Московский государственный университет им. М.В. Ломоносова
Физический факультет

# Курсовая работа на тему:
# «Решение нелинейной задачи магнитостатики»

Студент:
206 группа Галкин Иван
Евгеньевич
Научный руководитель: проф. Перепёлкин Евгений Евгеньевич

# Оглавление

# 1. Введение

Одной из актуальных и вычислительно ёмких задач математического моделирования является проблема комплексной оптимизации динамики пучка в циклотроне. При моделировании динамики пучка в качестве входных данных используются электромагнитные поля.

Цель данной работы - вычисление магнитного поля в области дипольного магнита с использованием подхода массивно - параллельных вычислений на GPU и x86 архитектуры (CPU). А также анализ эффективности использованных методов и их сравнение.

Численное решение поставленной дифференциальной задачи магнитостатики было реализовано на языке C++ как на CPU, так и на GPU с использованием технологии CUDA [**1**]. Также был использован язык Python для визуализации совершенных вычислений [**3**].

## 2. Математическая модель

В зависимости от геометрии области, в которой решается краевая задача для поиска, в данном случае, магнитного поля, используется метод конечных элементов или метод конечных разностей. Оба метода могут приводить к решению системы алгебраических уравнений итерационным методом.

Рассмотрим решение задачи магнитостатики на базе подхода массивно-параллельных вычислений на GPU. Рассмотрим дипольный магнит в форме «оконная рама», изображенный на рис.1.

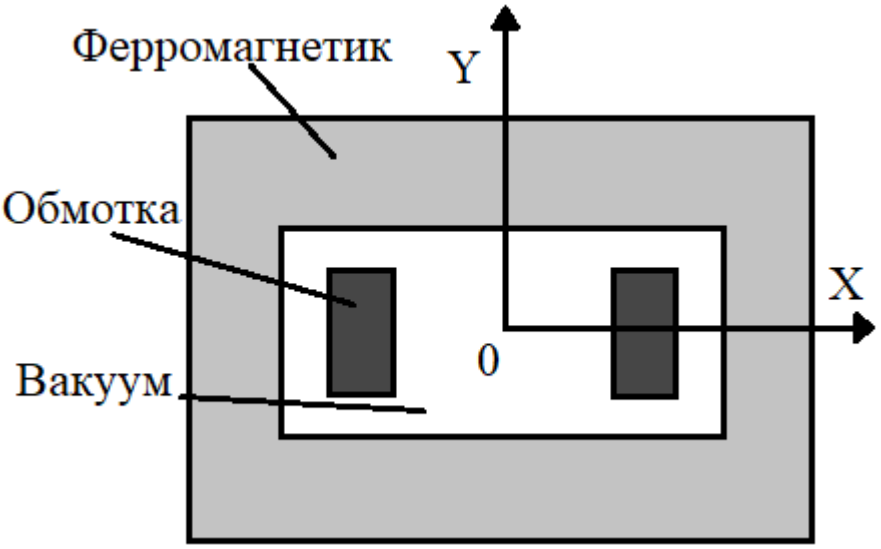
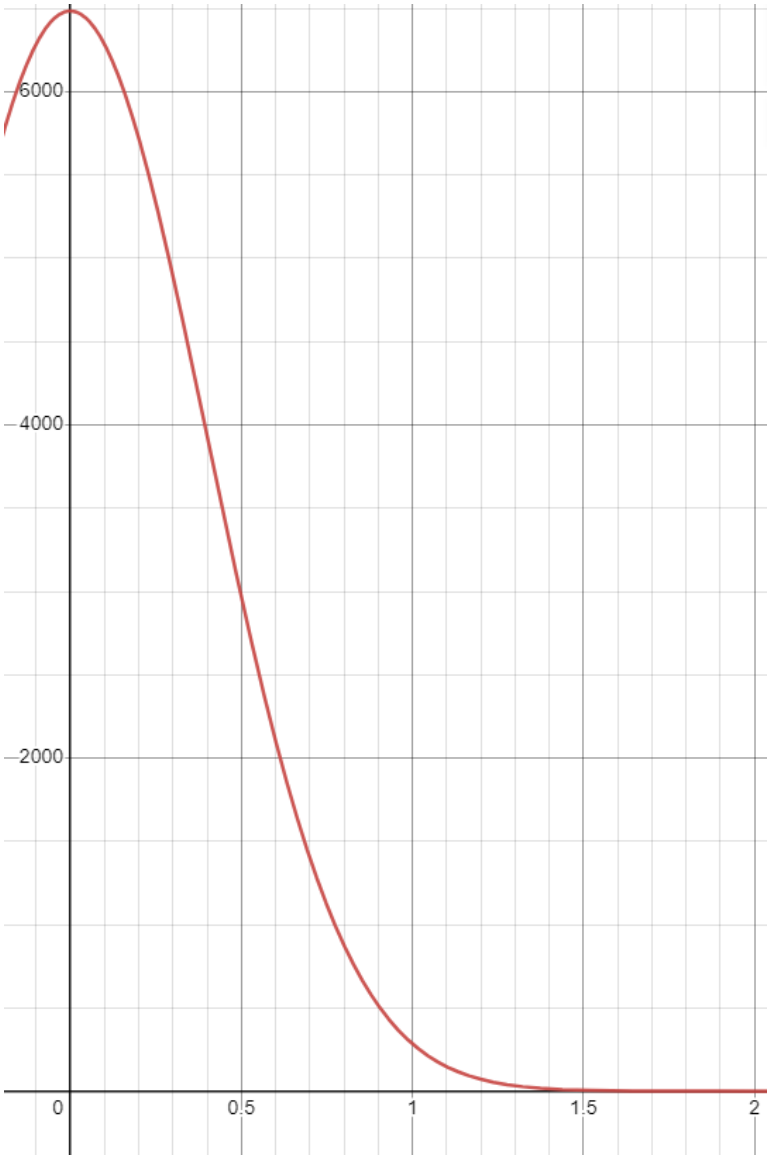



