Реализация стабилизированного метода бисопряженных градиентов для блочно-разреженных матриц на платформе CUDA

Г.В. Муратова[1], Т.С. Мартынова[1], П.А. Оганесян[2,1]

И.Д. Игнатенко[1], В.В. Крылов[1].

[1]Южный федеральный университет, Ростов-на-Дону
[2]Университет МГУ-ППИ, г. Шэньчжэнь, Китай

Аннотация: В статье рассматривается решение систем линейных алгебраических уравнений, возникающих при решении статических задач и задач на установившиеся колебания методом конечных элементов. Представлен формат блочно-разреженных матриц на основе CSR (формата сжатых строк), его реализация для GPU с использованием CUDA. Реализован стабилизированный метод сопряженных градиентов и решены модельные задачи различной размерности, проведено сравнение с референсной реализацией на языке MATLAB.

Ключевые слова: разреженные матрицы, метод конечных элементов, блочные матрицы, GPU, параллельные вычисления, системы линейных алгебраических уравнений, метод бисопряженных градиентов

Введение

При моделировании конструкций методом конечных элементов возникают системы линейных алгебраических уравнений (СЛАУ) с большими разреженными матрицами. Структура матрицы зависит от типа решаемой задачи. В данной работе рассматриваются задачи упругости и электроупругости в трехмерной постановке с различными типами конечных элементов (КЭ). Рассмотрим задачу определения собственных частот колебаний электроупругого тела в постановке, используемой в работе [1].

$$\rho\omega^2\mathbf{u} - \nabla\cdot\boldsymbol{\sigma} = \mathbf{f};\ \nabla\cdot\mathbf{D} = 0;$$
$$\boldsymbol{\sigma} = \mathbf{c}^{\mathbf{E}}\cdot\boldsymbol{\varepsilon} - \mathbf{e}^{\mathbf{T}}\cdot\mathbf{E};\ \mathbf{D} = \mathbf{e}\cdot\boldsymbol{\varepsilon} + \boldsymbol{\kappa}^{\mathrm{S}}\cdot\mathbf{E}, \quad (1)$$
$$\boldsymbol{\varepsilon} = (\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}})/2;\ \mathbf{E} = -\nabla\cdot\boldsymbol{\varphi}$$

где ρ — плотность материала, ω — круговая (угловая) частота, $\mathbf{u}$ — поле перемещений, $\boldsymbol{\sigma}$ — тензор напряжений, $\mathbf{f}$ — внешние силы, $\mathbf{D}$ — вектор электрического смещения, $\mathbf{c}^{\mathbf{E}}$. — тензор упругих констант, $\boldsymbol{\varepsilon}$ — тензор деформаций, $\mathbf{e}$ — тензор пьезоэлектрических коэффициентов, $\mathbf{E}$ — вектор напряжённости электрического поля, $\boldsymbol{\kappa}^{\mathrm{S}}$. — тензор диэлектрической

проницаемости, $\boldsymbol{\varphi}$ — электрический потенциал. Решение такой задачи является одним из ключевых этапов при анализе пьезоустройств, таких как генераторы энергии, ультразвуковые медицинские приборы, различные сенсоры и датчики. Определив собственные частоту и соответствующие собственные формы колебаний, можно получить необходимую информацию о рабочих режимах рассматриваемого устройства. Интерес представляют и задачи теории упругости, описываемые моделью (2).

$$\begin{gathered}\rho\omega^2\mathbf{u} - \nabla \cdot \boldsymbol{\sigma} = \mathbf{f};\\ \boldsymbol{\sigma} = \mathbf{c}^{\mathbf{E}} \cdot \boldsymbol{\varepsilon};\ \mathbf{D} = \mathbf{e} \cdot \boldsymbol{\varepsilon},\\ \boldsymbol{\varepsilon} = (\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}})/2;\end{gathered} \tag{2}$$

При переходе от континуальной постановки задачи к КЭ-дискретизации исходная задача (1) заменяется на задачу (3):

$$\boldsymbol{Az} = \mu\boldsymbol{Mz},\ \ \boldsymbol{A}, \boldsymbol{M} \in R^{n\times n}, \tag{3}$$

где $\boldsymbol{A}$ соответствует матрице жесткости исходной задачи, $\boldsymbol{M}$ — матрице масс, $\mu = \omega^2$ является собственным значением, а z — собственным вектором, который соответствует объединённому вектору перемещений и электрического потенциала. Матрицы $\boldsymbol{A}$ и $\boldsymbol{M}$ будем называть глобальными матрицами. Они получаются в результате сборки локальных элементных матриц. Рассмотрим линейный пространственный элемент в форме тетраэдра с 4 степенями свободы в каждом узле: $[ux, uy, uz, \varphi]$ – первые три переменные описывают перемещения, переменная φ отвечает за электрический потенциал. Отметим, что рассуждения о структуре рассматриваемых матриц, переносятся и на ряд других связанных задач, таких как магнито-упругость и термоупругость.

Для выбранного элемента локальная матрица имеет размер 16x16 и в общем случае является заполненной вне зависимости от выбранного в конечном элементе метода интегрирования. Значения из этой матрицы

копируются в соответствующие ячейки глобальной матриц с учетом глобальной нумерации узлов, входящих в рассматриваемый элемент.

