Dense Linear Algebra
—From Gauss to Strassen—

Alin Bostan

Specfun, Inria

MPRI C-2-22

September 28, 2020

The exercise from last week

Let $\mathbb{A}$ be a ring.

1. Estimate the number of multiplications in $\mathbb{A}$ needed by Karatsuba's algorithm to compute the product AB of any two polynomials A and B of degree at most 3 in $\mathbb{A}[X]$.
2. Let us assume that 2, 3, and 5 are invertible in $\mathbb{A}$ and that the divisions of elements of A by 2, 3, and 5 are free. Propose an algorithm that multiplies A and B of degree at most 3 using at most 7 multiplications in $\mathbb{A}$.
3. Let us assume that 2, 3, and 5 are invertible in $\mathbb{A}$. Propose an algorithm for polynomial product in $\mathbb{A}[X]$ whose arithmetic complexity is $O(n^{1,41})$.

In what follows, we assume that the ring $\mathbb{A}$ has characteristic zero.

4. Show that, for any integer $\alpha \geq 2$, there exists an algorithm for polynomial multiplication in $\mathbb{A}[X]$ whose arithmetic complexity is $O(n^{\log_\alpha(2\alpha-1)})$.
5. Show that for all $\varepsilon > 0$, there exists an algorithm for polynomial multiplication in $\mathbb{A}[X]$ whose arithmetic complexity is $O(n^{1+\varepsilon})$,where the implied constant in the $O(\cdot)$ depends on ε but not on n.

Solution, Q1.

1. Estimate the number of multiplications in $\mathbb{A}$ needed by Karatsuba's algorithm to compute the product AB of any two polynomials A and B of degree at most 3 in $\mathbb{A}[X]$.

▷ The number of multiplications $\mathsf{mulK}(n)$ used by Karatsuba's algorithm in degree $< n$ satisfies $\mathsf{mulK}(n) = 3 \cdot \mathsf{mulK}(\lceil n/2 \rceil)$, with the initial condition $\mathsf{mulK}(1) = 1$. Hence, $\mathsf{mulK}(2^s) = 3^s$ for all $s \geq 0$. In particular $\mathsf{mulK}(4) = 9$.

▷ More exactly, let $A = a_0 + a_1 x + a_2 x^2 + a_3 x^3$ and $B = b_0 + b_1 x + b_2 x^2 + b_3 x^3$. Then, Karatsuba's algorithm writes $A = A_0 + x^2 A_1$ and $B = B_0 + x^2 B_1$, with $A_0 = a_0 + a_1 x$, $A_1 = a_2 + a_3 x$ and $B_0 = b_0 + b_1 x$, $B_1 = b_2 + b_3 x$, then computes $AB = A_0 B_0 + x^2 (A_0 B_1 + A_1 B_0) + x^4 A_1 B_1$ by writing $A_0 B_1 + A_1 B_0 = (A_0 + A_1)(B_0 + B_1) - A_0 B_0 - A_1 B_1$. The 9 products in $\mathbb{A}$ are

$$a_0 b_0, a_1 b_1, (a_0 + a_1)(b_0 + b_1), \quad a_2 b_2, a_3 b_3, (a_2 + a_3)(b_2 + b_3),$$

$$(a_0 + a_2)(b_0 + b_2), (a_1 + a_3)(b_1 + b_3), (a_0 + a_1 + a_2 + a_3)(b_0 + b_1 + b_2 + b_3).$$

Solution, Q2.

2. Let us assume that 2, 3, and 5 are invertible in $\mathbb{A}$ and that the divisions of elements of A by 2, 3, and 5 are free. Propose an algorithm that multiplies A and B of degree at most 3 using at most 7 multiplications in $\mathbb{A}$.

▷ Recall: Karatsuba's algorithm in degree 1 can be interpreted as an evaluation-interpolation algorithm at $0, 1, \infty$.

▷ Idea: generalize this remark to higher degrees. Here $\deg(AB) \leq 6$, so 7 points are enough. Take for instance $\mathcal{E} := \{-3, -2, -1, 0, 1, 2, 3\}$. The algorithm computes $A(e), B(e)$ for $e \in \mathcal{E}$, then the products $A(e)B(e)$ and finally, it interpolates $C := AB$ from these values.

