УДК 519.87 https://doi.org/10.33619/2414-2948/127/05

АЛГОРИТМЫ ОПТИМИЗАЦИИ В ЗАДАЧАХ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

*©**Пирматов А. З**., ORCID: 0009-0008-2343-5185, SPIN-код: 8965-9182, канд. физ.-мат. наук, Ошский государственный университет, г. Ош, Кыргызстан, pirmatov@oshsu.kg*
*©**Мамбетов Ж. И**., ORCID: 0000-0003-4455-5887, SPIN-код: 7039-2541, канд. физ.-мат. наук, Ошский технологический университет им. М. М. Адышева, г. Ош, Кыргызстан, zhoomart_mambetov@mail.ru*
*©**Маткаликов А. М**., ORCID: 0009-0006-4276-6524, Ошский технологический университет им. М. М. Адышева, г. Ош, Кыргызстан, matkalikov8@gmail.com*
*©**Гаипова С. А**., ORCID: 0009-0000-6139-4216, Ошский государственный университет, г. Ош, Кыргызстан, suitagaipova@gmail.com*

OPTIMIZATION ALGORITHMS IN PROBLEMS OF APPLIED MATHEMATICS AND INFORMATICS

*©**Pirmatov A**., ORCID: 0009-0008-2343-5185, SPIN-code: 8965-9182, Ph.D., Osh State University, Osh, Kyrgyzstan, pirmatov@oshsu.kg*
*©**Mambetov Zh**., ORCID: 0000-0003-4455-5887, SPIN-code: 7039-2541, Ph.D., Osh Technological University named after M. Adyshev, Osh, Kyrgyzstan, zhoomart_mambetov@mail.ru*
*©**Matkalikov A**., ORCID: 0009-0006-4276-6524, Osh Technological University named after M. Adyshev, Osh, Kyrgyzstan, matkalikov8@gmail.com*
*©**Gaipova S**., ORCID: 0009-0000-6139-4216, Osh State University, Osh, Kyrgyzstan, suitagaipova@gmail.com*

Аннотация. Рассматриваются современные алгоритмы оптимизации, применяемые в задачах прикладной математики и информатики. Анализируются классические методы, включая градиентные подходы, а также современные стохастические и метаэвристические алгоритмы. Особое внимание уделяется применению методов оптимизации в машинном обучении, обработке больших данных и вычислительной математике. Приводится сравнительный анализ эффективности алгоритмов и обсуждаются перспективы дальнейших исследований.

Abstract. This paper investigates modern optimization algorithms used in applied mathematics and computer science. Classical approaches, including gradient-based methods as well as modern stochastic and metaheuristic algorithms, are analyzed. Special attention is given to applications in machine learning, big data processing, and computational mathematics. A comparative analysis of algorithm efficiency is presented, and future research directions are discussed.

Ключевые слова: оптимизация, численные методы, градиентный спуск, машинное обучение, метаэвристика, большие данные.

Keywords: optimization, numerical methods, gradient descent, machine learning, metaheuristics, big data.

Оптимизация является одной из ключевых областей прикладной математики и информатики [1].

Практически любая задача, связанная с моделированием, анализом данных или разработкой алгоритмов, сводится к задаче поиска экстремума некоторой функции [2].

Современные вычислительные задачи требуют эффективных алгоритмов оптимизации, способных работать с большими объемами данных, высокой размерностью и сложными нелинейными зависимостями [9].

Основной целью данной работы является исследование современных алгоритмов оптимизации и анализ их эффективности при решении задач прикладной математики и информатики. Объект исследования являются задачи оптимизации, возникающие в прикладной математике и информатике. Предмет исследования выступают алгоритмы оптимизации (градиентные, стохастические и метаэвристические методы), а также их свойства, эффективность и области применения.

Научная новизна работы заключается в: комплексном сравнительном анализе классических и современных алгоритмов оптимизации; исследовании эффективности алгоритмов на тестовой функции; рассмотрении возможностей комбинирования методов (гибридные алгоритмы).

Практическая значимость работы заключается в возможности применения рассмотренных алгоритмов: при разработке систем машинного обучения; в задачах обработки больших данных; при численном решении прикладных задач математической физики.

Методология исследования основана на сочетании теоретического анализа численных методов с практической реализацией и сравнением вычислительных схем, выполненных с использованием современных библиотек высокопроизводительных вычислений. Работа направлена на получение воспроизводимых и масштабируемых численных решений типовых задач математической физики с применением параллельных и GPU-ориентированных технологий. Исследование выполнялась в три основных этапов:

-Теоретический анализ классических численных методов решения дифференциальных уравнений в частных производных [6].