рис. 1

Дифференциальная постановка задачи магнитостатики относительно векторного потенциала $\vec{A} = \{0,\ 0,\ U\}$ имеет вид (1). Магнитная индукция $\vec{B} = rot\vec{A}$; $\vec{J}$ - плотность тока внешних источников; граница Г - соответствует границе раздела сред: ферромагнетик/вакуум.

$$\begin{cases} \dfrac{\partial}{\partial x}\left(\dfrac{1}{\mu_0\mu}\dfrac{\partial U}{\partial x}\right)+\dfrac{\partial}{\partial y}\left(\dfrac{1}{\mu_0\mu}\dfrac{\partial U}{\partial y}\right)=-J_z, \\ U|_{\Gamma_-}=U|_{\Gamma_+}, \\ \dfrac{1}{\mu}\dfrac{\partial U}{\partial n}\Big|_{\Gamma_-}=\dfrac{\partial U}{\partial n}\Big|_{\Gamma+}, \\ \lim\limits_{p\to\infty}|U(p)|\to 0. \end{cases} \quad (1)$$

Магнитные свойства ферромагнетика определяются функцией магнитной проницаемости $\mu(B)$. В данной задаче в качестве такой зависимости взята функция, пропорциональная функции Гаусса:




$$f(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2},$$

где $\mu=0,\ \sigma=0.4$

График $\mu(B)$ приведен на рис.2.

рис.2

Численное решение постановки (1) делается двухступенчатым методом релаксации:

$$U_{i,j}^{k+\frac{1}{2}} = \frac{\alpha_1 U_{i+1,j}^{k} + \alpha_2 U_{i,j+1}^{k} + \alpha_3 U_{i-1,j}^{k+1} + \alpha_4 U_{i,j-1}^{k+1} + 2F_{i,j}}{\left(\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4\right)},$$

$$U_{i,j}^{k+1} = (1-\omega)\, U_{i,j}^{k} + \omega U_{i,j}^{k+\frac{1}{2}}, \quad \nu_j^{k+1} = (1-\eta)\, \nu_j^{k} + \eta\, \nu_j^{k+\frac{1}{2}}, \tag{2}$$

$$\nu_j = \frac{1}{\mu_j},\ j = 1,\ 2,\ 3,\ 4,$$

где $\omega$, $\eta$ - параметры релаксации. Величина $F$ является интегралом от плотности тока $J$ по площади ячейки сетки:

$$F_{i,j} = -\frac{\mu_0}{4}\left(j_1 h_{i+1}^{x} h_j^{y} + j_2 h_{i+1}^{x} h_{j+1}^{y} + j_3 h_i^{x} h_{j+1}^{y} + j_4 h_i^{x} h_j^{y}\right) \tag{3}$$