Таким образом, структура глобальной матрицы зависит от выбранной нумерации узлов в конечно-элементной сетке. В зависимости от выбранного формата хранения разреженных матриц целью перенумерации может являться сокращение ширины полосы (фронта) разреженной матрицы – то есть максимальное расстояние между ненулевыми элементами в каждой строке, либо формирование заполненных блоков. При решении задач на собственные значения в случае электроупругости важную роль играет и нумерация степеней свободы. Отметим, что с точки зрения удобства нумерации узлов конечно-элементные сетки можно разделить на регулярные, квазирегулярные и нерегулярные (рис. 1).

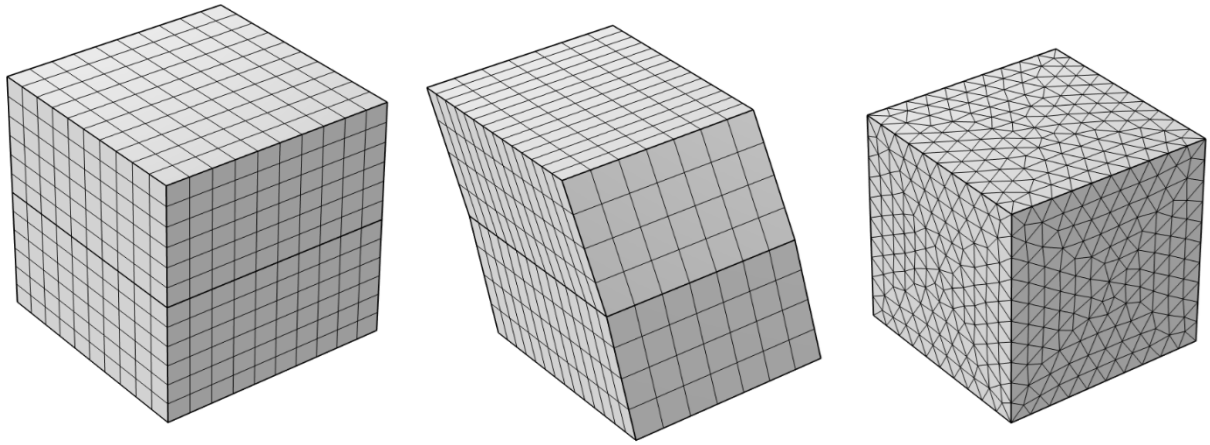

Рис. 1. Примеры регулярной, квазирегулярной и нерегулярной сеток.

Регулярные сетки наиболее удобны в работе, но возникают лишь в некоторых классах задач, например, при идентификации свойств композитных материалов [2]. Для таких сеток существует можно применять кэширование локальных матриц и эффективные схемы заполнения глобальных матриц. Квазирегулярные сетки в меньшей степени пригодны для кэширования локальных матриц, однако и для таких сеток покоординатная перенумерация дает хорошие результаты [3]. Эффективная перенумерация для нерегулярных матриц является NP-трудной задачей и

чаще строится за счет эвристик и заранее известной информации о геометрии сетки [4,5].

Работа с полученными разреженными матрицами чаще всего строится на выборе наиболее подходящего сжатого формата хранения и реализации для него основных операций: задания ненулевого элемента, доступа к произвольному элементу, строке, столбцу, умножения разреженной матрицы на плотный вектор, умножения матрицы на матрицу и других. Отметим, что не всегда необходимо реализовывать все операции, часто оказывается удобным выполнить операции, требующие частой записи в произвольные позиции, в одном формате, а затем перейти к операциям, требующим быстрого чтения, уже в другом формате [6,7].

Операции для блочно-разреженного формата хранения матриц

Программная реализация была выполнена на языке C++ (стандарт C++14) с использованием CUDA 12.5. Были разработан утилиты генерации тестовых данных с блочными матрицами на языке Ruby и вспомогательные скрипты на языке MATLAB, которые использовались для получения референсных оценок производительности и решений СЛАУ. Решение СЛАУ в пакете MATLAB производилось встроенными средствами с автоматическим выбором решателя, данные подавались в формате разреженных матриц. Также MATLAB использовался для визуализации внутренней структуры разреженных матриц.

Предложенная реализация предполагает использование заранее известного размера блока размера 3 или 4, что соответствует упругой и электроупругой задачам. Были разработаны два легковесных CUDA-ядра для эффективного умножения блочной матриц на плотный вектор в обычном и транспонированном представлении, с накоплением результата для последовательных умножений и без него. Реализована возможность передачи данных между памятью GPU и памятью машины-хоста. Исходные данные

поступают в форматах триплетов: собственном формате, разработанном для пакета ACELAN-COMPOS ранее, и в формате Matrix Market.

Реализация стабилизированного метода бисопряженных градиентов (BiCGStab) выполнена по классическому алгоритму [8]. Решение выполняется полностью на GPU. Основная операция – умножение разреженной матрицы на вектор – выполняется с использованием представленных выше CUDA-ядер. Операции сложения, скалярного произведения и нормировки плотных векторов выполняются средствами библиотеки cuBLAS. Алгоритм останавливается по критерию относительной невязки.

Был разработан консольный интерфейс для запуска программ, принимающий название метода решения СЛАУ, пути к файлу с матрицей, файлу с правой частью и выходному файлу с решением.