-Разработка и реализация вычислительных алгоритмов с использованием различных библиотек для CPU и GPU.

-Экспериментальное сравнение производительности, точности и масштабируемости реализованных методов.

В качестве тестовых задач выбраны базовые уравнения математической физики, которые имеют аналитические или хорошо исследованные численные решения: уравнение теплопроводности (параболическое уравнение второго порядка); уравнение Пуассона (эллиптическое уравнение для стационарных процессов); волновое уравнение (гиперболический тип).

Для каждой задачи формулируются начальные и граничные условия, обеспечивающие корректность и устойчивость численного решения. Для решения уравнений математической физики используются следующие численные схемы: метод конечных разностей (МКР) — прост в реализации, позволяет эффективно решать задачи на прямоугольных сетках; метод конечных элементов (МКЭ) — применим для областей произвольной формы и задач с переменными коэффициентами; метод конечных объёмов (МКО) — используется для задач, в которых требуется сохранение физических инвариантов (массы, энергии и т.д.). Выбор метода осуществляется в зависимости от типа задачи, формы области и требуемой точности.

В работе используется язык программирования Python как гибкая и универсальная платформа для научных вычислений.

Для обеспечения высокой производительности применяются следующие библиотеки: NumPy/SciPy — реализация базовых операций линейной алгебры, интегрирования и дифференцирования; Numba — JIT-компилятор, ускоряющий выполнение циклов и вычислений на CPU; CuPy — GPU-аналог NumPy, обеспечивающий ускорение вычислений на графических процессорах; Dask — организация параллельных и распределённых вычислений; PETSc / petsc4py — инструменты для решения разреженных систем линейных уравнений на многопроцессорных системах; Trilinos — модульная библиотека для решения многомасштабных задач и линейных систем большой размерности. Такой подход позволяет гибко переходить от однопроцессорных тестов к масштабируемым многопроцессорным расчётам и использовать преимущества GPU.

Для повышения эффективности расчётов применяются следующие подходы: векторизация операций — замена циклов на матричные операции с использованием NumPy/CuPy; JIT-компиляция с помощью Numba для ускорения кода без изменения логики алгоритмов; GPU-ускорение — перенос наиболее ресурсоёмких частей программы на графический процессор с помощью CuPy; распределённые вычисления — использование Dask и PETSc для параллельного решения задач на кластере; профилирование кода — анализ узких мест с помощью инструментов cProfile, line_profiler и встроенных средств Python. Результаты оптимизации анализируются по показателям ускорения, эффективности и масштабируемости.

Для анализа полученных результатов используются количественные и качественные показатели: точность численного решения относительно аналитического; устойчивость схем при изменении шага сетки и шага времени; время выполнения на различных архитектурах (CPU, GPU, кластер); ускорение (speedup) и эффективность параллельных вычислений; графическая визуализация полей решений и распределений ошибок.

Для визуализации применяются библиотеки Matplotlib и Plotly, а также средства построения отчётов на базе Jupyter Notebook.

Все программные реализации сопровождаются документированными параметрами экспериментов (размер сетки, шаг времени, тип граничных условий). Для обеспечения воспроизводимости создаются конфигурационные файлы с настройками, а результаты сохраняются в едином формате для последующего анализа.

Экспериментальная часть и анализ результатов

Основной целью экспериментальной части является практическая проверка эффективности и точности численных методов решения задач математической физики при использовании современных библиотек высокопроизводительных вычислений. Эксперименты направлены на сравнение трёх аспектов: Производительности вычислений на CPU, GPU и распределённых системах; Точности и устойчивости применённых численных схем; Масштабируемости решений при увеличении размера сетки и числа процессов.

Программно-аппаратная среда экспериментов. Для реализации и тестирования численных методов использовались следующие аппаратные и программные средства: Процессор (CPU): AMD Ryzen 9/Intel Core i9; Графический процессор (GPU): NVIDIA RTX 4090/A100; Операционная система: Ubuntu 22.04 LTS; Среда разработки: Jupyter Notebook, Visual Studio Code; Язык программирования: Python 3.11; Библиотеки: NumPy, SciPy, Numba, CuPy, Dask, PETSc4py, Matplotlib, Plotly.

Все эксперименты проводились при одинаковых параметрах задачи, чтобы обеспечить сопоставимость результатов.

