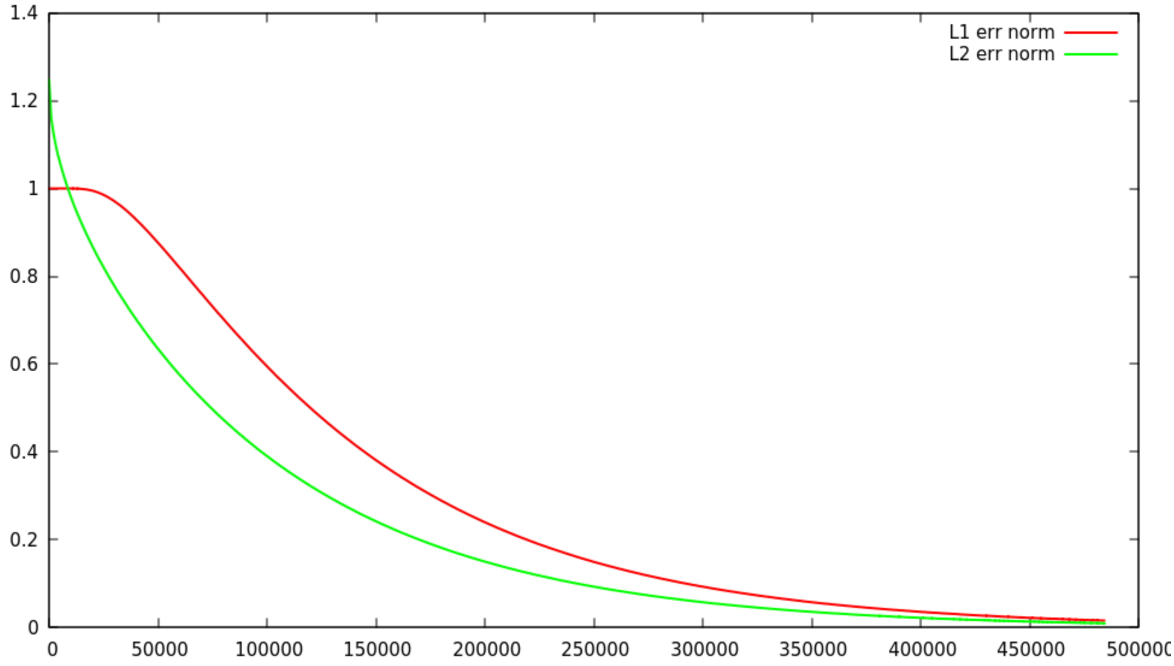

Рис. 1.3: Зависимость нормы невязки от числа итераций для сетки $(40, 40)$

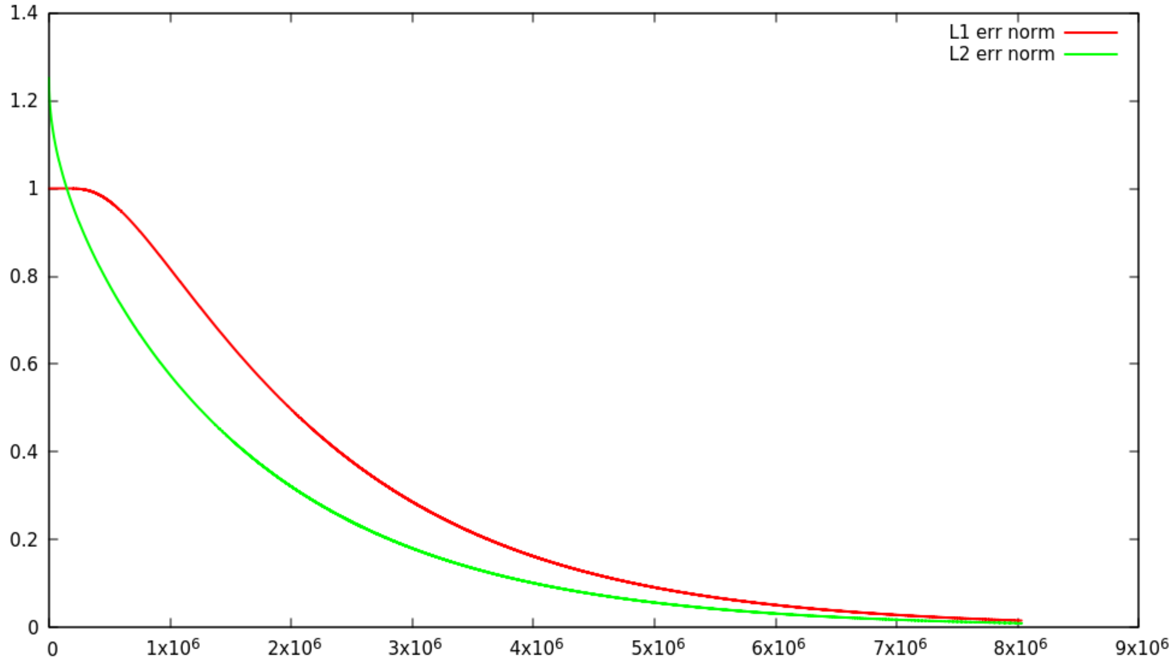

Рис. 1.4: Зависимость нормы невязки от числа итераций для сетки $(80, 80)$

1.3.5 Точка максимума невязки

Построим графики для определения точек, где ошибка принимает наибольшие значения. Каждую 1000 итераций вычисляем, в какой точке абсолютная величина ошибки равна L1 норме невязки, иными словами, в какой точке на этой итерации абсолютная величина ошибки принимает наибольшее значение. Из графиков видно, что это точка $(0, 0)$, являющаяся центром эллипса.