Тестовые данные

Для проведения тестирования разработанных программ использовались матрицы, полученные в пакете ACELAN-COMPOS для упругих и электроупругих задач, а также матрицы из известных датасетов: CYLSHELL[9], BFWAVE[10], SuiteSparse Matrix Collection [11]. Набор CYLSHELL содержит матрицы, возникающие при решении задач на собственные значения для цилиндрических оболочек. В этом наборе представлены в основном симметричные положительно определенные ленточные матрицы (рис. 2) с очень узким фронтом. Матрица s3rmt3m3 содержит элементы вне основной ленты. В наборе представлены матрицы двух видов: порядка 5000 строк и 90000 строк. Такие матрицы не являются специально адаптированными к блочно-разреженному формату, однако производительность разработанного программного обеспечения была оценена и для них.

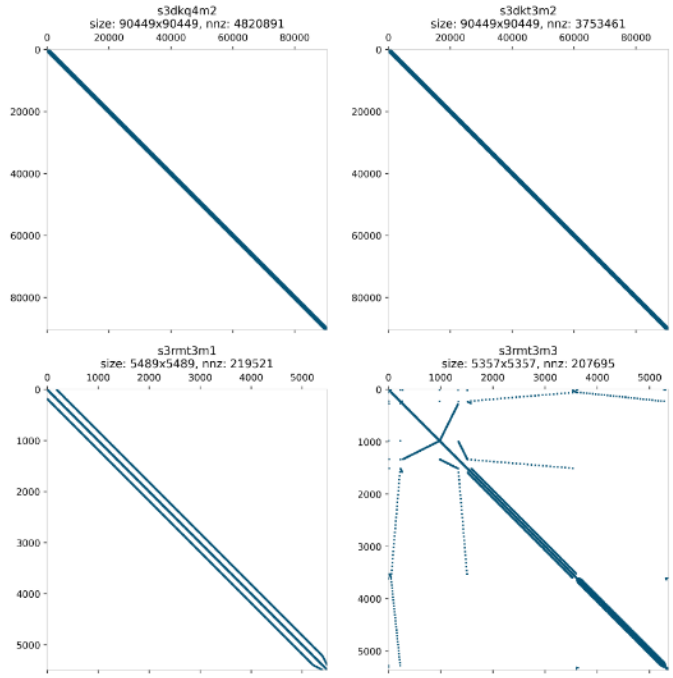

Рис. 2. Структура заполненности тестовых матриц из набора CYLSHELL.

Набор BFWAVE содержит матрицы размерами до 1000 строк, соответствующие задачам на собственные значения, полученные при решении уравнения Максвелла методом конечных элементов. Структура матриц близка с блочной на макроуровне: явно выделяются 2 крупных диагональных блока и симметричные блоки вне основной ленты.

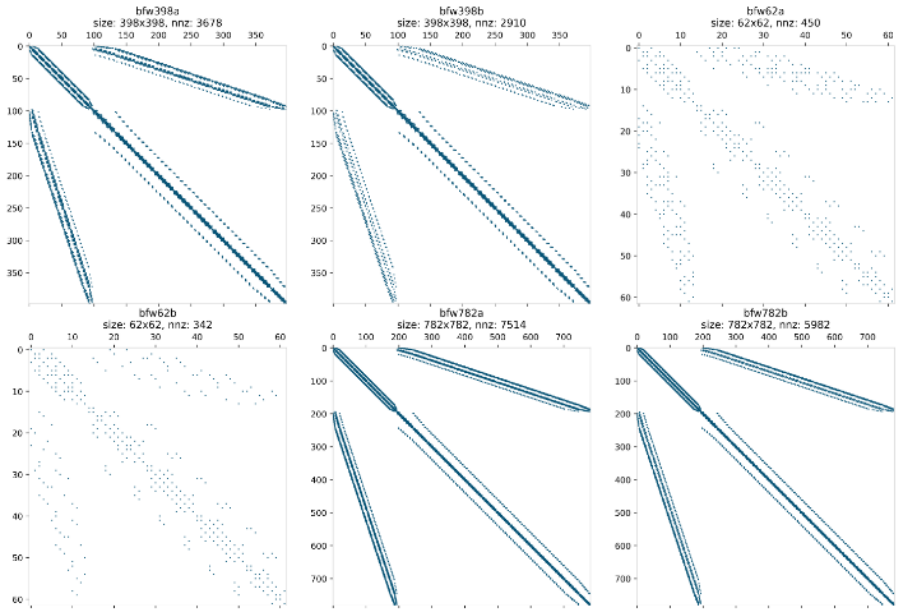

Рис. 3. Структура заполненности тестовых матриц из набора BFWAVE.

Также были выбраны различные матрицы из набора SuiteSparse Matrix Collection, соответствующие различным задачам и имеющие различную структуру, рис. 4. Названия матриц, для которых проводилось тестирование алгоритмов, представлены на рис 9 и 12.

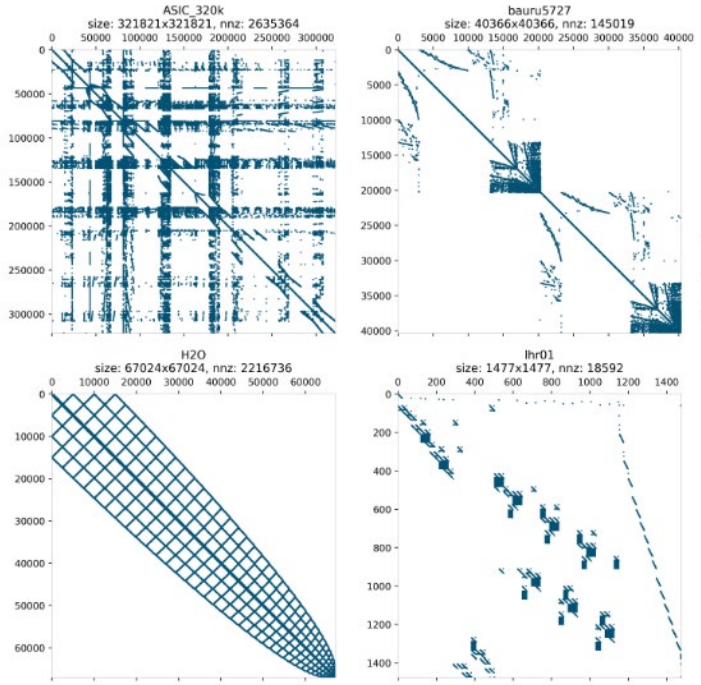

Рис 4. Структура заполненности тестовых матриц из набора SuiteSparse Matrix Collection.