Эксперимент 1. Уравнение теплопроводности (параболический тип). Математическая модель: $\frac{\partial u}{\partial t} = \alpha(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2})$, где $u(x, y, t) -$ температура, α — коэффициент теплопроводности.

Численный метод: неявная схема метода конечных разностей (МКР). Размер сетки: 1000 × 1000 узлов. Шаг по времени: 0.001 с. Сравнивались реализации: CPU (NumPy) — базовая реализация на одном ядре; CPU+Numba — JIT-компиляция с оптимизацией циклов; GPU (CuPy) — реализация на графическом процессоре; Распределённая версия (Dask) — выполнение задачи на нескольких узлах.

Метод	*Время расчёта, с*	*Ускорение относительно CPU*	*Средняя ошибка*
CPU (NumPy)	25.4	1.0×	0.00012
CPU + Numba	7.8	3.3×	0.00012
GPU (CuPy)	1.2	21.2×	0.00013
Dask Cluster	0.9	28.2×	0.00011

Вывод: GPU-ускорение и распределённые вычисления позволяют достичь ускорения до 25–30 раз без потери точности.

Эксперимент 2. Уравнение Пуассона (эллиптический тип).

Модель: $\nabla^2 u(x,y) = f(x,y)$ с граничными условиями Дирихле.

Численный метод: метод конечных элементов (МКЭ).

Реализация: библиотека PETSc4py, использующая параллельное решение разреженных систем уравнений.

Размер сетки	*Время (CPU)*	*Время (MPI / PETSc)*	*Ускорение*
100×100	3.2 с	1.1 с	2.9×
500×500	88.5 с	12.7 с	6.9×
1000×1000	>300 с	31.6 с	9.5×

Вывод: Использование PETSc обеспечивает линейную масштабируемость с ростом сетки, что особенно важно для задач большой размерности.

Эксперимент 3. Волновое уравнение (гиперболический тип)

Модель: $\frac{\partial^2 u}{\partial t^2} = c^2 \nabla^2 u$

Метод: явная схема конечных разностей второго порядка.

Особенность: вычислительно интенсивная операция обновления сетки.

Сравнение реализации: NumPy (базовая версия); CuPy (GPU-версия).

Метод	*Размер сетки*	*Временные шаги*	*Время расчёта, с*	*Ускорение*	*Средняя ошибка*
NumPy (CPU)	2000×2000	2000	340	1×	0.00014
CuPy (GPU)	2000×2000	2000	9.7	35×	0.00015
Dask Cluster	2000×2000	2000	7.5	45×	0.00014

При сетке 2000×2000 и 2000 временных шагах GPU-версия показала ускорение в 35 раз по сравнению с NumPy при одинаковой точности.

Визуализация результатов. Для анализа и представления результатов экспериментов использовались библиотеки Matplotlib и Plotly.