▷ Crucial remark: the evaluation/interpolation steps are for *free* w.r.t. ring multiplications. Indeed, only multiplications by integers arise, and they can be simulated by $+/-$.

```
> A:=add(a[k]*x^k, k=0..3):
> [seq(eval(A,x=i), i=-3..3)];
[a[0] - 3 a[1] + 9 a[2] - 27 a[3], a[0] - 2 a[1] + 4 a[2] - 8 a[3],
    a[0] - a[1] + a[2] - a[3], a[0], a[0] + a[1] + a[2] + a[3],
       a[0] + 2 a[1] + 4 a[2] + 8 a[3], a[0] + 3 a[1] + 9 a[2] + 27 a[3]]
```

▷ In the interpolation step, one computes similar linear combinations with integer coefficients, plus a division by $720 = 2^4 \cdot 3^2 \cdot 5$.

```
> P:=interp([seq(i,i=-3..3)],[seq(v[i],i=1..7)],x):
> denom(P);
                                720
> [seq(coeff(numer(P),x,k),k=0..6)];
[720 v[4], 12 v[7] - 108 v[6] + 540 v[5] - 540 v[3] - 12 v[1] + 108 v[2],
   4 v[7] - 54 v[6] + 540 v[5] + 540 v[3] + 4 v[1] - 980 v[4] - 54 v[2],
     -15 v[7] + 120 v[6] - 195 v[5] + 195 v[3] + 15 v[1] - 120 v[2],
       -5 v[7] + 60 v[6] - 195 v[5] - 195 v[3] - 5 v[1] + 280 v[4] + 60 v[2]
         3 v[7] - 12 v[6] + 15 v[5] - 15 v[3] - 3 v[1] + 12 v[2],
           v[7] - 6 v[6] + 15 v[5] + 15 v[3] + v[1] - 20 v[4] - 6 v[2]]
```

▷ As divisions by $2, 3, 5$ are assumed free, we conclude that 7 multiplications in $\mathbb{A}$ are enough to compute AB.

Solution, Q3.

3. Let us assume that 2, 3, and 5 are invertible in $\mathbb{A}$. Propose an algorithm for polynomial product in $\mathbb{A}[X]$ whose arithmetic complexity is $O(n^{1,41})$.

▷ We can use recursively the result from Q2. This yields a DAC (divide-and-conquer) algorithm that "cuts" input polynomials of degree $< n$ into 4 polynomials of degree $< \lceil n/4 \rceil$. Its complexity is driven by the recurrence $\mathsf{T}(n) = 7 \cdot \mathsf{T}(\lceil n/4 \rceil) + O(n)$, whose solution is $\mathsf{T}(n) = O(n^{\log_4(7)})$; as $\log_4(7) = 1{,}403677461\ldots$, the algorithm has complexity in $O(n^{1,41})$.

▷ Note that this improves on Karatsuba's algorithm.

▷ Remark that the assumption on free divisions by 2, 3, and 5 is not needed anymore. However, we still need the assumption that they are invertible. (We compute their inverses at the beginning of the algorithm.)

Solution, Q4.

In what follows, we assume that the ring $\mathbb{A}$ has characteristic zero.

4. Show that, for any integer $\alpha \geq 2$, there exists an algorithm for polynomial multiplication in $\mathbb{A}[X]$ whose arithmetic complexity is $O(n^{\log_\alpha(2\alpha-1)})$.

▷ If $A, B \in \mathbb{A}[x]$ have degrees at most $n = \alpha - 1$, then $C = AB$ has degree at most $2n = 2\alpha - 2 < 2\alpha - 1$. We choose $\mathcal{E} = \{-n, \ldots, n-1, n\}$ and compute C by evaluation-interpolation on the points of $\mathcal{E}$.

▷ Basic algorithm:

1. Compute the evaluations $A(e), B(e)$, for $e \in \mathcal{E}$.
2. Compute the products $v_e := A(e)B(e)$, for $e \in \mathcal{E}$.
3. Return the polynomial interpolating the values v_e at the points in $\mathcal{E}$.

▷ This algorithm performs $2n + 1 = 2\alpha - 1$ products in $\mathbb{A}$ and $O(n) = O(\alpha)$ scalar operations ($+$, $-$, and $\times$ of elements in $\mathbb{A}$ by precomputed scalars).

▷ Now, if A and B have degrees $< n$, for arbitrary n, one cuts them into α polynomials of degree $< \lceil n/\alpha \rceil$, (Karatsuba algo. corresponds to $\alpha = 2$), that one multiplies them using the basic algorithm, recursively.

▷ The complexity $\mathsf{T}(n)$ satisfies

$$\mathsf{T}(n) = (2\alpha - 1)\mathsf{T}(n/\alpha) + K(\alpha)n,$$

where $K(\alpha)$ is some function (with $K(2) = 4$). The solution is given by the DAC theorem (with $p = q = \alpha, s = 1, m = 2\alpha - 1$, so $q < m$, $T(x) = K(\alpha)x$ and $\kappa = 1$):

$$\mathsf{T}(n) = n^{\log_\alpha(2\alpha-1)} \left(1 + K(\alpha) \cdot \frac{\alpha}{\alpha - 1}\right) = O(n^{\log_\alpha(2\alpha-1)}),$$

where the constant in $O(\cdot)$ depends on α, but not on n.