Для всех перечисленных наборов данных проводились тестовые умножения матрицы на случайный вектор в разработанном ПО и в пакете MATLAB, умножения проводились по 100 раз и итоговый результат усреднялся.

Отдельный интерес представляют задачи, для которых разработанный формат предположительно наиболее эффективен. Были построены две модельные конечно-элементные задачи: пьезоупругая пластина зажата с двух торцов и нагружена разностью потенциалов на электродах на нижней и верхней поверхностях. Сетки были построены в пакете COMSOL Multiphysics для двух видов элементов: параллелепипедов и тетраэдров, рис.

5. В дальнейшем все данные, относящие к задаче с элементами-параллелепипедами, будем называть hex, а к задаче с тетраэдрами – tet.

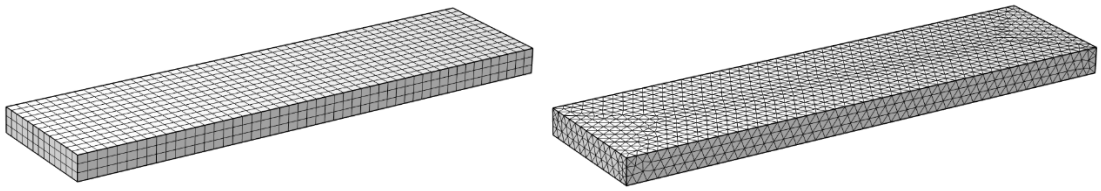

Рис. 5. Конечно-элементные сетки модельных задач hex (слева) и tet (справа).

Полученные сетки были экспортированы в формате. nas в пакет ACELAN-COMPOS, где на их основе были построены глобальные матрицы жесткости для статической электроупругой задачи, получаемой из постановки (1) при $\omega = 0$ для линейных элементов. Структура матриц представлена на рис. 6 и 7.

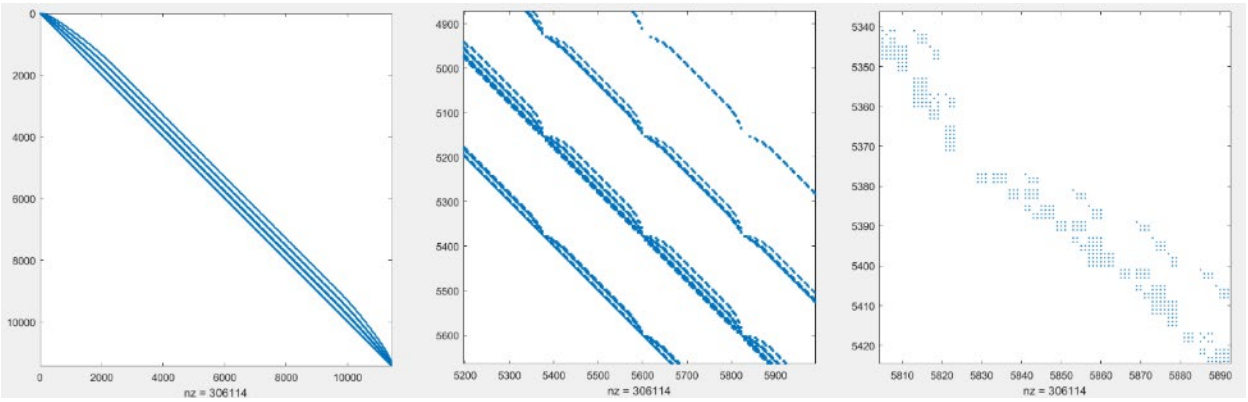

Рис. 6. Структура заполненности матрицы hex на трех уровнях приближения.

Из рисунков видно, что обе матрицы имеют ленточную структуру, которая достигается путем эффективной перенумерации геометрических узлов пакетом COMSOL еще на этапе построения сетки. При этом не выделяется блочная структура на макроуровне, так как была использована нумерация неизвестных $[ux_0, uy_0, uz_0, \varphi_0 \ldots ux_n, uy_n, uz_n, \varphi_n]$. Отметим, что в рамках данной работы такая нумерация вполне допустима, однако при переходе к решению задачи на собственные значения для ряда методов [12]

нулевые значения в матрице масс удобнее сгруппировать при помощи пересортировки $[ux_0, uy_0, uz_0 \dots ux_n, uy_n, uz_n, \varphi_0 \dots \varphi_n]$.

