Основы матричных вычислений

Весенний семестр 2021

Лекция 10: Умножение матриц и вычислительная устойчивость

Максим Рахуба

Высшая Школа Экономики

План

Матричное умножение: сложность

$$C = \underset{\in \mathbb{R}^{n\times n}}{A}\ \underset{\in \mathbb{R}^{n\times n}}{B}$$ — сложность: $2n^3 + O(n^2)$

память: $O(n^2)$

Можно ли быстрее, чем за $O(n^3)$?

План

Матричное умножение: метод Штрассена[1]

$$\begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix},$$ A_{ij}, B_{ij} — $\frac{n}{2} \times \frac{n}{2}$ матрицы

"Строка на столбец":

$$\begin{aligned} C_{11} &= A_{11}B_{11} + A_{12}B_{21} \\ C_{12} &= A_{11}B_{12} + A_{12}B_{22} \\ C_{21} &= A_{21}B_{11} + A_{22}B_{21} \\ C_{22} &= A_{21}B_{12} + A_{22}B_{22} \end{aligned}$$

8 умножений и 4 сложения

$$2\left(\frac{n}{\cancel{2}}\right)^3 \cdot \cancel{8} -$$

Нет выигрыша

Штрассен:

$$\begin{aligned} M_1 &= (A_{11} + A_{22})(B_{11} + B_{22}) \\ M_2 &= (A_{21} + A_{22})B_{11} \\ M_3 &= A_{11}(B_{12} - B_{22}) \\ M_4 &= A_{22}(B_{21} - B_{11}) \\ M_5 &= (A_{11} + A_{12})B_{22} \\ M_6 &= (A_{21} - A_{11})(B_{11} + B_{12}) \\ M_7 &= (A_{12} - A_{22})(B_{21} + B_{22}) \\ C_{11} &= M_1 + M_4 - M_5 + M_7 \\ C_{12} &= M_3 + M_5 \\ C_{21} &= M_2 + M_4 \\ C_{22} &= M_1 + M_3 - M_2 + M_6 \end{aligned}$$

7 умножений и 18 сложений

[1] Strassen, V. (1969). Gaussian elimination is not optimal. Numerische mathematik, 13(4), 354-356. (https://link.springer.com/content/pdf/10.1007/BF02165411.pdf)

Матричное умножение: метод Штрассена

$$M(n) = 7\,M\left(\frac{n}{2}\right) = 7\cdot 7\,M\left(\frac{n}{4}\right) = \ldots = 7^{\log_2 n} =$$

$$= n^{\log_2 7 \approx 2.81}$$

6

$$A(n) = 7\,A\left(\frac{n}{2}\right) + 18\cdot\left(\frac{n}{2}\right)^2 \overset{\text{мастер теор.}}{\Longrightarrow} A(n) = O\left(n^{\log_2 7}\right)$$

$$7\,n^{\log_2 7} < 2n^3 \quad \text{при} \quad n \gtrsim 700$$

$\Rightarrow$ не надо "раскруч." рекурсию до конца

Мастер теор

$$T(n) = a\,T\left(\frac{n}{b}\right) + O(n^c) \text{ и } c < \log_b a \Rightarrow$$

$$T(n) = O\left(n^{\log_b a}\right)$$

Матричное умножение: метод Штрассена

$$\begin{bmatrix} C_1 & C_2 \\ C_3 & C_4 \end{bmatrix} = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix} \begin{bmatrix} B_1 & B_2 \\ B_3 & B_4 \end{bmatrix}$$

$$C_1 = A_1 B_1 + A_2 B_3$$

$$C_2 = A_1 B_2 + A_2 B_4 \Rightarrow \qquad C_k = \sum_{i,j=1}^{4} x_{kij} A_i B_j$$

$$C_3 = A_3 B_1 + A_4 B_3 \qquad x_{1,:,:} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\vdots$$