Коэффициенты $\alpha$ имеют вид:

$$\alpha_1 = \frac{1}{h_{i+1}^{x}}\left(\frac{h_j^{y}}{\mu_1} + \frac{h_{j+1}^{y}}{\mu_2}\right),\ \alpha_2 = \frac{1}{h_{j+1}^{y}}\left(\frac{h_i^{x}}{\mu_3} + \frac{h_{i+1}^{x}}{\mu_2}\right)$$
$$\alpha_3 = \frac{1}{h_i^{x}}\left(\frac{h_j^{y}}{\mu_4} + \frac{h_{j+1}^{y}}{\mu_3}\right),\ \alpha_4 = \frac{1}{h_j^{y}}\left(\frac{h_i^{x}}{\mu_4} + \frac{h_{i+1}^{x}}{\mu_1}\right) \tag{4}$$

где $\mu_i$, $i = 1,\ 2,\ 3,\ 4$ - значения магнитной проницаемости в углах ячейки, вычисляемые по формуле аналогичной (6).

Значение на границе Г раздела сред находятся из (5):

$$U_{i,j} = \frac{\mu_4 h_i^{x} U_{i+1,j} + h_{i+1}^{x} U_{i-1,j}}{\mu_4 h_i^{x} + h_{i+1}^{x}}, \tag{5}$$

$$\mu_4 =$$

$$= \mu \left( \sqrt{ \left( \frac{U_{i,j} + U_{i,j-1} - U_{i-1,j} - U_{i-1,j-1}}{2h_i^x} \right)^2 + \left( \frac{U_{i-1,j} + U_{i,j} - U_{i-1,j-1} - U_{i,j-1}}{2h_j^y} \right)^2 } \right). \quad (6)$$

С точки зрения массивно-параллельных вычислений на GPU удобно в выражении (2) использовать “метод простой итерации”:

$$U_{i,j}^{k+1} = \frac{\alpha_1 U_{i+1,j}^k + \alpha_2 U_{i,j+1}^k + \alpha_3 U_{i-1,j}^k + \alpha_4 U_{i,j-1}^k + 2F_{i,j}}{\left( \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 \right)}. \quad (7)$$

То есть значения функции на следующей итерации в каждой точке пересчитывается только через значения с предыдущего шага.

В случае (7) в каждой точке функция может считаться независимо, то есть параллельно. Конечно, сходимость процесса (7) будет медленнее, чем (2), что потребует дополнительного числа итераций в случае (7).

.

# 3. Практическая часть

## 3.1 Вычисление на CPU

Был реализован двухступенчатый метод релаксации (2) для вычисления на архитектуре x86. Структура основного цикла программы выглядит следующим образом:

```
while ([условие необходимой точности])
{
   Relax(U0, U1, alph1, alph2, alph3, alph4, N, J, h, w); // вызов вычисляющей функции
   // [копирование U1 в U0]
   //вызов функции пересчета коэффициентов
   Alph(U0, alph1, alph2, alph3, alph4, N, h);
}
```

На вход функция Relax получает массив данных распределения потенциала на сетке с предыдущей итерации U0, набор коэффициентов, размер сетки N, массив распределения плотности тока J, ширину шага по координате h (постоянна на всей сетке), а также параметр релаксации w. В процессе исполнения функция вычисляет массив данных для следующей итерации U1.

Далее, как того требует алгоритм, массив последующей вычисленной итерации копируется, и используется как начальный для следующей итерации цикла. После этого вызывается функция для пересчета коэффициентов Alpha.