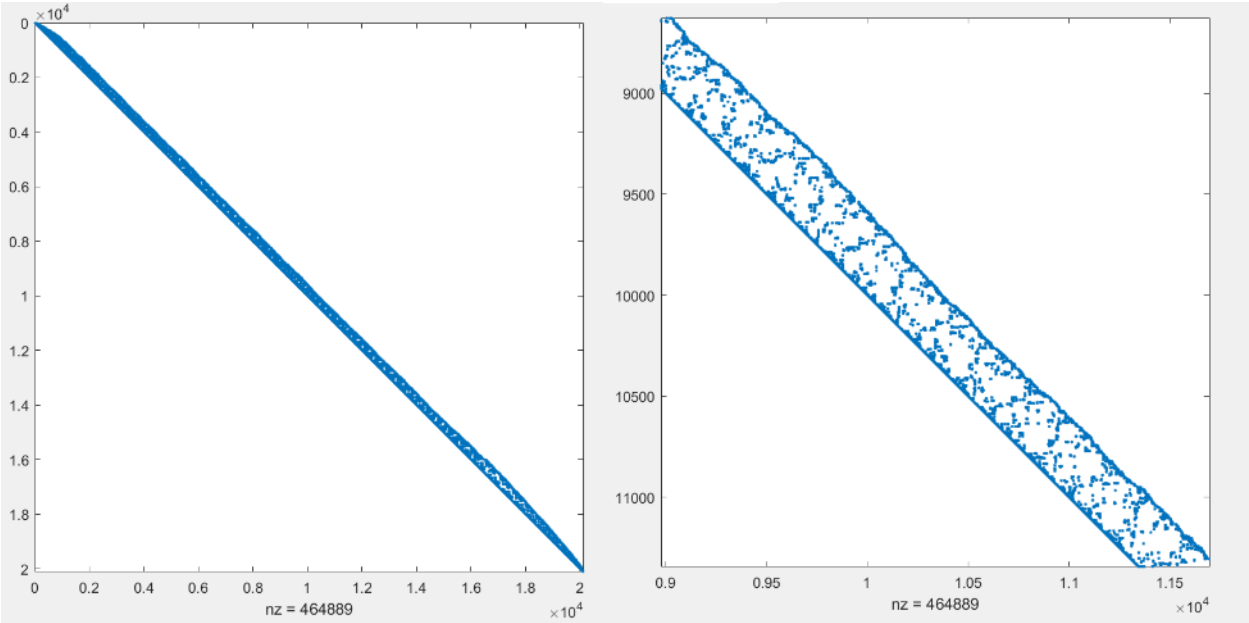

Рис. 7. Структура заполненности матрицы для сетки tet на двух уровнях приближения.

Результаты численных экспериментов

Приводятся усредненные результаты серий численных экспериментов, проведенных на устройстве со следующими характеристиками: CPU AMD Ryzen 5 4600H 3.00 GHz; 6 ядер; 12 потоков, GPU NVIDIA GeForce GTX 1650 4GB GDDR5 шина 128 бит. В первом численном эксперименте рассматривалась зависимость производительности операции умножения блочно-разреженной матрицы на вектор от размера блока для различных данных из открытых источников. Результаты представлены на рис. 8, измерялось время выполнения операции для матриц из набора CYLSHELL. Блок размера 1 соответствует базовому формату CSR и не имеет преимуществ, связанных с учетом структуры матрицы. Максимальный размер блока – 16, превышает размеры блоков, возникающих в глобальных матрицах для задач, рассматриваемых в наборе данных CYLSHELL. Видно,

что наиболее эффективными оказываются размеры блоков от 4 до 8 в зависимости от задачи, при этом с ростом размера блока повышается количество нулей, захватываемых блоком, что приводит к снижению производительности.

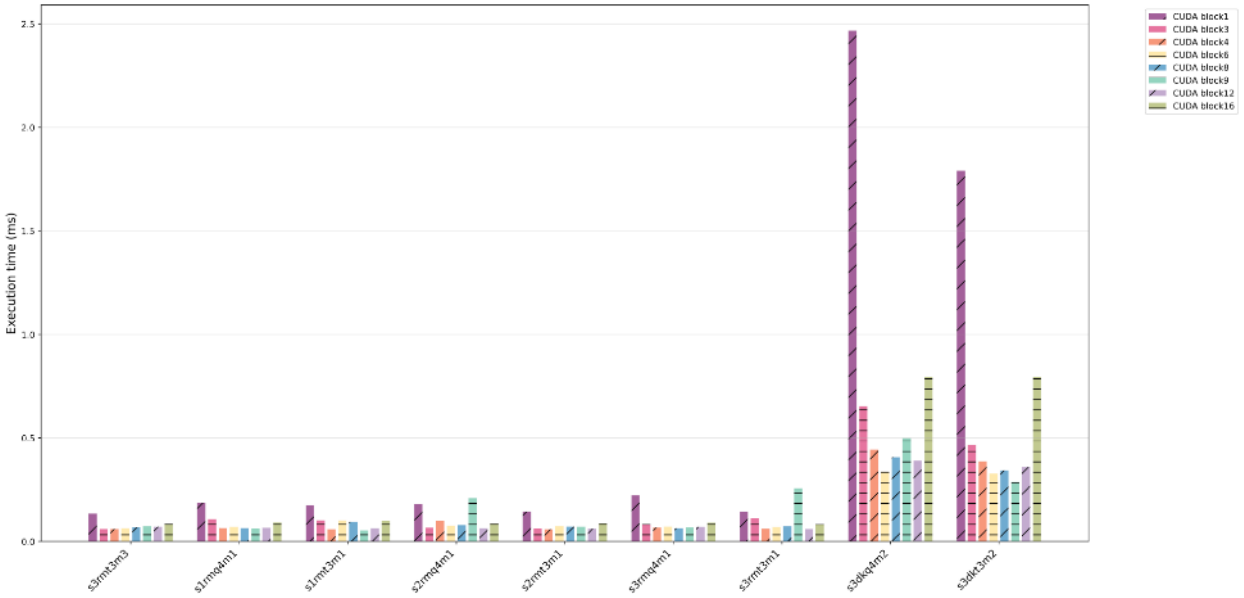

Рис 8. Среднее затраченное время на умножение матрицы на вектор для набора данных CYLSHELL.

