Московский государственный университет имени М. В. Ломоносова

Факультет вычислительной математики и кибернетики

Отчет

По курсу: «Суперкомпьютерное моделирование и технологии»

«OpenMP»

Цирунов Леонид Александрович

группа 628

29 октября 2024 г.

Математическая постановка задачи

В трехмерной замкнутой области

$$\Omega = \left[0 \le x \le L_x\right] \times \left[0 \le y \le L_y\right] \times \left[0 \le z \le L_z\right]$$

Для $0 < t \le T$ требуется найти решение $u\left(x,\ y,\ z,\ t\right)$ уравнения в частных производных

$$\frac{\partial^2 u}{\partial t^2} = \triangle u$$

С начальными условиями

$$u|_{t=0} = \varphi(x,\ y,\ z)$$

$$\frac{\partial u}{\partial t}|_{t=0} = 0$$

При условии, что на границах области заданы однородные граничные условия первого рода

$$\begin{aligned} &u\left(0, y,\ z,\ t\right) = 0, \qquad &&u\left(L_x,\ y,\ z,\ t\right) = 0,\\ &u(x,\ 0, z,\ t) = 0, \qquad &&u\left(x,\ L_y,\ z,\ t\right) = 0,\\ &u\left(x,\ y,\ 0, t\right) = 0, \qquad &&u\left(x,\ y,\ L_z,\ t\right) = 0 \end{aligned}$$

Либо периодические граничные условия

$$\begin{aligned} &u\left(0, y,\ z,\ t\right) = u\left(L_x,\ y,\ z,\ t\right), \qquad &&u_x\left(0, y,\ z, t\right) = u_x\left(L_x,\ y,\ z,\ t\right),\\ &u(x,\ 0, z,\ t) = u\left(x,\ L_y,\ z,\ t\right), \qquad &&u_y(x,\ 0, z, t) = u_y\left(x,\ L_y,\ z,\ t\right),\\ &u\left(x,\ y,\ 0, t\right) = u\left(x,\ y,\ L_z,\ t\right), \qquad &&u_z\left(x,\ y,\ 0, t\right) = u_z\left(x,\ y,\ L_z,\ t\right), \end{aligned}$$

Численный метод решения задачи

Введем на Ω сетку: $\omega_{h\tau} = \bar{\omega}_h \times \omega_\tau$

$T = T_0$

$L_x = L_{x_0},\ L_y = L_{y_0},\ L_z = L_{z_0}$

$\bar{\omega}_h = \{\left(x_i = ih_x,\ y_j = jh_y,\ z_k = kh_z\right),\ i, j, k = 0, 1, \ldots, N,\ h_xN = L_x, h_yN = L_y, h_zN = L_z\}$

$\omega_\tau = \left\{t_n = n\tau,\ n = 0, 1, \ldots, K,\ \tau K = T\right\}$

Через ω_h обозначим множество внутренних, а через γ_h - множество гранитных узлов сетки $\bar{\omega}_h$.

Для аппроксимации исходного уравнения с начальными условиями воспользуемся следующей системой уравнений:

$$\frac{u_{ijk}^{n+1} - 2u_{ijk}^{n} + u_{ijk}^{n-1}}{\tau^2} = \triangle_h u^n,\ \left(x_i, y_j,\ z_k\right) \in \omega_h,\ n = 1, 2, \ldots,\ K - 1$$

Где $\triangle_h$ - семиточечный разностный аналог оператора Лапласа:

$$\triangle_h u^n = \frac{u_{i-1,j,k}^n - 2u_{i,j,k}^n + u_{i+1,j,k}^n}{h^2} + \frac{u_{i,j-1,k}^n - 2u_{i,j,k}^n + u_{i,j+1,k}^n}{h^2} + \frac{u_{i,j,k-1}^n - 2u_{i,j,k}^n + u_{i,j,k+1}^n}{h^2}$$

Для начала счета должны быть заданы значения $u_{i,j,k}^0$ и $u_{i,j,k}^1$, $(x_i, y_j, z_k) \in \omega_h$.

$$u_{i,j,k}^0 = \varphi(x_i, y_j,\ z_k), \qquad (x_i, y_j, z_k) \in \omega_h.$$

$$u_{i,j,k}^1 = u_{i,j,k}^0 + \frac{\tau^2}{2} \triangle_h \varphi(x_i, y_j,\ z_k)$$

Для вычисления значений u^0 , $u^1 \in \gamma_h$ допускается использование аналитического значения , которое задается в программе еще для вычисления погрешности решения задачи.

