Применение алгоритмов оптимизации параметров при решении систем уравнений на центральных процессорах и графических ускорителях

Б.И. Краснопольский[1], Р.М. Куприй[1,2]

[1] НИИ механики МГУ

[2] Факультет ВМиК МГУ

Суперкомпьютерные дни в России, Москва, 29 сентября 2025 г.

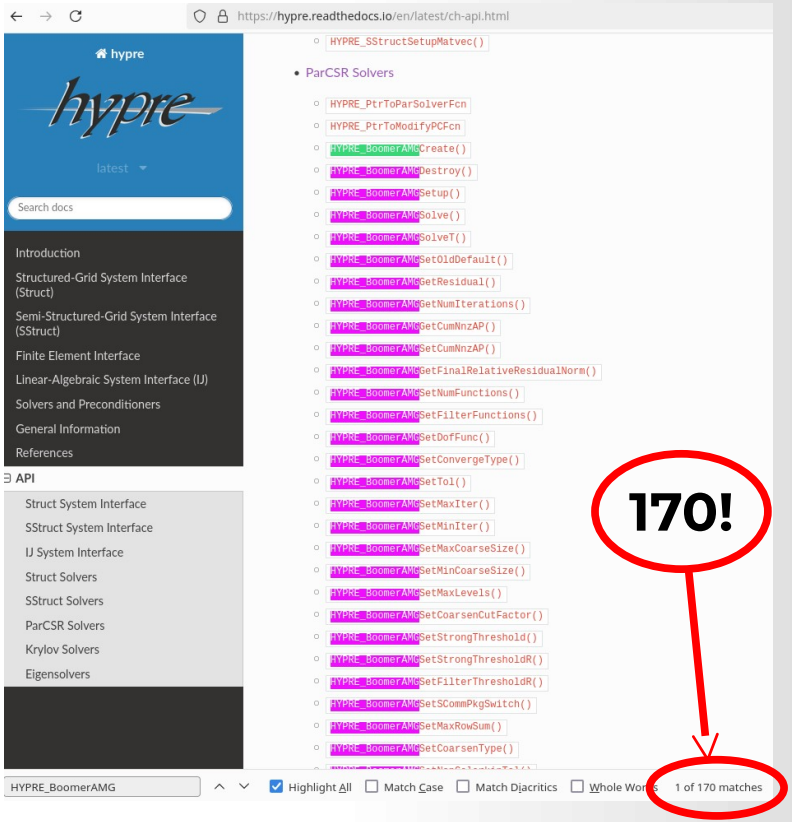
Решение систем уравнений

- Прямые и итерационные методы
- Итерационные методы:
  - + Обычно более универсальные
  - + Меньше ограничений по потребляемой памяти
  - + Лучше масштабируемость
  - - Большое количество различных методов
  - - Множество настроечных параметров у наиболее перспективных методов...

Параметры многосеточного метода

Параметры в *hypre*:

- 170 функций для задания параметров
- Нет универсальных значений, только общие рекомендации
- Реализации методов для CPU и частично — для GPU: эффективность методов существенно разная



Особенности развития оборудования

- **Ориентация на алгоритмы AI, но не задачи мат. моделирования**
- Сопроцессоры и ускорители вычислений
  - AMD Instinct MI, NVidia Tesla GPU, Intel Xe-HPC

	CPU	GPU	
Model	AMD Rome 7742	NVidia A100	
Bandwidth	204.8 GB/s	~2000 GB/s	**x10**
Fp64 performance	2.3 TFlop/s	9.7 TFlop/s	**x4**

Как бы сделать так, чтобы...