Аналогичный эксперимент был проведен для более разнообразного набора матриц из SuiteSparse Matrix Collection, результаты представлены на рис. 9. Как видно, в случае, когда природа матрицы заранее неизвестна, выбор более крупных блоков может привести к заметной потере производительности. При этом блоки размеров 3 и 4 оказываются универсальными и для больших матриц дают кратный прирост скорости. В данной работе использовалось фиксированное количество потоков равное 128, что соответствует возможностям потребительских GPU Nvidia прошлого и позапрошлых поколений. Полная утилизация наиболее современных потребительских GPU и профессиональных устройств может быть достигнута при большем числе потоков для больших матриц. На рис. 10–12 представлены сравнительные результаты серий численных экспериментов, проведенных с использованием

разработанных программных модулей и пакета MATLAB. При усреднении авторы избегали возможного кэширования данных на GPU, MATLAB запускался без каких-либо дополнительных настроек.

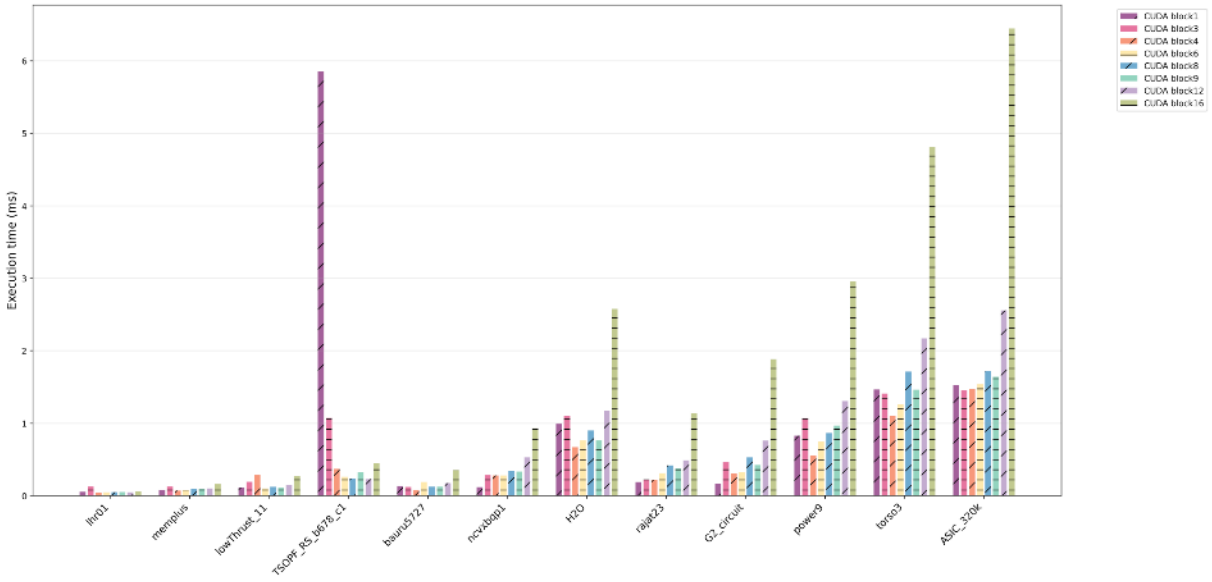

Рис. 9. Среднее затраченное время на умножение матрицы на вектор для набора данных SuiteSparse Matrix Collection.

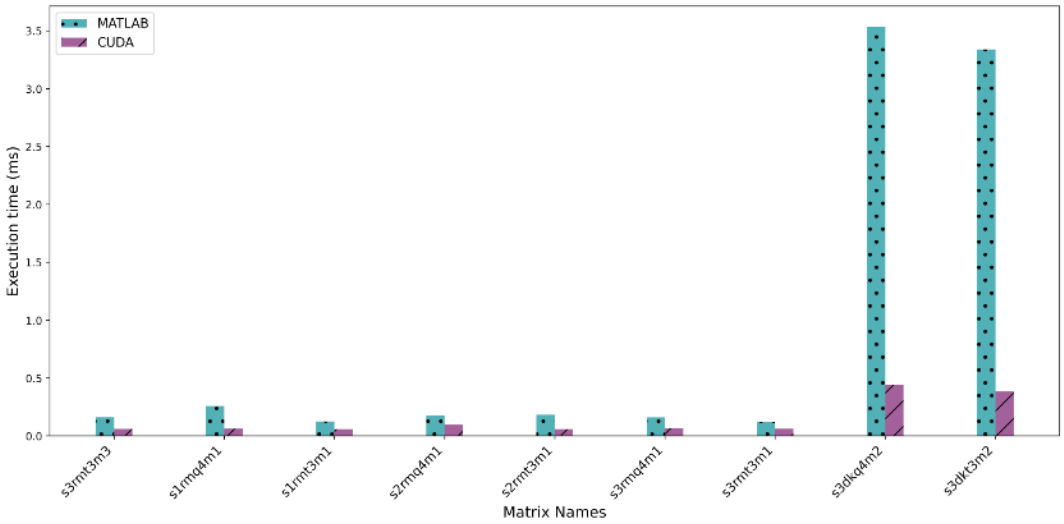

Рис. 10. Сравнение среднего времени умножения блочно-разреженной матрицы на вектор в реализованном пакете на платформе CUDA и в MATLAB для набора данных CYLSHELL.

