

Гетерогенный параллельный алгоритм для моделирования многоступенчатых турбомашин на основе метода нелинейных гармоник

Р. Р. Гайсин¹, А. В. Горобец¹, А. П. Дубень¹

¹Институт прикладной математики имени М. В. Келдыша РАН

Суперкомпьютерные дни в России, 2026

Актуальность и цель работы

- Моделирование нестационарных течений в турбомашинах (компрессоры, турбины) требует учёта эффектов взаимодействия ротор–статор.
- В промышленных расчетах для моделирования турбомашин используются стационарные RANS-подходы.
- Рассматривается один межлопаточный канал на венец в условиях окружной периодичности.
- Для интерфейсов ротор–статор используется метод поверхности смешения (Mixing Plane).

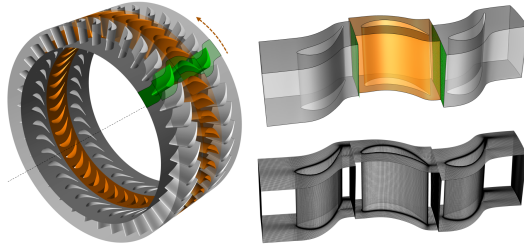


Рис. 1: Расчётная область для модельной турбины

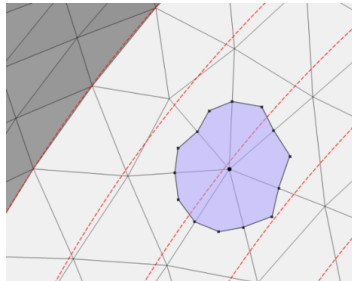
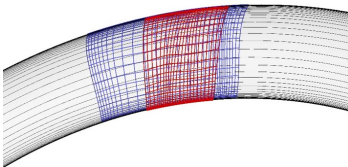
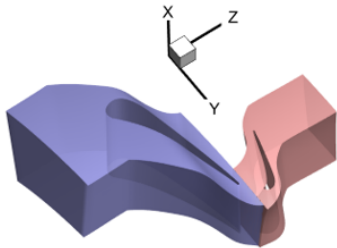


Рис. 2: Mixing Plane

- МР – одна на интерфейс
- Каждый след CV дробится на пересечения с различными полосами
- Полностью консервативный интерфейс: используется солвер Роу для решения задачи Римана

Актуальность и цель работы (продолжение)

- Метод нелинейных гармоник (NLH) позволяет учитывать нестационарные эффекты на основе недорогих RANS-подходов
- Метод широко используется на практике (например, Cadence Fidelity Fine/Turbo - бывш. NUMECA)
- Ранее метод реализован в коде NOISEtte для неструктурированных сеток с параллелизацией MPI+OpenMP.
- **Цель:** разработка гетерогенного параллельного алгоритма NLH с использованием GPU (OpenCL) для эффективных промышленных расчётов на гибридных кластерах.

- NLH раскладывает решение на осреднённую по времени часть и набор гармоник:

$$Q(t, x) = \overline{Q}(x) + \sum_{k=1}^{N_h} \left[\tilde{Q}_k e^{ik\omega t} + \overline{\tilde{Q}}_k e^{-ik\omega t} \right].$$

- Каждая гармоника добавляет к базовой RANS-системе два набора переменных (действительная и мнимая части).
- Метод NLH требует глобальной модификации всего алгоритма.

- Исходные уравнения — осреднённые по Рейнольдсу уравнения Навье–Стокса (RANS) для сжимаемого вязкого потока:

$$\frac{\partial Q}{\partial t} + \nabla \cdot F^C(Q) - \nabla \cdot F^D(Q) = 0.$$

- Вектор консервативных переменных: $Q = (\rho, m, E)^T$.
- Потоки F^C и F^D включают конвективные и диффузионные члены.
- Для гармонических амплитуд система принимает вид:

$$\frac{\partial \tilde{Q}}{\partial t} + \nabla \cdot \tilde{F}^C(\tilde{Q}, \bar{Q}) + i\omega \tilde{Q} - \nabla \cdot \tilde{F}^D(\tilde{Q}, \bar{Q}) = 0.$$