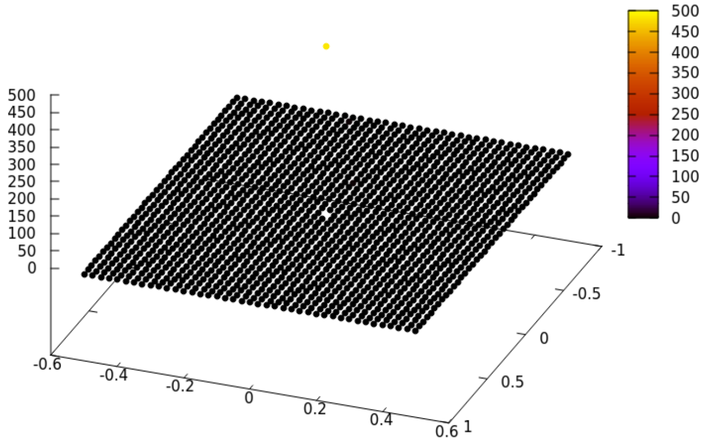

Рис. 1.5: для сетки $(40, 40)$

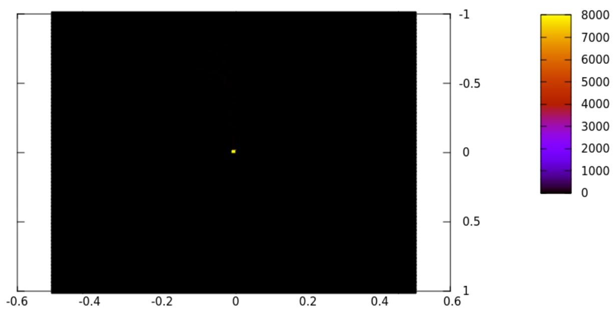

Рис. 1.6: для сетки $(80, 80)$

1.3.6 Графики приближенного решения

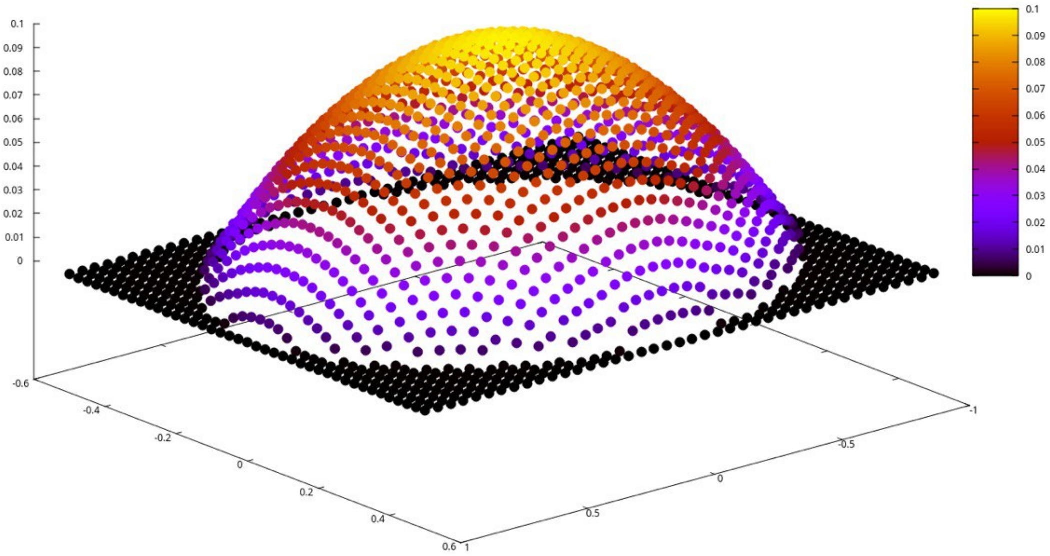

Рис. 1.7: График решения с сеткой 40×40 точек