$$C_4 = A_3 B_2 + A_4 B_4$$

$$x_{kij} = \sum_{\alpha=1}^{R=7} u_{i\alpha} v_{j\alpha} w_{k\alpha}$$

7 штук

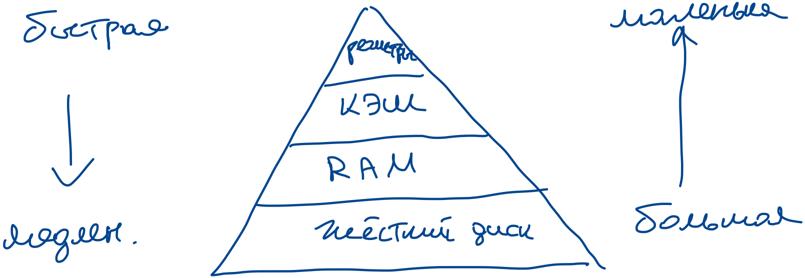
$$C_k = \sum_{\alpha=1}^{7} w_{k\alpha} \left(\sum_{i=1}^{4} u_{i\alpha} A_i \right) \cdot \left(\sum_{j=1}^{4} v_{j\alpha} B_j \right)$$

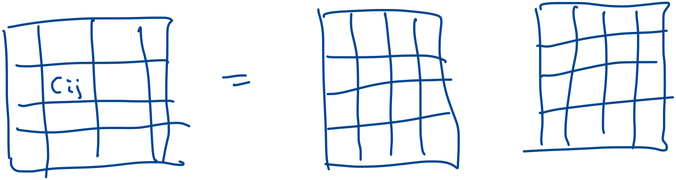
Матричное умножение: метод Штрассена

- Мировой рекорд [J. Alman, V.V. Williams, 2020][2]: $< \mathcal{O}(n^{2.37286})$. Но большая константа в $\mathcal{O}(\cdot)$.
- Неизвестен минимальный показатель α в числе операций $\mathcal{O}(n^{\alpha})$ (очевидно, $\alpha \geq 2$).
- Алгоритм Штрассена не часто используется на практике. В недавней статье[3] (2016) утверждается, что алгоритм Штрассена может быть эффективен и для небольших матриц.

[2] https://arxiv.org/pdf/2010.05846.pdf
[3] http://jianyuhuang.com/papers/sc16.pdf

Матричное умножение: иерархия памяти





$C_{ij} \mathrel{+}= A_{ik} B_{kj}$, надо, чтобы C_{ij}, A_{ik}, B_{kj} помещались в кэш при фикс k

План

BLAS (Basic Linear Algebra Subprograms)

Оригинальная версия BLAS: 1979 год на fortran. С того времени переписан множество раз, но интерфейс функций стандартизован.

Разделяют 3 уровня операций в BLAS (далее $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$):

Уровень 1

$\mathcal{O}(n)$ flops, $\mathcal{O}(n)$ memops:

$\frac{\text{flops}}{\text{memops}} = \text{const}$

$$\text{(AXPY)} \quad y \leftarrow \alpha x + y, \quad x, y \in \mathbb{F}^n, \quad \alpha \in \mathbb{F},$$

Уровень 2

$\mathcal{O}(n^2)$ flops, $\mathcal{O}(n^2)$ memops:

$$\text{(MV)} \quad y \leftarrow \alpha A x + \beta y, \quad x, y \in \mathbb{F}^n, \quad A \in \mathbb{F}^{n \times n}, \quad \alpha, \beta \in \mathbb{F},$$

обращение треугольных матриц, ранг-1 апдейт матрицы, и т.д.

Уровень 3

$\mathcal{O}(n^3)$ flops, $\mathcal{O}(n^2)$ memops:

$$\text{(MM)} \quad C \leftarrow \alpha AB + \beta C, \quad A, B, C \in \mathbb{F}^{n \times n}, \quad \alpha, \beta \in \mathbb{F}$$

Надо стараться записывать алгоритмы через матричное произведение.

BLAS (Basic Linear Algebra Subprograms)

Оригинальная версия BLAS: 1979 год на fortran. С того времени переписан множество раз, но интерфейс функций стандартизован.