Из варианта №8 следуют следующие формулы:

$$u_{analytical} = \sin(\frac{2\pi}{L_x}x)*\sin\left(\frac{4\pi}{L_y}y\right)*\sin\left(\frac{6\pi}{L_z}z\right)*\cos(a_t*t),$$

$$a_t = \pi\sqrt{\left(\frac{4}{{L_x}^2} + \frac{16}{{L_y}^2} + \frac{36}{{L_z}^2}\right)}$$

Граничные условия:

П: $u(0, y, z, t) = u(L_x, y, z, t), \quad u_x(0, y, z, t) = u_x(L_x, y, z, t),$
П: $u(x, 0, z, t) = u(x, L_y, z, t), \; u_y(x, 0, z, t) = u_y(x, L_y, z, t),$
П: $u(x, y, 0, t) = u(x, y, L_z, t), \; u_z(x, y, 0, t) = u_z(x, y, L_z, t)$

Алгоритм численного решения:

1. Вычисление граничных значений u^0 и u^1.
2. Вычисление u^0 внутри области: $u^0_{i,j,k} = \varphi\left(x_i, y_j, z_k\right)$
3. Вычисление u^1 внутри области: $u^1_{i,j,k} = u^0_{i,j,k} + \dfrac{\tau^2}{2} \triangle_h \varphi(x_i, y_j, z_k)$
4. Вычисление K – 1 раз u^{n+1}:
   $u^{n+1}_{ijk} = 2u^n_{ijk} - u^{n-1}_{ijk} + \tau^2 \triangle_h u^n$

Программный метод решения задачи

Программная реализация состоит из 4 файлов: ***main.cpp***, ***equation_solution.cpp***, ***equation.h*** и ***Makefile***.

Файл ***main.cpp*** содержит основную функцию получающую входные значения и запускающую решение задачи, а также две функции для сохранения результатов:

- ***dump_grid_to_CSV*** - сохранение погрешности или сетки, полученной численным или аналитическим способом, в файл в формате CSV (для дальнейшей визуализации);
- ***save_statistics*** - вывод в файл информации о времени работы решения и максимальной погрешности вычисления.

На вход программа получает 3 или 6 значений в зависимости от переданного ***L_type***. Описание параметров в порядке передачи:

- ***N*** - размер сетки по одной координате (конечный размер N^3);
- ***threads_num*** - количество потоков, которые будут задействованы;
- ***L_type*** - тип значений L по разным координатам, принимает значения: *1*, *pi*, *custom.*

В случае *1* и *pi*, значения L по всем координатам равны числу соответвенно. Если ***L_type*** передано значение *custom* , в этом случае необходимо передать значения L по каждой координате: L_x, L_y, L_z.

Файл ***equation.h*** содержит объявления типов, класса сетки с функцией получения индекса элемента в линейном массиве описывающем сетку, а также функцию $u_{analytical}$ для вычисления аналитических значений точек сетки.

Файл ***equation_solution.cpp*** содержит основной алгоритм решения задачи. Содержит функции:

- ***laplace_operator*** - вычисление значения разностного аналога оператора Лапласа;

- ***init*** - вычисление внутренних u^0 , u^1 и граничных начальных точек, необходимых для запуска алгоритма;
- ***run_algo*** - итерация по K – 1 оставшимся временным шагам алгоритма с вычислением значений граничных и внутренних точек на новом шаге алгоритма. После завершения всех шагов производится подсчет максимальной погрешности численного решения от аналитического.
- ***solve_equation*** - задается количество потоков, отключается динамическое управление количеством потоков, инициализируются переменные, выполняется запуск алгоритма и ведется подсчет времени алгоритма.

Распараллеливание производится с использованием технологии OpenMP.