Для матриц порядка 5000 строк разница несущественная, в то время как для матриц порядка 90000 строк предложенное решение показывает прирост производительности до 10 раз. Это связано со временем передачи данных по шине от хоста к GPU и обратно. В случае большого объема данных это время нивелируется активной работой на GPU, но для маленьких матриц выгоднее провести вычисления на CPU без дополнительной пересылки данных. Это подтверждается результатами, представленными на рис. 11.

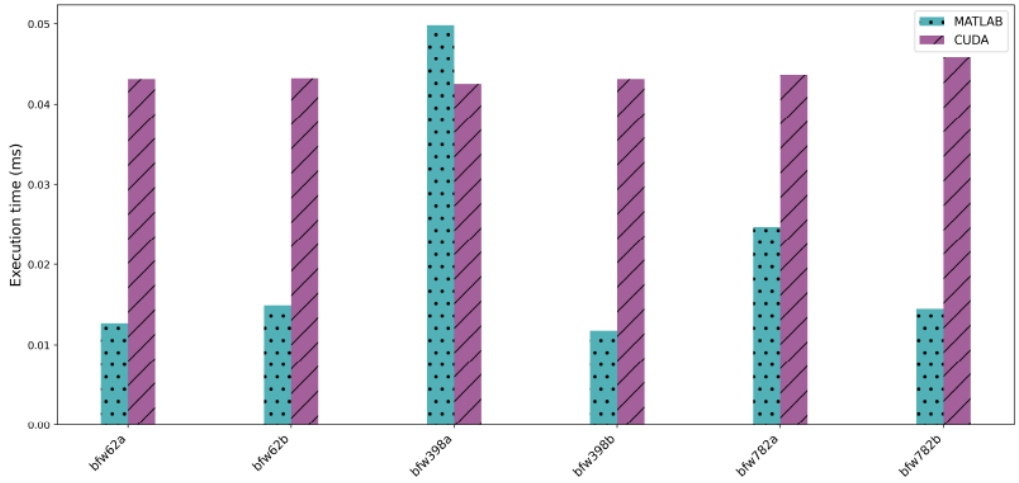

Рис. 11. Сравнение среднего времени умножения блочно-разреженной матрицы на вектор в реализованном пакете на платформе CUDA и в MATLAB для набора данных BFWAVE.

В случае, когда размер разреженной матрицы не превышает 5-10 тысяч строк, использование блочно-разреженной структуры на GPU не дает прироста скорости относительно базовых реализаций разреженных структур на CPU. При этом небольшие размеры блоков оказываются эффективными для больших матриц даже в ситуациях, когда исходная структура матрицы не известна, что демонстрируется на рис. 12. Так, матрица ASIC 320k, визуализированная на рис. 4, построена для решения задачи

схемотехнического моделирования и не соответствует предположениям, сделанным для метода конечных элементов. Однако использование блоков размера 3 и 4 ускоряет операцию умножения на вектор в 8–9 раз.

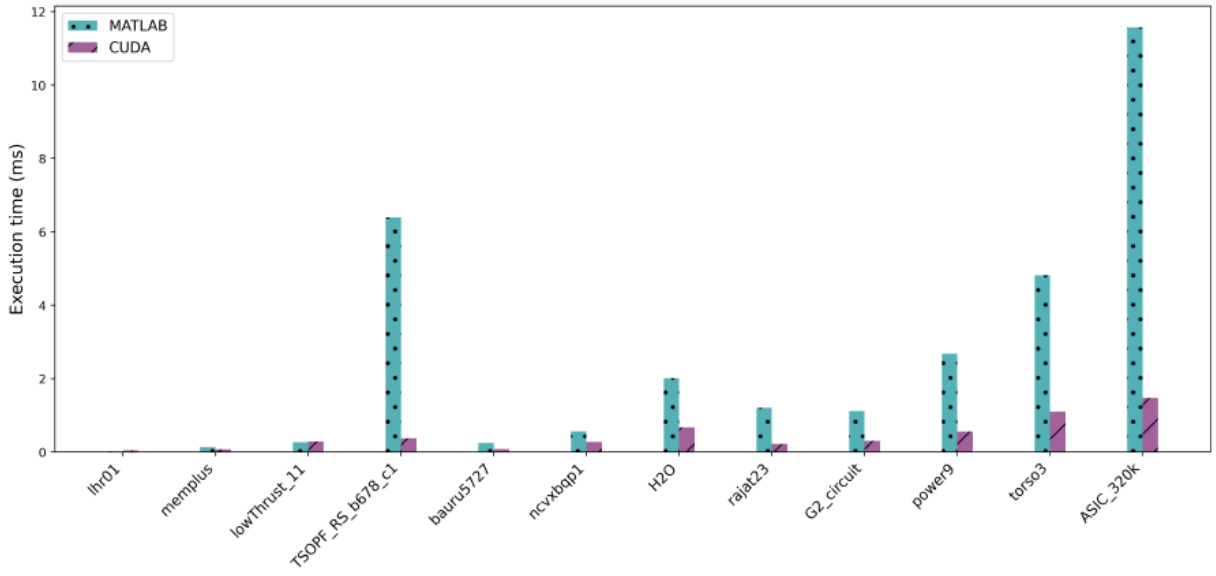

Рис. 12. Сравнение среднего времени умножения блочно-разреженной матрицы на вектор в реализованном пакете на платформе CUDA и в MATLAB для матриц из коллекции SuiteSparse Matrix Collection.