Разделяют 3 уровня операций в BLAS (далее $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$):

Уровень 1

$\mathcal{O}(n)$ flops, $\mathcal{O}(n)$ memops:

$$\text{(AXPY)} \quad y \leftarrow \alpha x + y, \quad x, y \in \mathbb{F}^n, \quad \alpha \in \mathbb{F},$$

Уровень 2

$\mathcal{O}(n^2)$ flops, $\mathcal{O}(n^2)$ memops:

$\frac{\text{flops}}{\text{memops}} = \text{const}$

$$\text{(MV)} \quad y \leftarrow \alpha A x + \beta y, \quad x, y \in \mathbb{F}^n, \quad A \in \mathbb{F}^{n \times n}, \quad \alpha, \beta \in \mathbb{F},$$

обращение треугольных матриц, ранг-1 апдейт матрицы, и т.д.

Уровень 3

$\mathcal{O}(n^3)$ flops, $\mathcal{O}(n^2)$ memops:

$$\text{(MM)} \quad C \leftarrow \alpha AB + \beta C, \quad A, B, C \in \mathbb{F}^{n \times n}, \quad \alpha, \beta \in \mathbb{F}$$

Надо стараться записывать алгоритмы через матричное произведение.

BLAS (Basic Linear Algebra Subprograms)

Оригинальная версия BLAS: 1979 год на fortran. С того времени переписан множество раз, но интерфейс функций стандартизован.

Разделяют 3 уровня операций в BLAS (далее $\mathbb{F} \in \{\mathbb{R}, \mathbb{C}\}$):

Уровень 1

$\mathcal{O}(n)$ flops, $\mathcal{O}(n)$ memops:

$$(\text{AXPY}) \quad y \leftarrow \alpha x + y, \quad x, y \in \mathbb{F}^n, \quad \alpha \in \mathbb{F},$$

Уровень 2

$\mathcal{O}(n^2)$ flops, $\mathcal{O}(n^2)$ memops:

$$(\text{MV}) \quad y \leftarrow \alpha A x + \beta y, \quad x, y \in \mathbb{F}^n, \quad A \in \mathbb{F}^{n \times n}, \quad \alpha, \beta \in \mathbb{F},$$

обращение треугольных матриц, ранг-1 апдейт матрицы, и т.д.

Уровень 3

$\mathcal{O}(n^3)$ flops, $\mathcal{O}(n^2)$ memops:

$\frac{\text{flops}}{\text{memops}} = \mathcal{O}(n)$ - хорошо, т.к. $t_{mem} \gg t_{flop}$

$$(\text{MM}) \quad C \leftarrow \alpha AB + \beta C, \quad A, B, C \in \mathbb{F}^{n \times n}, \quad \alpha, \beta \in \mathbb{F}$$

Надо стараться записывать алгоритмы через матричное произведение.

BLAS (Basic Linear Algebra Subprograms)

Названия операций

- DOT: $x^\top y$
- AXPY: $y \leftarrow \alpha x + y$
- MV: $y \leftarrow \alpha A x + \beta y$
- MM: $C \leftarrow \alpha A B + \beta C$
- R: $A \leftarrow \alpha x y^\top + A$ (добавить ранг-1)
- ...

Типы матриц

- GE – general
- GB – general band
- SY – symmetric
- SB – symm. band
- TR – triangular
- ...

Precision:

- S – single
- D – double
- C – single complex
- Z – double complex

Пример: ZGEMM (матрично-матричное умножение с произвольными плотными матрицами из комплексных чисел в двойной точности)

BLAS (Basic Linear Algebra Subprograms)

Релевантные пакеты программ

- LAPACK (Linear Algebra PACKage): матричные факторизации, решение линейных систем, SVD, ... Использует BLAS.
- Intel MKL (Math Kernel Library): оптимизованные под Intel процессоры BLAS и LAPACK.
- OpenBLAS: оптимизированный BLAS. Базируется на GotoBLAS, который долгое время был рекордсменом (K. Гото написал GotoBLAS во время саббатикала в 2002).
- ATLAS (Automatically Tuned Linear Algebra Software): автоматически оптимизирует BLAS под конкретную систему.
- cuBLAS (CUDA BLAS): имплементация для GPU от NVIDIA.

Линейно-алгебраические операции в scipy и numpy – врапперы для функций из BLAS и LAPACK. В Anaconda Python Distribution (версия 2.5 и старше) и MATLAB по умолчанию используется MKL.