... можно было бы:

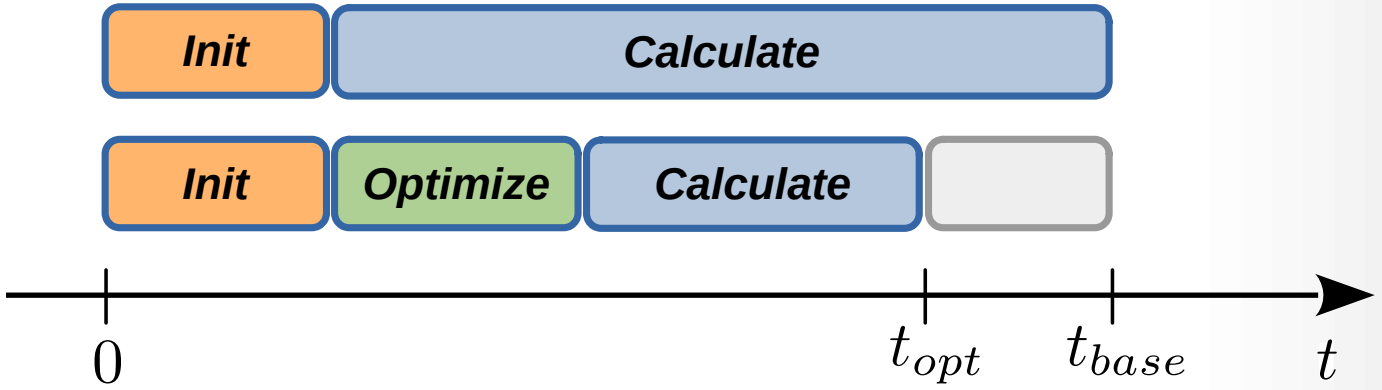
- быстро решать различные системы уравнений
- не заниматься долгим ручным подбором параметров
- иметь адаптацию под конкретную вычислительную платформу
- ...
- **и... и... и чтобы само...**

vecteezy.com

Сценарий использования

Моделирование несжимаемых турбулентных течений (HighRes):

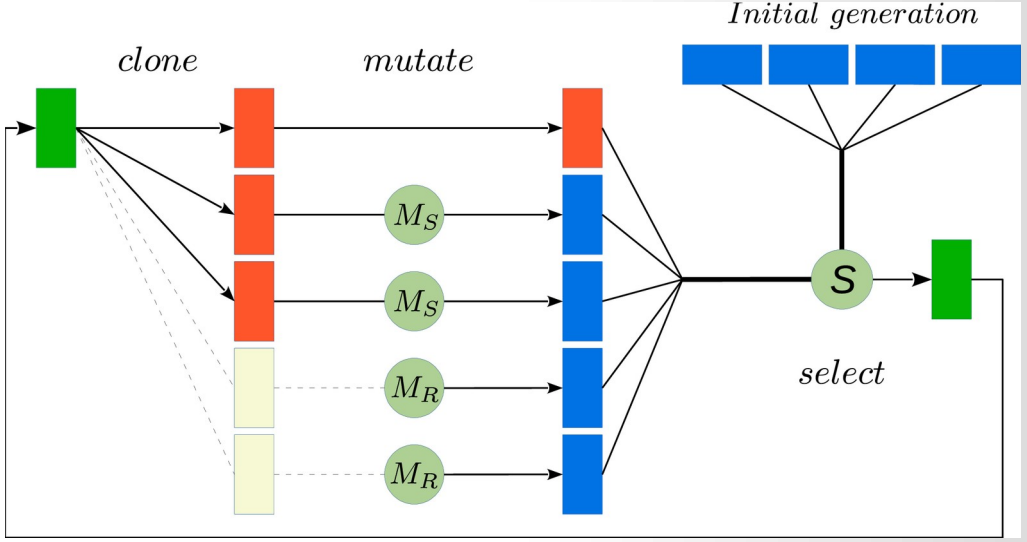
- Множество шагов по времени
- Решение СЛАУ для давления, матрица неизменна в ходе расчета
- Оптимизация до начала основного расчета



Гибридная эволюционная стратегия

Оптимизационный алгоритм: клонирование, мутация, отбор

- Мутация:
  - мягкие мутации (M_S)
  - случайные мутации (M_R)
  - НС как фильтр для оператора случайных мутаций
- Отбор: на основе прямого решения СЛАУ



- вектор параметров численного метода

Дальнейшее развитие алгоритма

Было ранее:

- оптимизация времени решения СЛАУ (“solve”-часть)

Дополнение:

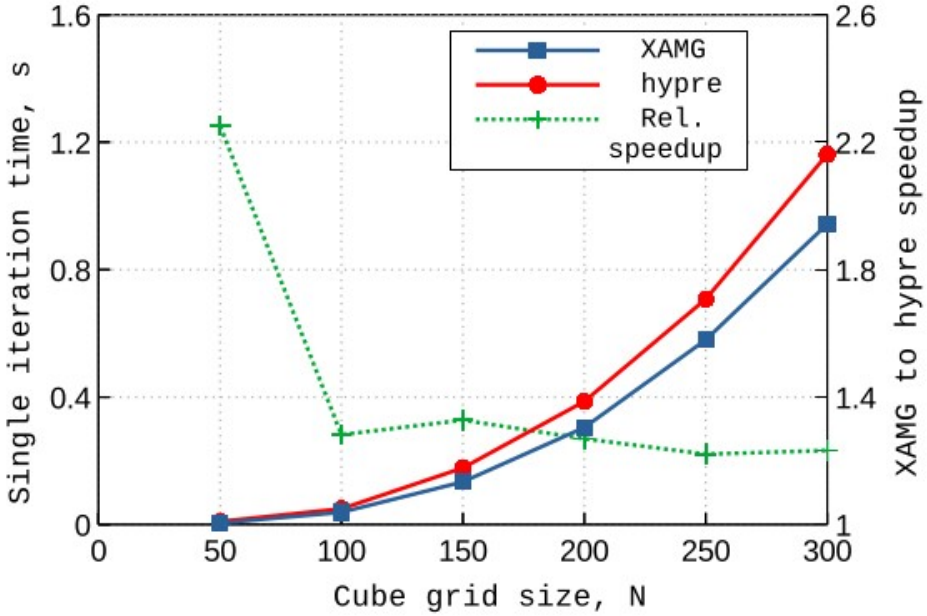
- оптимизация полного времени решения СЛАУ
- оптимизация с учетом ограничений по памяти
- оптимизация под разные вычислительные устройства

Платформа для экспериментов

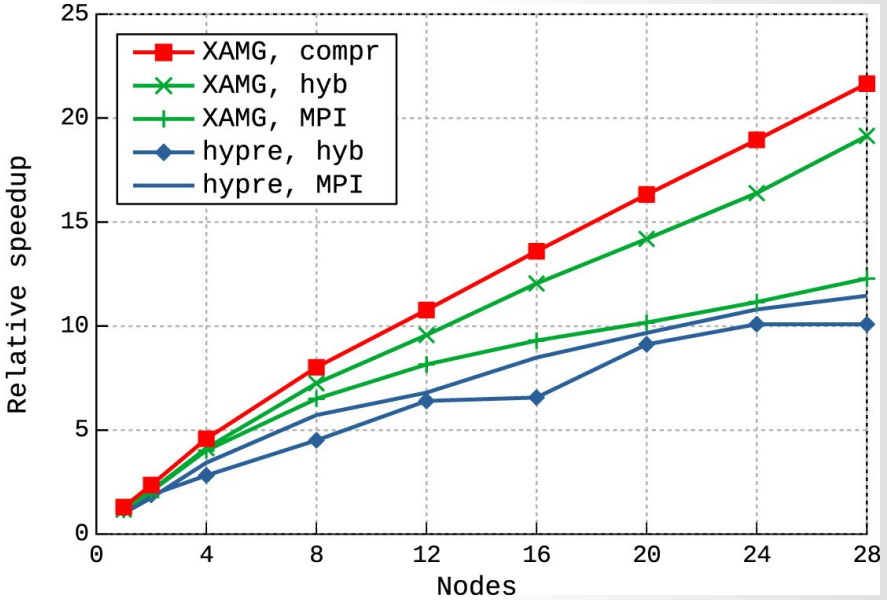
XAMG: https://gitlab.com/xamg/xamg

- C++11, yaml, C API
- Шаблоны, выравнивание данных, векторизация, прагмы для компиляторов, гибридная модель MPI+Posix ShM, и т.п.
- Смешанная точность вычислений, итерационное уточнение
- “Стандартный” набор итерационных методов:
  - Крыловские методы (CG, BiCGSstab, модифицированные варианты)
  - Классический алгебраический многосеточный метод (**setup** из ***hypre***)
  - Метод Чебышева, Якоби, гибридный метод Гаусса-Зейделя