Для задач hex и tet было проведено решение системы линейных уравнений методом BiCGStab. Результаты расчетов представлены в таблице 1.

Таблица № 1

Результаты решения СЛАУ методом BiCGStab

Задача	Решатель	Невязка	Относительная ошибка по сравнению с MATLAB
hex	Блочно-разреженный	$2 \cdot 10^{-6}$	0,0017%
hex	MATLAB	$1 \cdot 10^{-16}$	-
tet	Блочно-разреженный	$1{,}2 \cdot 10^{-6}$	0,0018%
tet	MATLAB	$7 \cdot 10^{-17}$	-

Как видно, в обоих случаях получена допустимая невязка, представленная реализация BiCGStab сошлась к целевой невязке, указанной при запуске, в то время как MATLAB получил более точные результаты при настройках по умолчанию. Нужно отметить, что MATLAB оперировал типом double, и именно в таком типе данных проводились итоговые сравнения. Вычисления на GPU проводились с типом float. Переход к типу double возможен, но на данный момент представляется избыточным, так как относительная ошибка, вычисленная как нормированное расстояние между решениями, оказалась мала. Кроме того, в прикладных задачах, решаемых методом конечных элементов, входные параметры, а именно материальные свойства, редко имеют точность свыше 1%. Более важными для точности итогового решения являются вопросы качества сетки и степень конечного элемента.

Заключение

Реализованный формат хранения блочно-разреженных матриц, использующие его операции и основанных на этих операциях стабилизированный метод бисопряженных градиентов могут успешно применяться для решения статических задач теории упругости, электроупругости и других связанных задач. Полученные программные модули были протестированы на разреженных матрицах из открытых источников и на глобальных матрицах, построенных в пакете ACELAN-COMPOS. Был проведен сравнительный анализ производительности основной операции – умножения блочно-разреженной матрицы на плотный вектор. Данные результаты позволяют применить разработанные модули для реализации более сложных алгоритмов, в том числе для решения обобщенной задачи на собственные значения.

Исследование выполнено за счет гранта Российского научного фонда (проект № 24–21–00205) в Южном федеральном университете

Литература (References)

1. Belokon A.V., Eremeyev V.A., Nasedkin A.V., Solov'yev A.N. Partitioned schemes of the finite element method for dynamic problems of acoustoelectroelasticity. J. Appl. Math. Mech. 2000. Vol. 64. No. 3. Pp. 367-377.

2. Kudimova A.B., Nadolin D.K., Nasedkin A.V., Nasedkina A.A., Oganesyan P.A., Soloviev A.N. Finite element homogenization of piezocomposites with isolated inclusions using improved 3-0 algorithm for generating representative volumes in Acelan-Compos package. Materials Physics and Mechanics. 2020. Vol. 44. Iss. 3. Pp. 392-403.

3. Steinberg B.Ya., Steinberg O.B., Oganesyan P.A., Vasilenko A.A., Veselovskiy V.V., Zhivykh N.A.. Fast Solvers for Systems of Linear Equations with Block-Band Matrices. East Asian Journal on Applied Mathematics. 2023. Vol. 13. No. 1. Pp. 47-58

4. Papadimitriou, C.H. The NP-Completeness of the bandwidth minimization problem. Computing. 1976. 16. Pp. 263–270.

5. Cuthill, E., & McKee, J. Reducing the bandwidth of sparse symmetric matrices. Proceedings of the 24th national conference of the ACM. New York. ACM. 1969. Pp. 157–172.

6. Im, E. J., Yelick, K. Optimizing sparse matrix computations for register reuse in SPARSITY. International Conference on Computational Science. Heidelberg. Springer. 2001. Pp. 127-136.

7. Pinar, A., Heath, M. T. Improving performance of sparse matrix-vector multiplication. Proceedings of the 1999 ACM/IEEE conference on Supercomputing. New York. ACM. 1999. P. 30.

8. Van der Vorst, H.A. Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems. SIAM Journal on Scientific and Statistical Computing. 1992. 13(2). Pp. 631–644.

9. Kouhia, R. CYLSHELL: Finite element analysis of cylindrical shells. In The University of Florida Sparse Matrix Collection. sparse.tamu.edu/Cylshell, accessed on 19.11.2025.

10. B. Schultz, S. Gedney. BFWAVE: Bounded Finline Dielectric Waveguide. math.nist.gov/MatrixMarket/data/NEP/bfwave/bfwave.html, accessed on 19.11.2025.

11. Davis T.A., Hu Y. The University of Florida Sparse Matrix Collection. ACM Transactions on Mathematical Software. 2011. Vol. 38, № 1(1). 25 p.

12. Martynova T., Muratova G., Oganesyan P., Shtein O. The numerical solution of large-scale generalized eigenvalue problems arising from finite element modeling of electroelastic materials. Symmetry. 2023. 15. 171. URL: mdpi.com/2073-8994/15/1/171

Авторы согласны на обработку и хранение персональных данных.

Дата поступления: 8.12.2025

Дата публикации: 13.01.2026