Функция Alpha принимает на вход вычисленный массив распределения потенциала для данной итерации, массивы коэффициентов (для записи), а также размер сетки N и шаг по координате h. Пересчет коэффициентов сводится к вычислению относительной магнитной проницаемости в области ферромагнетика по формулам аналогичным (6), с последующим вычислением коэффициентов по формулам (4).

Цикл не завершается, пока разность между вычисленными итерациями не станет ниже определенного значения точности. Задача (1) решается в ¼ области с учетом симметрии магнита.

Для начала, программа была проверена на корректность: был вычислен потенциал для случая отсутствия ферромагнетика и внешней среды (линейная задача), а также при расположении обмотки в центре сетки и размере сетки 64x64. График вычисленного потенциала показан на рис. 3.

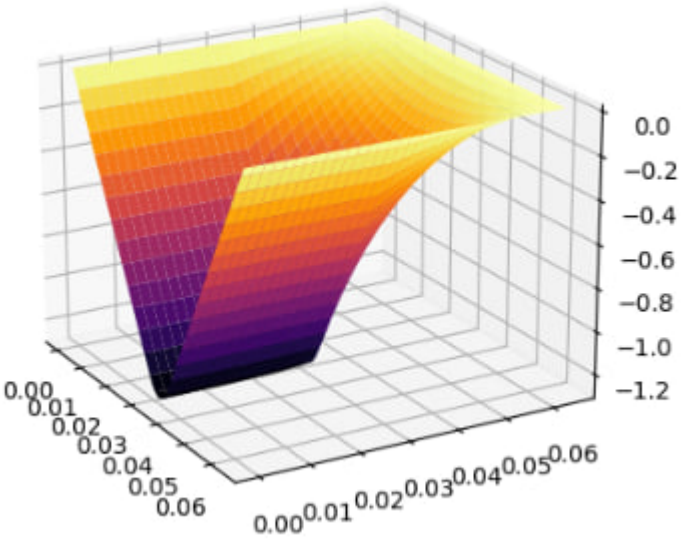



рис. 3 Результат решения линейной задачи магнитостатики

Далее был рассчитан потенциал в случае наличия ферромагнетика и внешней среды (нелинейная задача), обмотка находилась на трети длины координатной оси (рис. 4):

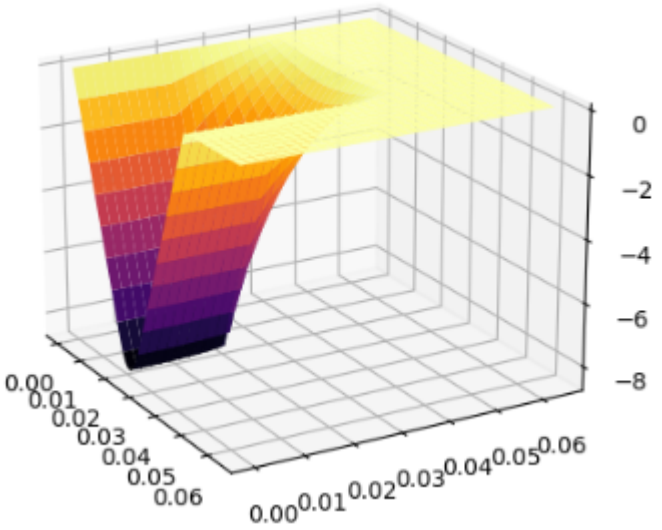



рис.4

Также была исследована скорость сходимости процесса вычисления потенциала от параметра релаксации $\omega$ на сетке 64x64 (рис. 5):