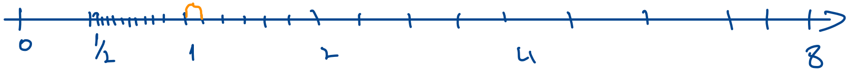
План

Машинные числа

$$\mathbb{FP} = \left\{ \pm \overbrace{\left(\frac{d_1}{b} + \frac{d_2}{b^2} + \dots + \frac{d_m}{b^m} \right)}^{\text{мантисса}} b^e, \quad d_i = 0, 1, \dots, b-1, \quad e_{\min} \leq e \leq e_{\max} \right\}$$

- $\mathbb{FP} \subset \mathbb{R}$ – конечное множество машинных чисел
- $b \in \mathbb{N}$ – основание (base) арифметики
- $m \in \mathbb{N}$ – длина мантиссы
- $e \in \mathbb{Z}$ – порядок (exponent) конкретного $x \in \mathbb{FP}$
- $d_i \in \{0, \dots, b-1\}$ – разряды числа $x \in \mathbb{FP}$

b^{1-m} – иногда называют машинным эпсилон
мы дадим другое опр.



Машинные числа

IEEE[4] стандарт 754

Был принят в 1985 для унификации представления чисел и операций с ними.

Точность	b	m	$e_{\max}$	$e_{\min}$
single	2	23	127	-126
double	2	52	1023	-1022
- Задает правило округления $\mathfrak{fl}\colon \mathbb{R} \to \mathbb{FP}$.
- Задает правило для разрешения неопределенностей. Например, для $\frac{0}{0}$.
- Арифметические и другие операции.
- ...

[4] Institute of Electrical and Electronics Engineers

Машинные числа: округление

Для операции округления

$$\mathfrak{fl}\colon \mathbb{R} \to \mathbb{FP}$$

можно записать

$$\mathfrak{fl}(x) = x(1+\epsilon), \quad |\epsilon| \le \epsilon_{\text{machine}},$$

где $\epsilon_{\text{machine}}$ (машинная эпсилон) — точная верхняя грань для $|\epsilon|$. То есть мы "привязали" определение $\epsilon_{\text{machine}}$ к $\mathfrak{fl}$[5].

Фундаментальная аксиома машинной арифметики

Для любых $x, y \in \mathbb{FP}$ существует $\epsilon : |\epsilon| \le \epsilon_{\text{machine}}$, такое что для любой операции op $\in \{+, -, \times, /\}$ выполняется:

$$\mathfrak{fl}(x \operatorname{op} y) = (x \operatorname{op} y)(1+\epsilon), \quad |\epsilon| \le \epsilon_{\text{machine}}$$

Для вычислений на компьютере, построенном по принципу этой аксиомы, будет удобно строить теоретический анализ ошибок округления.

[5] Для "школьного" $\mathfrak{fl}$ – отбрасывания лишних цифр (truncation), $\epsilon_{\text{machine}} = \frac{1}{2} b^{1-m}$. В литературе также встречается $\epsilon_{\text{machine}} = b^{1-m}$.

Устойчивость и обусловленность

Перейдем к обсуждению двух ключевых понятий численного анализа: обусловленность и устойчивость.

Важно помнить

1. Устойчивость определяется для алгоритма.
2. Обусловленность определяется для задачи.

План

Вычислительная устойчивость

конечномер., нормир.

Пусть задана задача $f\colon X \to Y$ и $\tilde{f}\colon X \to Y$ – некоторый алгоритм ее решения.

Прямая устойчивость

Алгоритм обладает свойством прямой устойчивости (forward stability), если

$\le C\, \varepsilon_{\text{machine}}$

$$\frac{\|\tilde{f}(x) - f(x)\|}{\|f(x)\|} = \mathcal{O}(\epsilon_{\text{machine}}).$$

Сложно анализировать (нужно следить за каждой операцией) и нужно учитывать, что при большой ошибке “плохой” может оказаться задача, а не алгоритм.