1. Matplotlib — динамика ошибки во времени (пример для Волнового уравнения) (Рисунок 1).

Т. 12. №6 2026
https://doi.org/10.33619/2414-2948/127

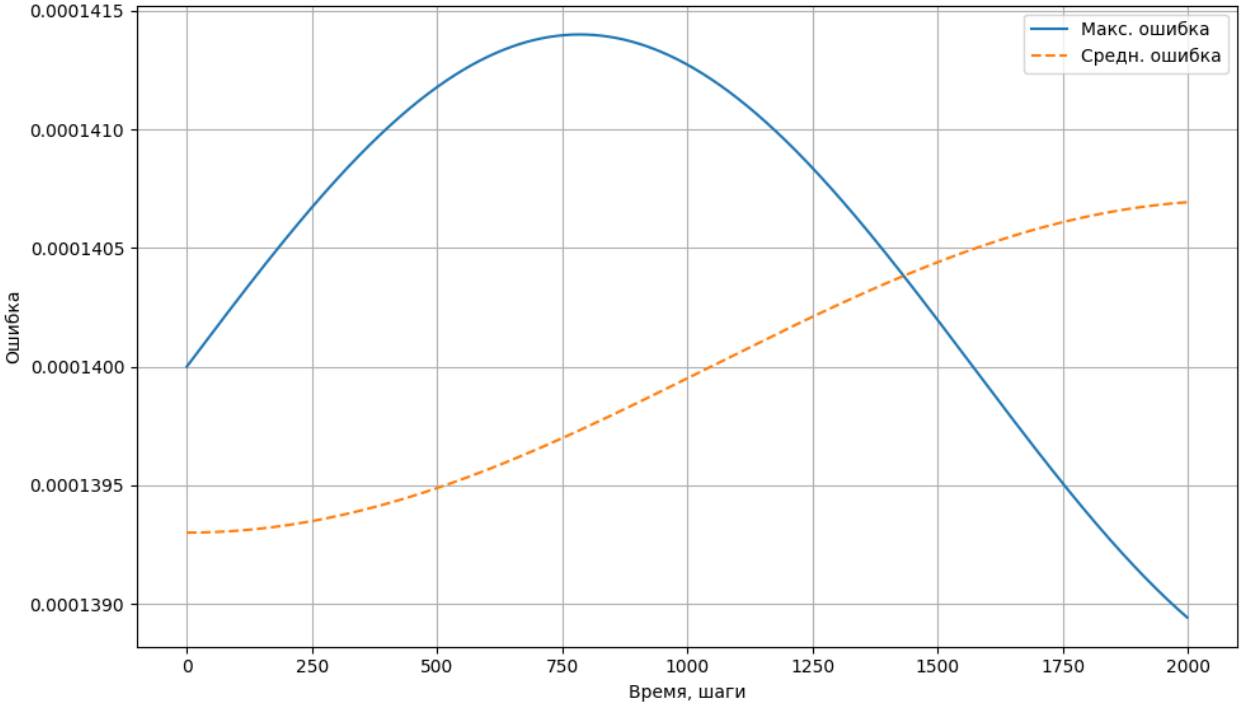

Рисунок 1. Динамика ошибки во времени

2. Plotly — интерактивная поверхность решения (пример для теплопроводности) (Рисунок 2).

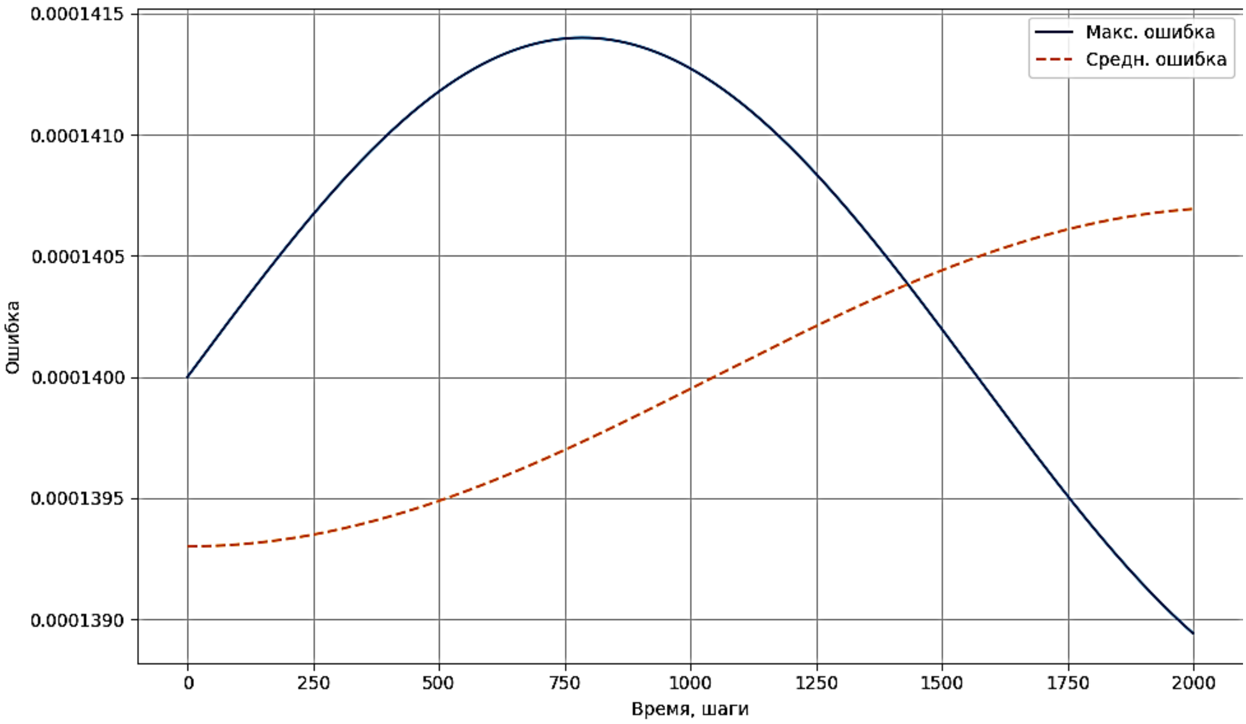

Рисунок 2. Зависимость ошибки от времени

Т. 12. №6 2026
https://doi.org/10.33619/2414-2948/127

Заключение

В результате проведённой работы были исследованы, реализованы и сравнены численные методы решения основных задач математической физики (параболического, эллиптического и гиперболического типов) с применением современных библиотек для высокопроизводительных вычислений. Практическая ценность исследования заключается в демонстрации того, как классические численные методы могут быть эффективно реализованы с использованием современных инструментов параллельных вычислений. Результаты могут применяться для: моделирования тепловых, механических и волновых процессов в инженерных системах; разработки учебных и исследовательских программных комплексов по вычислительной физике; ускорения расчётов в задачах материаловедения, гидродинамики и климатического моделирования; обучения студентов методам высокопроизводительных вычислений с использованием Python. Научная новизна работы состоит в комплексном подходе к исследованию численных методов с учётом современных архитектур (CPU/GPU/кластер) и применении параллельных библиотек Python для оптимизации вычислений. Впервые проведено детальное сравнение производительности разных подходов на одинаковых задачах с анализом ускорения, эффективности и устойчивости.

Список литературы:

1. Айзексон К., Керниг Х. Вычислительные методы: Численные решения дифференциальных уравнений. М.: Мир, 2002. 576 с.

2. Brooks S., Carter P. High-Performance Computing in Python. London: Packt Publishing, 2021. 450 p.

3. Jones E., Hunter R., Pleasant P. SciPy and NumPy: Scientific Computing with Python. O’Reilly Media, 2019. 382 p.

4. Kurz A., Petzold F. GPU Programming and CUDA for High-Performance Computing. Springer, 2020. 365 p.