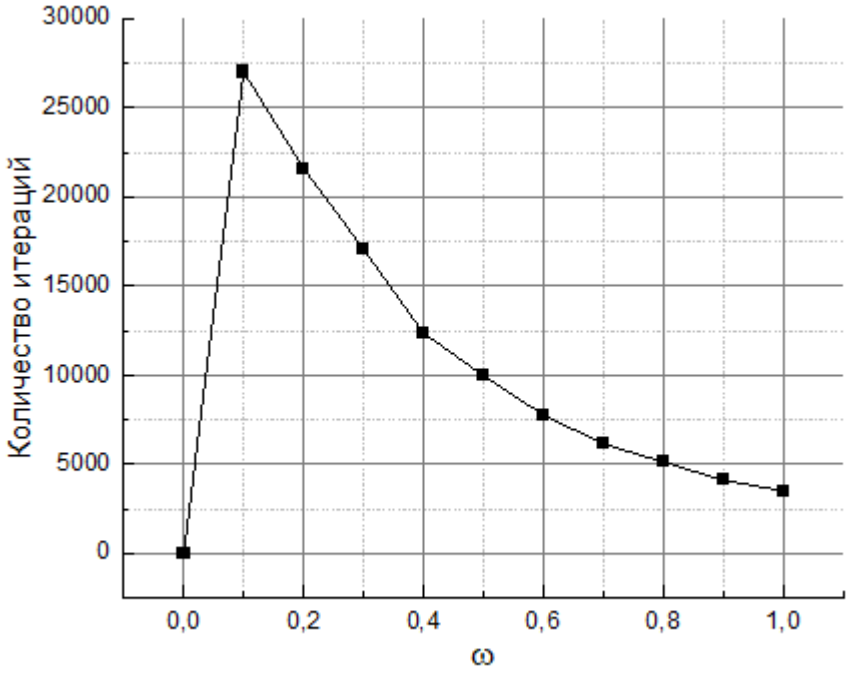



рис.5

Как видно из рис. 5 при нулевом параметре процесса вычисления потенциала как такого не происходило, т.к. каждая следующая итерация была идентична первому приближению (см. формулы (2)). Наиболее выгодным по количеству итераций, а значит и по времени, оказался параметр релаксации, равный единице: для достижения необходимой точности вычисления программе понадобилось примерно 3500 итераций. В дальнейшем, при решении задачи магнитостатики методом релаксации (на CPU), данный параметр берется равным единице.

Также была исследована примерная (т.к.точность чисел с плавающей точкой довольно высока для больших N, а значит “ловить” изменения вычисленного потенциала в двух соседних итерациях с каждой итерацией все труднее) зависимость времени выполнения программы от размера сетки N (рис. 6):

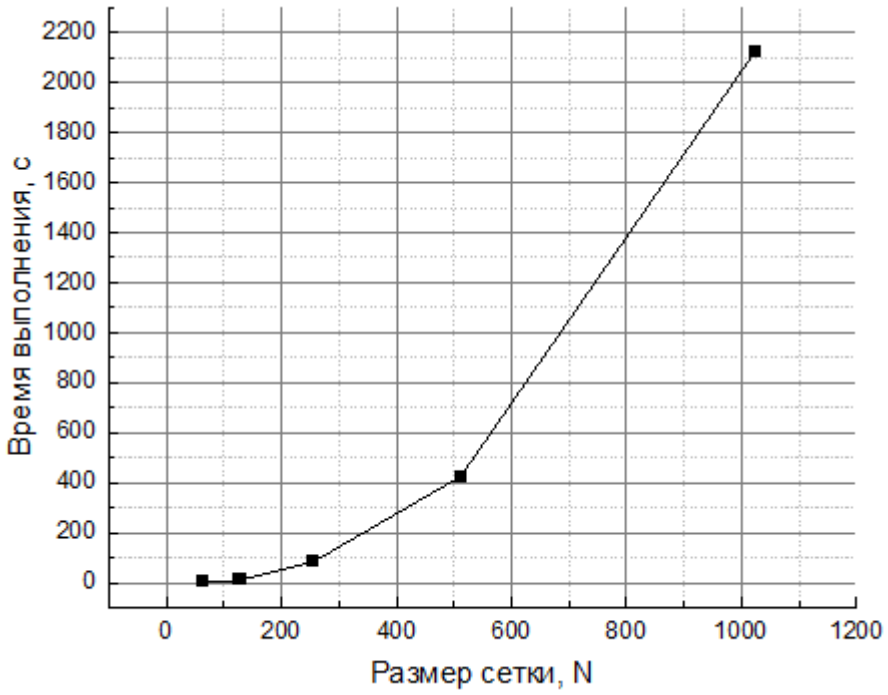



рис. 6

Для вычисления сетки 64x64 и 128x128 понадобилось несколько секунд, 512x512 - примерно 7 минут, для 1024x1024 - около 30 минут. На рис. 7 показан результат выполнения программы для сетки 512x512:

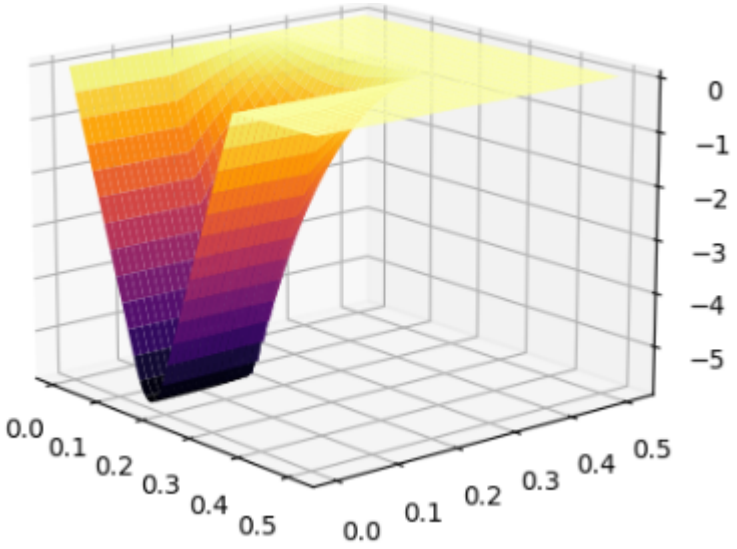



рис. 7

### 3.2 Вычисление на GPU

Для использования метода массивно-параллельных вычислений использовалась программно-аппаратная архитектура CUDA компании NVIDIA [2]. Были написаны две “функции-ядра” - функции, выполняющиеся параллельно каждым вычислительным потоком (thread), который с остальными потоками объединен в блоки, выполняющиеся физически целиком на одном потоковом мультипроцессоре (SM) GPU.

Рассмотрим “функцию-ядро” Relax для параллельного вычисления по формуле (7). На вход функция получает массив данных с предыдущей итерации, набор коэффициентов, а также параметры сетки и массив плотности тока, на выходе получается приближение для новой итерации по формуле (7).

```
__global__ void Relax (float* U0, float* U1,
float* Alph1, float* Alph2, float* Alph3, float*
Alph4,int n, float *J, float h)
{// выбор узла (i, j) где вычисляется функция
   int ind = blockDim.x * blockIdx.x +
threadIdx.x;// ind = N*j + i
// нахождение приближения новой итерации по
// формуле (7)
   U1[ind]=[необходимые вычисления с U0 и Alph1,
2,3,4] }
```

На “host” “функция-ядро” Relax запускается на каждой итерации до достижения необходимой точности сходимости итерационного процесса. Далее идет пересчет коэффициентов, который тоже производится параллельно для каждого узла сетки “функцией-ядром” Alph. Структура функции Alph аналогична функции Relax. На вход функция Alph получает массив $U^k_{i,j}$. Каждая нить на GPU вычисляет в своем узле значение функций $\mu_i,\ i=1,\ 2,\ 3,\ 4$ по формулам, аналогичным (6). На выходе получаются четыре массива коэффициентов Alph1, Alph2, Alph3, Alph4.

На рис. 8 показан результат решения дифференциальной задачи (1). С учетом симметрии геометрии магнита, задача решалась в ¼ области, на сетке 1024x1024.