Solution, Q5.

In what follows, we assume that the ring $\mathbb{A}$ has characteristic zero.

5. Show that for all $\varepsilon > 0$, there exists an algorithm for polynomial multiplication in $\mathbb{A}[X]$ whose arithmetic complexity is $O(n^{1+\varepsilon})$,where the implied constant in the $O(\cdot)$ depends on ε but not on n.

▷ The sequence $\log_\alpha(2\alpha - 1)$ tends to 1 :

$$\log_\alpha(2\alpha - 1) = 1 + \frac{\ln 2}{\ln \alpha} + O\left(\frac{1}{\alpha \ln a}\right), \qquad \alpha \to \infty$$

```
> [seq(evalf(log[a](2*a-1)),a=2..10)];
[1.584962501, 1.464973520, 1.403677461, 1.365212390, 1.338290833,
          1.318123223, 1.302296865, 1.289450961, 1.278753601]
```

▷ Thus, for all $\varepsilon > 0$, there exists $\alpha \in \mathbb{N}$ depending on $\varepsilon > 0$, such that $\log_\alpha(2\alpha - 1) < 1 + \varepsilon$.

▷ The preceding reasoning shows that there exists a multiplication algorithm in $\mathbb{A}[X]$ of complexity $O(n^{\log_\alpha(2\alpha-1)})$, where the constant in $O(\cdot)$ depends on α, but not on n. This algorithm has a complexity $O(n^{1+\varepsilon})$, where the constant in $O(\cdot)$ depends on ε, but not on n.

Dense Linear Algebra
—From Gauss to Strassen—

Alin Bostan

Specfun, Inria

MPRI C-2-22

September 28, 2020

Introduction

Context

▷ Customary philosophy in mathematics:
"a problem is trivialized when it is reduced to a linear algebra question"

▷ From a computational viewpoint, it is important to address *efficiency issues* of the various linear algebra operations

▷ The most fundamental problems in linear algebra:

- linear system solving $Ax = b$,
- computation of the inverse A^{-1} of a matrix A,
- computation of determinant, rank,
- computation of minimal polynomial, characteristic polynomial,
- computation of canonical forms (LU / LDU / LUP decompositions, echelon forms, Frobenius forms = block companion, ...),
- computation of row/column reduced forms.

Warnings

▷ Natural mathematical ideas may lead to highly inefficient algorithms! E.g., the definition of $\det(A)$, with exponential complexity in the size of A. Also, Cramer's formulas for system solving are not very useful in practice.

▷ In all what follows, we will work with a commutative effective field $\mathbb{K}$, and with the algebra $\mathcal{M}_n(\mathbb{K})$ of square matrices with entries in $\mathbb{K}$.

▷ NB: most results extend to the case where $\mathbb{K}$ is replaced by a commutative effective ring $\mathbb{A}$, and to rectangular (instead of square) matrices.

Gaussian elimination

Theorem 0

For any matrix $A \in \mathcal{M}_n(\mathbb{K})$, one can compute in $O(n^3)$ operations in $\mathbb{K}$:

1. the rank $\mathrm{rk}(A)$
2. the determinant $\det(A)$
3. the inverse A^{-1}, if A is invertible
4. a (vector/affine) solutions basis of $Ax = b$, for any b in $\mathbb{K}^n$
5. an LUP decomposition (L = unit lower triangular, U = upper triangular, P = permutation matrix)
6. an LDU decomposition (L/U = unit lower/upper triangular, D = diag)
7. a reduced row echelon form (Gauss-Jordan) of A.

▷ based on elementary row operations: (1) swapping rows; (2) multiplying rows by scalars; (3) adding a multiple of one row to another row.

Main messages

▷ One can do better than Gaussian elimination!

▷ There exists $2 \leq \omega < 3$, the "matrix multiplication exponent", which controls the complexity of all linear algebra operations.