Файл ***Makefile*** содержит цели для компиляции и запуска задач. Для запуска на Polus используются команды «***make compile_polus***» и «***make run_all_polus***».

Результаты расчетов

Таблица 1: Результаты при L = 1

Число OpenMP нитей	Число точек сетки N по одной оси	Время решения	Ускорение	Погрешность
0 - Sequential	128	1,55036	1,000	1.4004e-06
1	128	1,58439	0,9785	1.4004e-06
2	128	0,806249	1,9229	1.4004e-06
4	128	0,408948	3,7911	1.4004e-06
8	128	0,210073	7,3801	1.4004e-06
16	128	0,158542	9,7789	1.4004e-06
32	128	0,132751	11,6787	1.4004e-06
0 - Sequential	256	11,6665	1,000	3.49927e-07
1	256	11,9851	0,9734	3.49927e-07
2	256	6,07845	1,9193	3.49927e-07
4	256	3,06002	3,8126	3.49927e-07
8	256	1,72389	6,7675	3.49927e-07
16	256	1,23653	9,4349	3.49927e-07
32	256	1,03338	11,2897	3.49927e-07
0 - Sequential	512	90,4241	1,000	8.71479e-08
1	512	93,0409	0,9719	8.71479e-08
2	512	47,1411	1,9182	8.71479e-08
4	512	23,6495	3,8235	8.71479e-08
8	512	12,2445	7,3849	8.71479e-08
16	512	8,91206	10,1463	8.71479e-08
32	512	7,17687	12,5994	8.71479e-08

Таблица 2: Результаты при L = π

Число OpenMP нитей	Число точек сетки N по одной оси	Время решения	Ускорение	Погрешность
0 - Sequential	128	1,55327	1,000	1.41974e-07
1	128	1,58615	0,9793	1.41974e-07
2	128	0,806694	1,9255	1.41974e-07
4	128	0,480756	3,2309	1.41974e-07
8	128	0,212061	7,3246	1.41974e-07
16	128	0,157942	9,8344	1.41974e-07
32	128	0,133489	11,6359	1.41974e-07
0 - Sequential	256	11,6618	1,000	3.55074e-08
1	256	11,9916	0,9725	3.55074e-08
2	256	6,07655	1,9191	3.55074e-08
4	256	3,06577	3,8039	3.55074e-08
8	256	1,82984	6,3731	3.55074e-08
16	256	1,18342	9,8543	3.55074e-08
32	256	0,938112	12,4311	3.55074e-08
0 - Sequential	512	90,5702	1,000	8.8744e-09
1	512	93,196	0,9718	8.8744e-09
2	512	47,2076	1,9186	8.8744e-09
4	512	24,0255	3,7698	8.8744e-09
8	512	12,2912	7,3687	8.8744e-09
16	512	8,87994	10,1994	8.8744e-09
32	512	7,20551	12,5696	8.8744e-09

График 1: зависимость ускорения от количества потоков при L = 1:

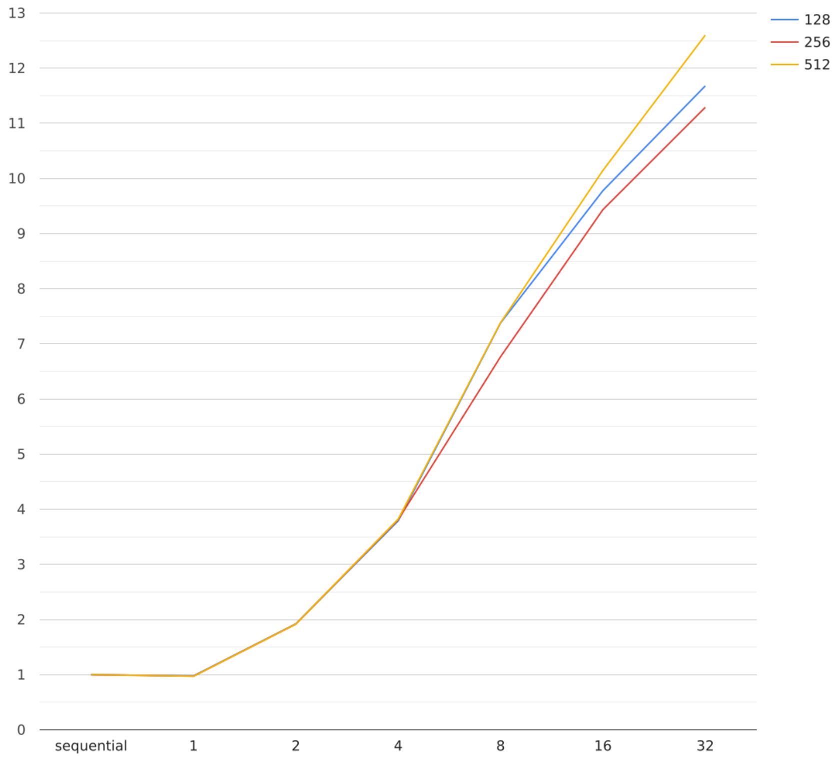
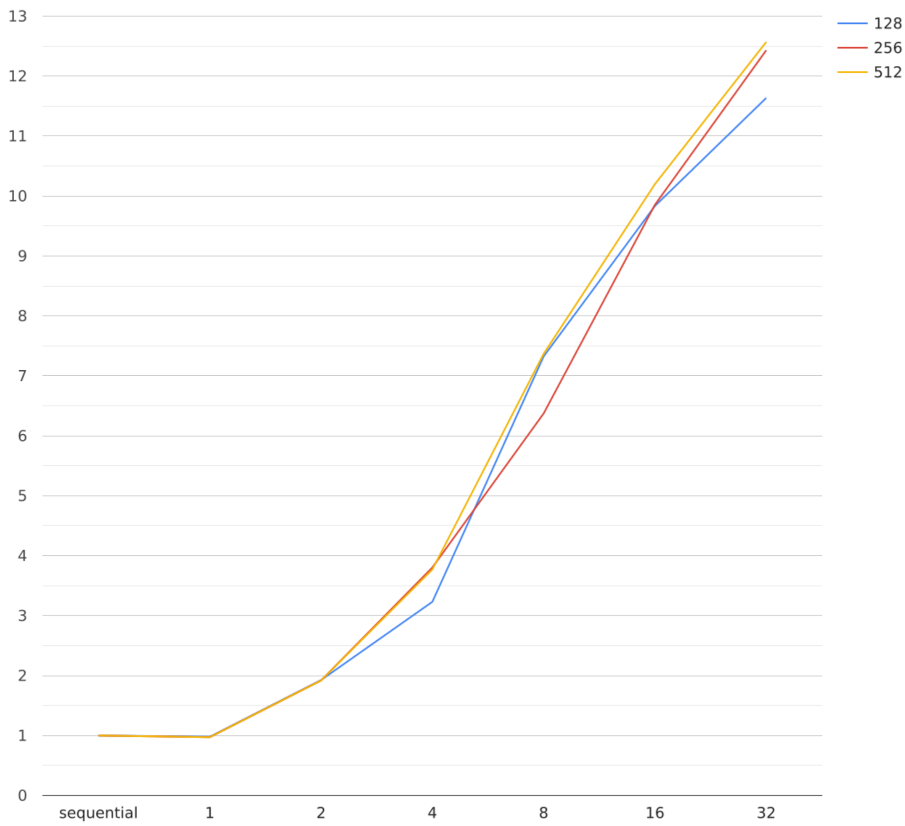