- Пространственная дискретизация: схемы повышенной точности EBR на неструктурированных сетках (EBR-TVD или EBR-WENO), вязкие члены — метод усреднённого разбиения элементов (AES).
- По времени — неявная схема BDF с квази-ньютоновской линеаризацией.
- Псевдовременная итерационная формула:

$$\left(\frac{1}{\tau^n} + \tilde{J}\right) (\tilde{Q}^{n+1} - \tilde{Q}^n) = \tilde{\Phi}(\tilde{Q}^n, \overline{Q}^n, \omega),$$

где $\tilde{\Phi} = -\nabla \cdot \tilde{\mathcal{F}}^C(\tilde{Q}, \overline{Q}) - i\omega\tilde{Q} + \nabla \cdot \tilde{\mathcal{F}}^D(\tilde{Q}, \overline{Q})$.

- Линейная система решается методом BiCGStab с предобуславливателями на основе метода Гаусса–Зейделя.
- Системы для осреднённых переменных и гармоник решаются по отдельности на каждом псевдошаге.

- Для гармоник матрица Якоби имеет блочную структуру 10×10 :

$$A_{ii} = \begin{pmatrix} A_{ii} & -\omega v_i I \\ \omega v_i I & A_{ii} \end{pmatrix}, \quad A_{ij} = \begin{pmatrix} A_{ij} & 0 \\ 0 & A_{ij} \end{pmatrix}, \quad i \neq j.$$

- Такая структура позволяет хранить не 100, а 26 значений на блок.
- Матрица для гармоник зависит только от осреднённых переменных, кроме диагональных элементов.
- После построения матрицы для осреднённых переменных она переиспользуется для гармоник с обновлением только диагонали.
- Это даёт четырёхкратную экономию памяти GPU.

- OpenCL-версия решателя BiCGStab адаптирована под специальный формат матриц.
- Реализованы специализированные кернел-функции для SpMV.
- Предобуславливатель на основе блочного метода Якоби/Гаусса–Зейделя переработан.
- Используется приближённая инверсия диагональных блоков с дополнительным членом Ωx_{im} .
- Все вычисления перенесены в OpenCL-ядра, исходный код хранится во внешних файлах.



Рис. 3: Схема одного шага неявного интегрирования по времени

Параллельная реализация (продолжение)

- Алгоритм полностью адаптирован к потоковой обработке.
- Из-за ограничений локальной памяти GPU вычисление потоков выполняется в цикле по гармоникам.
- Пульсационная часть переменных объединена в один массив для сокращения вызовов функций.
- Решатель запускается в цикле по гармоникам с использованием одной матрицы.
- Это кратно сокращает потребление памяти GPU на хранение якобиана и решения.

Оптимизация матрично-векторного умножения (SpMV)

- Разработаны две версии SpMV:
 - ❶ «Наивная»: один блок из 10 элементов обрабатывается одной GPU-нитью.
 - ❷ Оптимизированная: каждый блок обрабатывается 60 GPU-нитеями (по 5 нитей на строку блока A_{ij} и одна на внедиагональный элемент).
- Ускорение оптимизированной версии по сравнению с наивной: **в 3,5–3,9 раза** на сетках от 64 тыс. до 1,7 млн узлов.

Таблица 1: Время выполнения (с) различных версий операции SpMV

Версия SpMV	64000 узлов	512000 узлов	1728000 узлов
Одна GPU-нить на блок	0.0006	0.0057	0.0196
60 GPU-нитей на блок	0.0002	0.0015	0.0050

- **NASA Rotor-67** — изолированное рабочее колесо с 22 лопатками, сетка 300 тыс. узлов.
- **Aachen Turbine** — одноступенчатая осевая турбина (статор, ротор, статор), сетка 3 млн узлов.
- **Многоступенчатый компрессор** — входной направляющий аппарат и 4 ступени ротор–статор, 8 интерфейсов, сетка 11,9 млн узлов.