Масштабируемость, сравнение с *hypre*



Вычисления на 1 узле, все ядра



Масштабируемость, $N=250^3$

«solve»-часть, Ломоносов-2

Тестовая платформа и набор матриц

Рабочая станция:

- CPU: Intel Core-i7 11700 (8 cores, ~43 GB/s)
- GPU: NVidia 3060Ti (~417 GB/s)

Набор тестовых матриц:

- уравнение Пуассона с переменным коэффициентом:

$$-\nabla \cdot (\kappa \nabla u) = f \qquad \kappa = \begin{cases} 10^3, & \mathbf{x} \in [0.1, 0.9]^3, \\ 0.1, & \mathbf{x} \in \text{cubes } 0.1^3 \text{at the corners}, \\ 1, & \text{elsewhere} \end{cases}$$

- SuiteSparse Matrix Collection

Конфигурации методов

Базовая конфигурация итерационных методов:

- BiCGStab + Algebraic Multigrid + Chebyshev Polynomials
- Точность решения: rel_tol < 10^{-8}

Параметры оптимизационного алгоритма:

- Вектор из 13 параметров
- 20 индивидуумов: 10 soft + 10 random, без HC
- Макс. число поколений: 15
- «baseline» как начальное приближение

Параметры

Parameter	“baseline”
Preconditioner:	
max_iters	1
mg_cycle	V
mg_max_row_sum	1
mg_nonGalerkin_tol	0
mg_coarse_matrix_size	500
mg_num_paths	3
mg_coarsening_type	6
mg_interpolation_type	0
mg_strength_threshold	0.25
mg_trunc_factor	0.3
mg_Pmax_elements	4
mg_agg_num_levels	2
mg_agg_interpolation_type	4
mg_agg_trunc_factor	0.3
mg_agg_P12_trunc_factor	0
mg_agg_Pmax_elements	4
mg_agg_P12max_elements	4
Pre/post-smoother:	
polynomial_order	2
spectrum_fraction	0.3

- «Базовый» вектор параметров
  - что-то, подобранное опытным путем при решении разных СЛАУ
- Жирным выделены оптимизируемые параметры

Оптимизация полного времени решения СЛАУ

Целевая функция: полное время решения СЛАУ

	“baseline”			Оптимизация времени solve-части			Оптимизация времени всего расчета		
	Setup, s	Solve, s	Total, s	Setup, s	Solve, s	Total, s	Setup, s	Solve, s	Total, s
cube100	0.87	0.62	1.49	1.66	**0.42**	2.08	0.45	0.59	**1.04**
jumps100	1.03	0.80	1.83	2.09	**0.40**	2.49	0.43	0.68	**1.11**