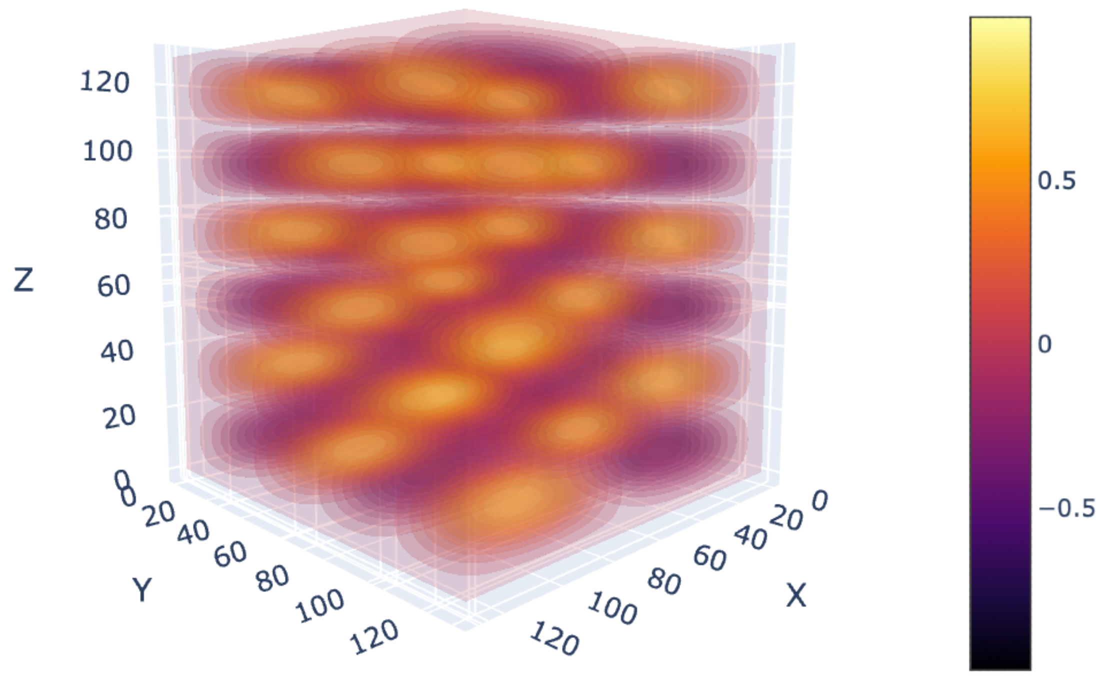
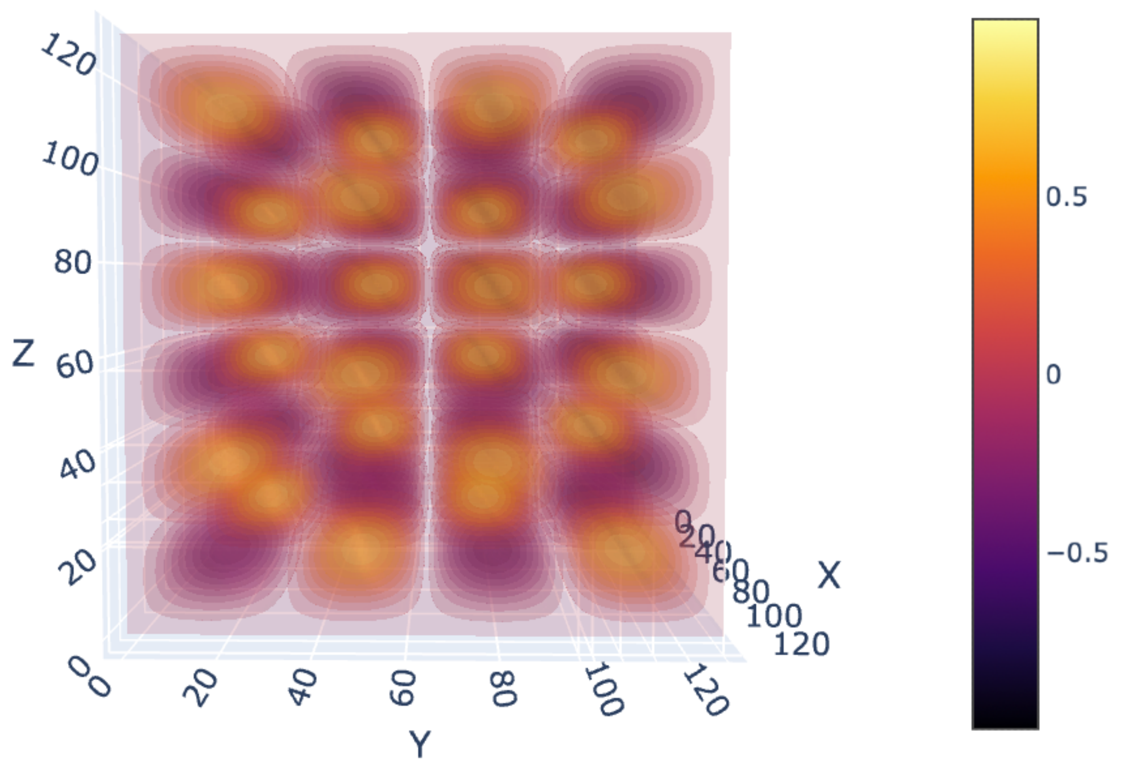