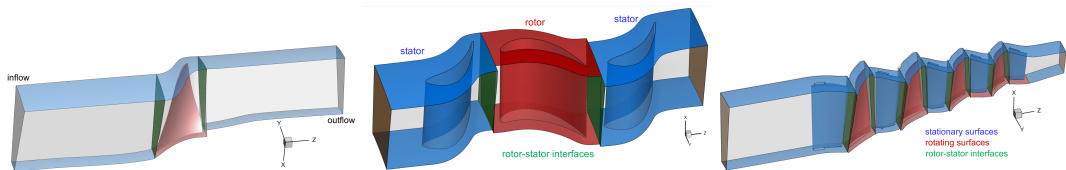


Рис. 4: Тестовые задачи: NASA Rotor-67, Aachen Turbine, многоступенчатый компрессор

Анализ распределения времени этапов CPU и GPU

- Тестовая задача Aachen Turbine: 3 млн узлов, 3 гармоники.
- CPU: Intel Xeon E5-2697 v2 (2.7 ГГц), время шага 37,8 с.
- GPU: NVIDIA RTX A5000, время шага 2,48 с.

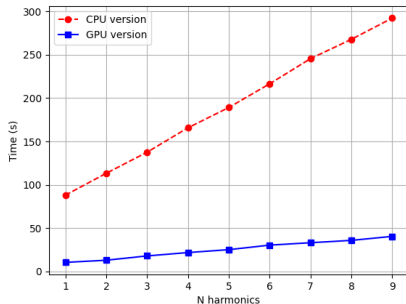
Таблица 2: Относительное время этапов шага неявного интегрирования, %

Этап	CPU-версия	GPU-версия
Вычисление осреднённых переменных	27	19
Вычисление потоков	25	15,6
Mixing Plane	0,3	3,4
Решатель BiCGSTAB	46	64
Предобуславливатель	36,7	53,4
SpMV	6,4	2,4

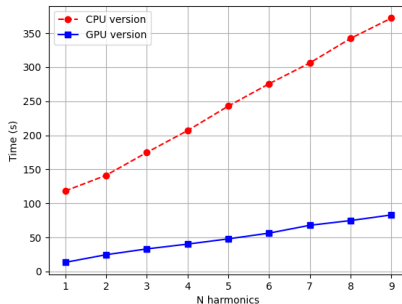
- Предобуславливатель был оптимизирован по аналогии с SpMV, ускорение 2 – 3 раза.

Сравнение производительности CPU и GPU

- На графике слева тестовая задача Rotor67: Intel Xeon CPU E5-2697 v2 (12 ядер) vs GPU NVIDIA RTX A5000.
- Справа тестовая задача Aachen Turbine: 2 Intel Xeon Gold 5218 (по 16 ядер) vs 2 GPU NVIDIA RTX A5000.
- Ускорение GPU-версии относительно CPU: **примерно в 5 – 7 раз.**



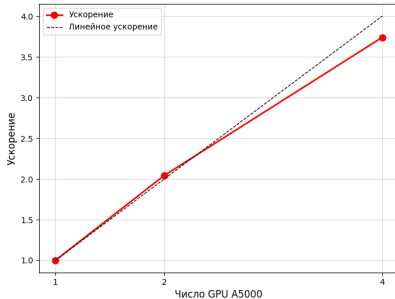
Rotor67



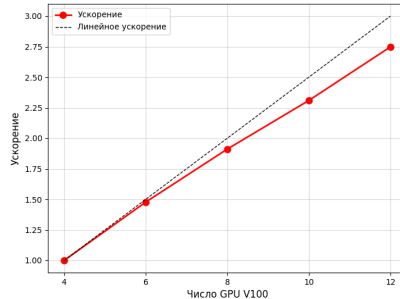
Aachen Turbine

Параллельное ускорение

- Aachen Turbine: запуск на 1, 2, 4 GPU NVIDIA A5000.
- Многоступенчатый компрессор: кластер с 4 GPU NVIDIA V100 на узел, запуск на 4, 6, 8, 10, 12 GPU.
- Наблюдается близкое к линейному ускорение при увеличении числа GPU.



Aachen Turbine



Многоступенчатый компрессор

- Разработана гетерогенная параллельная реализация метода NLH в коде NOISEtte.
- Реализована оптимизированная версия решателя СЛАУ для специализированного формата матрицы.
- Показано ускорение GPU-версии относительно многоядерного CPU 5 – 7 раз на тестовых задачах.

Спасибо за внимание!

Вопросы?