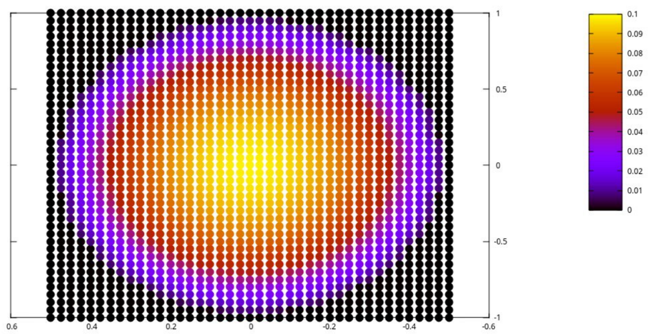

Рис. 1.8: Линии уровня с сеткой 40×40 точек

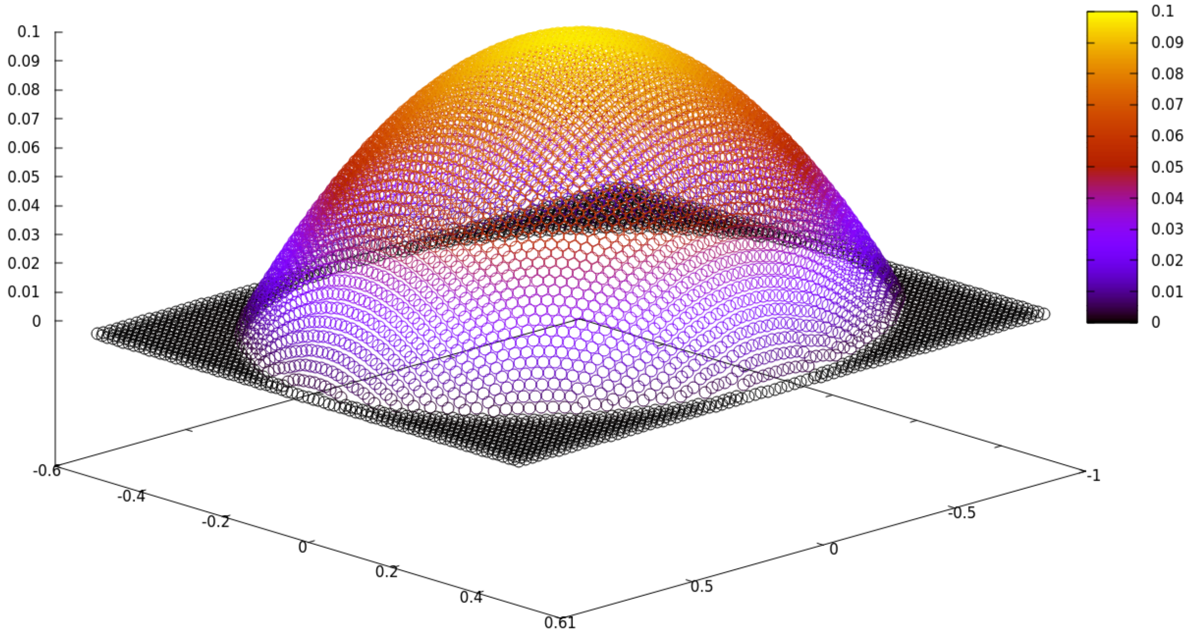

Рис. 1.9: График решения с сеткой 80×80 точек

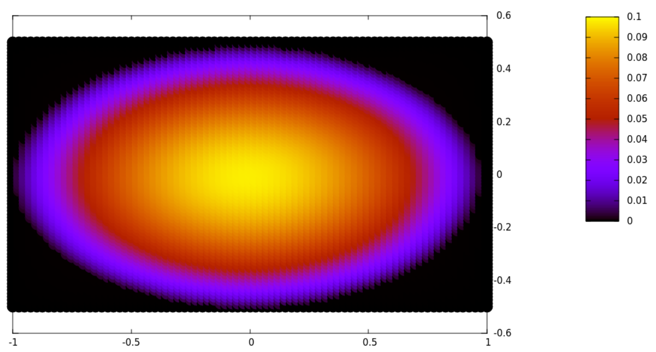

Рис. 1.10: Линии уровня с сеткой 80×80 точек