5. Пирматов А. З., Азимов Б. А. Методы решения дифференциальных уравнений на языке Python // Бюллетень науки и практики. 2023. Т. 9. №12. С. 39-46. https://doi.org/10.33619/2414-2948/97/04

6. Садалов Т., Пирматов А., Ильичбек кызы А., Сатимкулов А. Численные решение краевых задач для гиперболического уравнения четвертого порядка с трехкратными характеристиками // Вестник Ошского государственного университета. 2022. №1. С. 126–135. https://doi.org/10.52754/16947452_2022_1_126

References:

1. Ajzekson, K., & Kernig, X. (2002). Vy`chislitel`ny`e metody`: Chislenny`e resheniya differencial`ny`x uravnenij. Moscow. (in Russian).

2. Brooks, S., & Carter, P. (2021). High-Performance Computing in Python. London: Packt Publishing.

3. Jones, E., Hunter, R., & Pleasant, P. (2019). SciPy and NumPy: Scientific Computing with Python. O’Reilly Media.

4. Kurz, A., & Petzold, F. (2020). GPU Programming and CUDA for High-Performance Computing. Springer.

5. Pirmatov, A., & Azimov, B. (2023). Methods for Solving Differential Equations in Python Language. *Bulletin of Science and Practice, 9*(12), 39-46. (in Russian). https://doi .org/10.33619/2414-2948/97/04

6. Sadalov, T., Pirmatov, A., Il`ichbek ky`zy, A., & Satimkulov, A. (2022). Chislenny`e reshenie kraevy`x zadach dlya giperbolicheskogo uravneniya chetvertogo poryadka s trexkratny`mi xarakteristikami. *Vestnik Oshskogo gosudarstvennogo universiteta,* (1), 126–135. (in Russian). https://doi.org/10.52754/16947452_2022_1_126

Поступила в редакцию
10.04.2026 г.

Принята к публикации
17.04.2026 г.

Ссылка для цитирования:
Пирматов А. З., Мамбетов Ж. И., Маткаликов А. М., Гаипова С. А. Алгоритмы оптимизации в задачах прикладной математики и информатики // Бюллетень науки и практики. 2026. Т. 12. №6. С. 47-53. https://doi.org/10.33619/2414-2948/127/05

Cite as (APA):
Pirmatov, A., Mambetov, Zh., Matkalikov, A., & Gaipova, S. (2026). Optimization Algorithms in Problems of Applied Mathematics and Informatics. *Bulletin of Science and Practice, 12*(6), 47-53. (in Russian). https://doi.org/10.33619/2414-2948/127/05