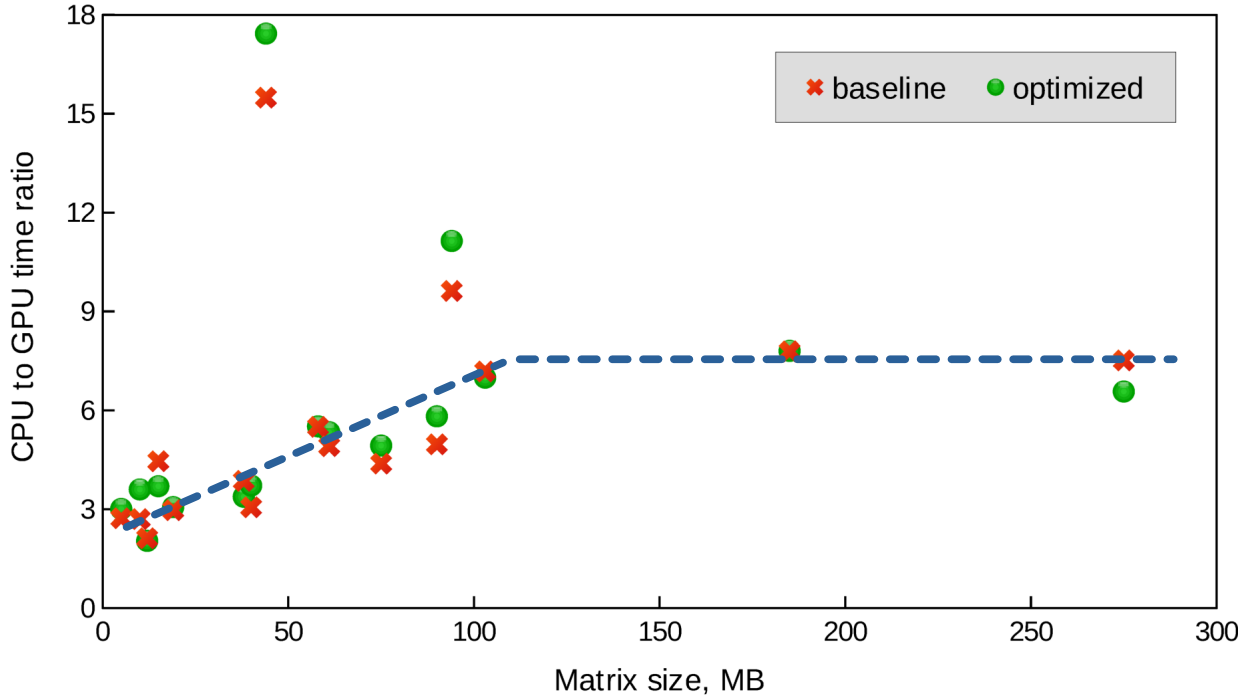
* Расчеты на CPU, 8 ядер; характерное время на оптимизацию — 10-15 минут (100-200 испытаний)

Оптимизация с ограничением объёма используемой памяти

Целевая функция: время «solve»-части и ограничение по памяти

	“baseline”			Оптимизация времени solve-части, без доп. ограничений			Оптимизация времени solve-части, M < 1400 MB		
	Setup, s	Solve, s	Total, s	Setup, s	Solve, s	Total, s	Setup, s	Solve, s	Total, s
jumps100	1.03	0.80	1.83	1.61	**0.48**	2.09	1.91	**0.54**	2.45
memory	1348 MB			1535 MB			1281 MB		