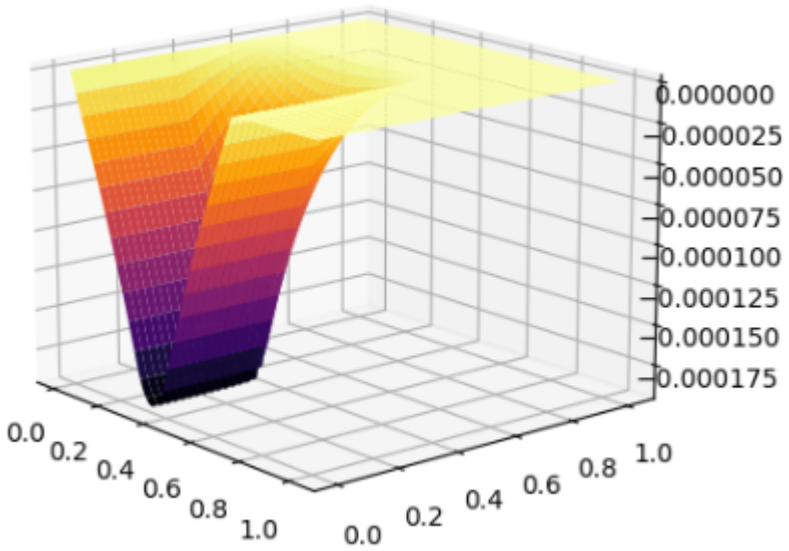



рис. 8

Также исследована зависимость времени выполнения от размера сетки N (рис. 9).

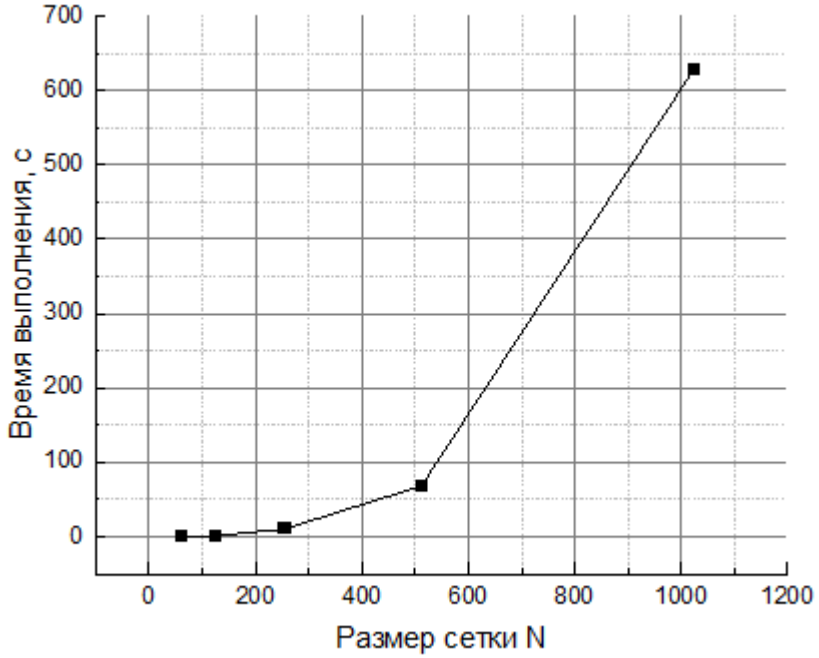



рис. 9

Значения для каждого N в несколько раз меньше, чем для соответствующих размеров в случае вычисления исключительно на CPU (см. рис. 6), что говорит о высокой эффективности метода вычисления векторного потенциала с использованием GPU.

# 4. Основные итоги

Реализованы методы вычисления проекции векторного потенциала в дипольном магните: с использованием архитектуры x86, а также с помощью технологии CUDA и GPU. Для случая вычисления на CPU найден оптимальный параметр релаксации, для обоих случаев найдены времена выполнения программы для различных размеров сетки. Показано, что использование технологии CUDA позволяет существенно ускорить процесс вычисления потенциала магнитного поля.

# Источники информации

## Литература

[1] Иноземцева Н.Г., Перепёлкин Е.Е., Садовников Б.И. Оптимизация алгоритмов задач математической физики для графических процессоров. М.:Физический факультет МГУ, 2012. - 255 с.ISBN 978-5-8279-0107-5

## Ссылки

[2] https://docs.nvidia.com/cuda/

[3] https://matplotlib.org/stable/api/index.html