График 2: зависимость ускорения от количества потоков при L = π:

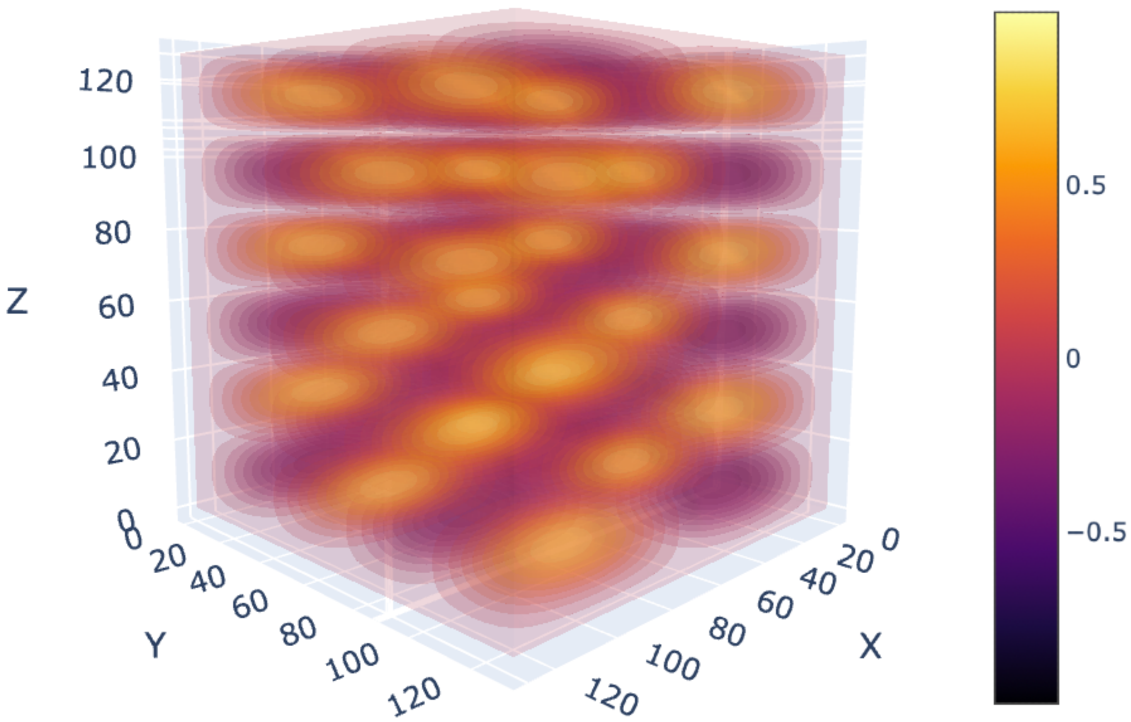


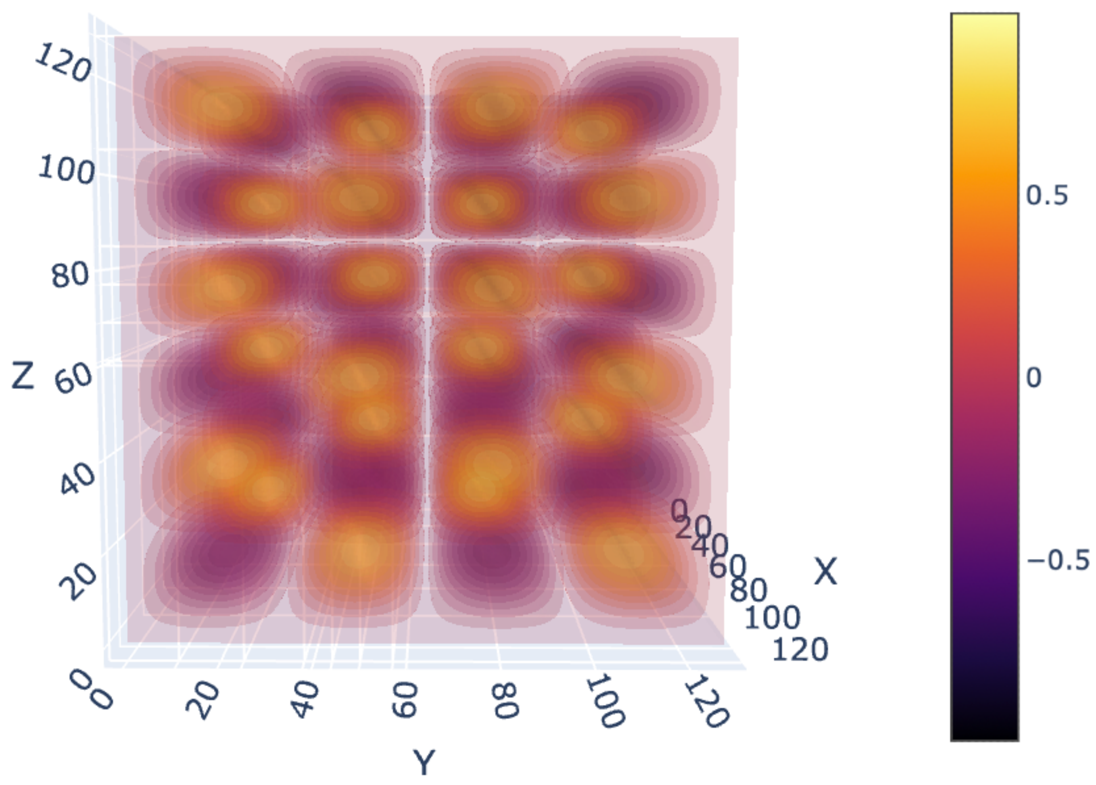
Визуализация сетки полученной аналитическим решением:



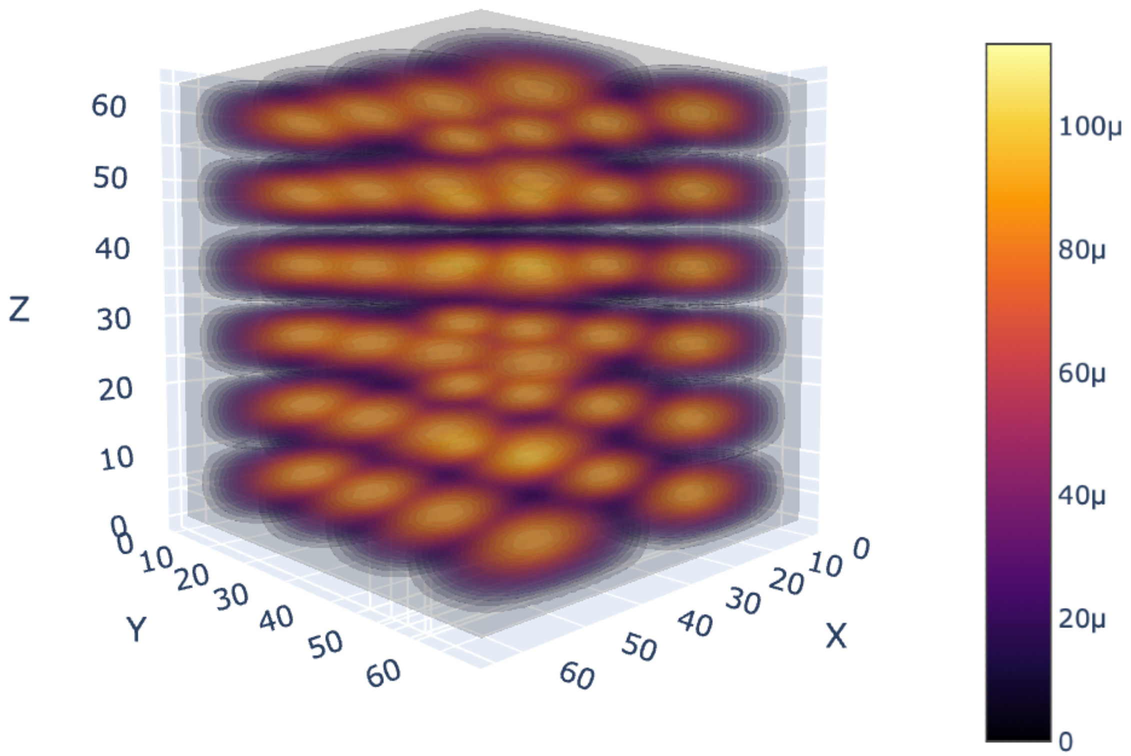


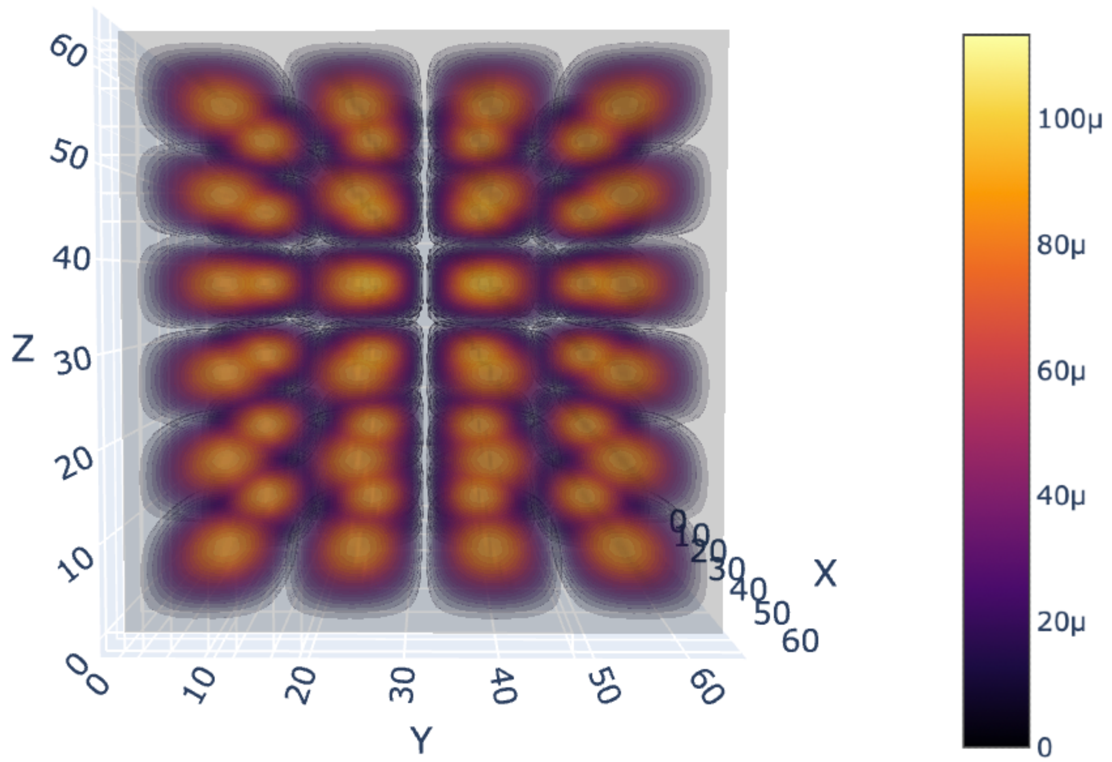
Визуализация сетки полученной численным решением:





Визуализация сетки погрешности:





Вывод

Задача для трехмерного гиперболического уравнения в области, представляющей из себя прямоугольный параллелепипед, подходит для распараллеливания с помощью технологии OpenMP, позволяя получить ускорение вплоть до 12 раз. Причем при большем размере сетки ниже погрешность, а распараллеливание дает немного лучшие результаты по сравнению с мелкими сетками.
Также было замечено, что последовательный код работает немного быстрее, чем код в один поток, это связано с накладными расходами на создание потоков и ожидание их завершения, а также с оптимизациями компилятора.