Обратная устойчивость

Алгоритм обладает свойством обратной устойчивости, если

$$\tilde{f}(x) = f(\tilde{x}) \text{ для некоторого } \tilde{x}\colon \frac{\|\tilde{x} - x\|}{\|x\|} = \mathcal{O}(\epsilon_{\text{machine}}).$$

То есть, мы хотим заменить вычисленную величину как точное вычисление с возмущенными данными. Подход удобен для анализа (обратный анализ ошибок). Будем использовать в след. лекциях.

Вычислительная устойчивость

(Смешанная) устойчивость

Алгоритм является устойчивым, если

$$\frac{\|\tilde{f}(x) - f(\tilde{x})\|}{\|f(\tilde{x})\|} = \mathcal{O}(\epsilon_{\text{machine}}) \text{ для некоторого } \tilde{x}\text{: } \frac{\|\tilde{x} - x\|}{\|x\|} = \mathcal{O}(\epsilon_{\text{machine}}).$$

Вычислительная устойчивость

$\boxed{\text{Пример}}$

$\tilde{x} + \tilde{y}$, где $\tilde{x} = x(1+\varepsilon)$, $\tilde{y} = y(1+\varepsilon)$ $\Rightarrow$ $\dfrac{|\tilde{x} - x|}{|x|} = |\varepsilon| \leq \varepsilon_{machine}$

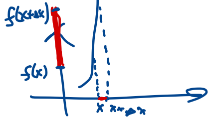
$\Uparrow$

backward stable

$$\frac{|x+y - \overbrace{(x+y)(1+\varepsilon)}^{\tilde{x}+\tilde{y}}|}{|x+y|} = |\varepsilon| \leq \varepsilon_{machine} \Rightarrow \text{forward stable}$$

План

Обусловленность



$$f : X \to Y$$

любая

$$f(x+\Delta x) = f(x) + f'(x)\Delta x + O(\|\Delta x\|^2)$$

$$\frac{\|f(x+\Delta x) - f(x)\|}{\|f(x)\|} \approx \frac{\|f'(x)\|\,\|x\|}{\|f(x)\|} \frac{\|\Delta x\|}{\|x\|}$$

$\| \text{def}$

число обусл. f – мера чувствит. задачи

$\text{cond}(f, x)$

$$\text{cond}_2(A) = \|A\|_2 \|A^{-1}\|_2$$

$$1 = \|I\| = \|AA^{-1}\| \leq \|A\|\,\|A^{-1}\|$$

$\boxed{\text{Пример}}$ $f(x) = Ax, \quad X = Y = \mathbb{C}^n$

$$f'(x) = A$$

$$\text{cond}(f,x) = \frac{\|A\|}{\|Ax\|}\|x\| = \frac{\|A\|}{\|Ax\|}\|A^{-1}Ax\| \leq \frac{\|A\|}{\cancel{\|Ax\|}}\|A^{-1}\|\,\cancel{\|Ax\|} =$$

$$= \|A\| \cdot \|A^{-1}\| = \underline{\text{cond}(A)}$$

число обусл. матрицы A

Обусловленность

$$= f(\tilde{x}), \quad \frac{\|\tilde{x} - x\|}{\|x\|} = O(\varepsilon_{machine})$$

$$\text{forward\_err} = \frac{\|\tilde{f}(x) - f(x)\|}{\|f(x)\|} = \frac{\|f(\tilde{x}) - f(x)\|}{\|f(x)\|} \le$$

$$\le \left(\mathrm{cond}(f, x) + \underbrace{O(\|\tilde{x} - x\|)}_{= O(\varepsilon_{machine})} \right) \frac{\|\tilde{x} - x\|}{\|x\|} \simeq$$

$$\simeq \mathrm{cond}(f, x) \cdot \text{backward\_err}$$

Литература

- N. Higham “Accuracy and Stability of Numerical Algorithms”, SIAM, 2002.
- Тыртышников, Е.Е. Матричный анализ и линейная алгебра, Москва, Физматлит, 2007. – 477 с
- Тыртышников Е. Е. Методы численного анализа. – Издательский центр Академия Москва, 2007. – 320 с.
- Trefethen, L. N., & Bau III, D. (1997). Numerical linear algebra. (Vol. 50). Siam. Philadelphia.