Ускорение при переходе на графические ускорители



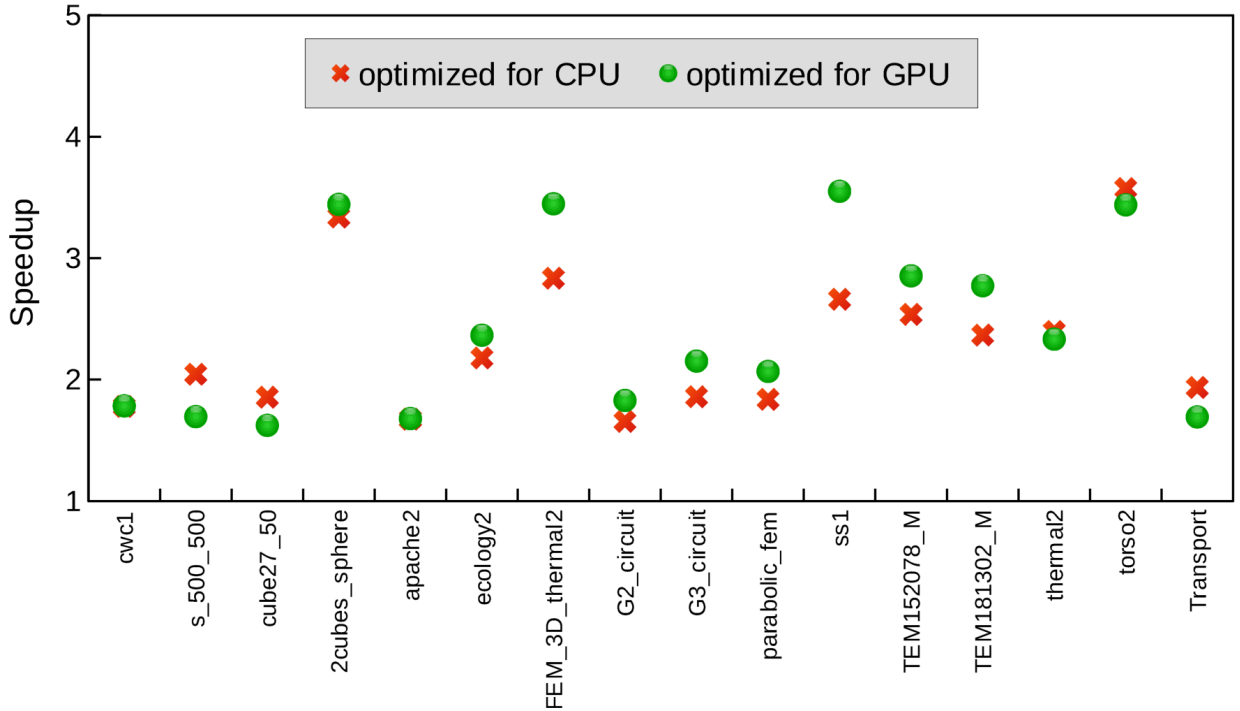
$$P_{baseline} = \frac{T^{CPU}_{baseline}}{T^{GPU}_{baseline}}$$

$$P_{optimized} = \frac{T^{CPU}_{optimized}}{T^{GPU}_{optimized}}$$

- 13 матриц из SSMC
- 3 сгенерированных матрицы

Ускорение зависит от объема занимаемой памяти (эффективности кэша CPU)

Эффект от оптимизации для CPU и GPU



$$S^{CPU} = \frac{T^{CPU}_{baseline}}{T^{CPU}_{optimized}}$$

$$S^{CPU} = \frac{T^{CPU}_{baseline}}{T^{CPU}_{optimized}}$$

- 13 матриц из SSMC
- 3 сгенерированных матрицы

Среднее ускорение для CPU — 2.3 раза, для GPU — 2.4 раза

Примеры «оптимальных» параметров

Parameter	“baseline”	2cubes_sphere		FEM_3D_thermal2		torso2		cwc1	
mg_max_row_sum	1	0.55	0.45	0.55	0.75	0.15	0.05	0.2	0.2
mg_nonGalerkin_tol	0	0.05	0.15	0.05	0.1	0.5	1	0.85	0.1
mg_coarse_matrix_size	500	300	250	350	250	200	300	400	500
mg_num_paths	3	3	4	4	2	3	3	2	4
mg_coarsening_type	6	3	10	6	3	0	10	8	6
mg_interpolation_type	0	6	6	8	3	17	8	17	0
mg_strength_threshold	0.25	0.85	0.5	0.85	0.45	0.15	0.25	0.2	0.2
mg_trunc_factor	0.3	0.85	0.3	0.4	0.4	0.05	0.9	0.4	0.3
mg_agg_num_levels	2	8	9	0	0	2	0	0	0
mg_agg_interpolation_type	4	5	6	8	5	6	3	3	4
mg_agg_trunc_factor	0.3	0	0.35	0.2	0.2	0.65	0.9	0.8	0.2
polynomial_order	2/2	2/2	3/4	3/2	2/3	3/1	2/1	1/1	1/2

Заключение

- Реализован расширенный набор целевых функций для алгоритма оптимизации параметров численных методов.
- Проведено тестирование новых сценариев оптимизации полного времени решения СЛАУ и решения СЛАУ с ограничением на объем потребляемой памяти, получены адекватные результаты тестирования.
- Показана значимость оптимизации параметров под конкретную вычислительную платформу.