▷ One can classify linear algebra algorithms in three categories:

- dense, without any structure (today): their manipulation boils down essentially to matrix multiplication $O(n^3) \rightarrow O(n^\omega)$, where $\omega < 2.38$
- sparse (lecture 6): algos based on linear recurrences: $O(n^3) \rightarrow \tilde{O}(n^2)$
- structured (Vandermonde, Sylvester, Toeplitz, Hankel,.., lecture 7): algorithms based on the theory of the displacement rank: $O(n^3) \rightarrow \tilde{O}(n)$

Applications

▷ Linear algebra is ubiquitous:

- computations with (dense and D-finite) power series (lecture 3)
- computation of terms of a recurrent sequence (lecture 5)
- Hermite-Padé approximants (lecture 6)
- symbolic integration and summation, Zeilberger's algorithm (lecture 9)
- solutions of linear differential equations (lecture 10)

⋮

- integer factorization (sparse, over $\mathbb{F}_2$) and polynomial factorization over finite fields (dense, for Berlekamp's and Shoup's algorithms)
- PageRank webpage ranking system relies on (sparse) linear algebra
- crypto-analysis: discrete logs (sparse)

Matrix multiplication

Matrix multiplication

Together with integer and polynomial multiplication, matrix multiplication is one of the most basic and most important operations in computer algebra.

Matrix-vector product

Theorem [Winograd'67]

The naive algorithm for multiplying a $m \times n$ *generic* matrix by a $n \times 1$ vector (using mn multiplications and $m(n-1)$ additions) *is optimal*.

▷ Natural question: is the naive matrix product in size n (using n^3 $\otimes$ and $n^3 - n^2$ $\oplus$) also optimal?

Complexity of matrix product: main results

Theorem 1 ["naive multiplication is not optimal"]

One can multiply two matrices $A, B \in \mathcal{M}_n(\mathbb{K})$ in:

1. $n^2\lceil\frac{n}{2}\rceil + 2n\lfloor\frac{n}{2}\rfloor \simeq \frac{1}{2}n^3 + n^2$ multiplications in $\mathbb{K}$ [Pan'66-Winograd'68]
2. $n^2\lceil\frac{n}{2}\rceil + (2n-1)\lfloor\frac{n}{2}\rfloor \simeq \frac{1}{2}n^3 + n^2 - \frac{n}{2}$ multiplications in $\mathbb{K}$ [Waksman'69]
3. $O(n^{\log_2 7}) \simeq O(n^{2.81})$ operations in $\mathbb{K}$ [Strassen 1969]
4. $O(n^{2.38})$ operations in $\mathbb{K}$ [Coppersmith-Winograd, 1990]
5. $O(n^{2.3728639})$ operations in $\mathbb{K}$ [Le Gall, 2014]

Exponent of matrix multiplication

Def. $\theta \in [2,3]$ is a *feasible exponent* for matrix multiplication over $\mathbb{K}$ if one can multiply any A, B in $\mathcal{M}_n(\mathbb{K})$ using $O(n^\theta)$ ops. in $\mathbb{K}$.

Def. Exponent of matrix multiplication $\omega = \inf\{\theta \mid \theta \text{ is a feasible exponent}\}$.

Def. $\mathsf{MM} : \mathbb{N} \to \mathbb{N}$ is a *matrix multiplication function* (for a field $\mathbb{K}$) if:

- one can multiply any A, B in $\mathcal{M}_n(\mathbb{K})$ using at most $\mathsf{MM}(n)$ ops. in $\mathbb{K}$
- MM satisfies $\mathsf{MM}(n) \leq \mathsf{MM}(2n)/4$ for all $n \in \mathbb{N}$
- $n \mapsto \mathsf{MM}(n)/n^2$ is increasing

▷ $\omega \in [2, 2.38]$

▷ if $\mathbb{K} \subset \mathbb{L}$ then $\omega_{\mathbb{K}} = \omega_{\mathbb{L}}$ [Schönhage'72], so $\omega_{\mathbb{K}}$ only depends on $\mathrm{char}(\mathbb{K})$

▷ Conjectured: ω does not depend on $\mathbb{K}$

▷ Big open problem: Is $\omega = 2$?

Mathematics > Algebraic Geometry

[Submitted on 23 Sep 2020]

Bad and good news for Strassen's laser method: Border rank of the 3x3 permanent and strict submultiplicativity

Austin Conner, Hang Huang, J. M. Landsberg

We determine the border ranks of tensors that could potentially advance the known upper bound for the exponent ω of matrix multiplication. The Kronecker square of the small $q = 2$ Coppersmith–Winograd tensor equals the 3×3 permanent, and could potentially be used to show $\omega = 2$. We prove the negative result for complexity theory that its border rank is 16, resolving a longstanding problem. Regarding its $q = 4$ skew cousin in $C^5 \otimes C^5 \otimes C^5$, which could potentially be used to prove $\omega \leq 2.11$, we show the border rank of its Kronecker square is at most 42, a remarkable sub-multiplicativity result, as the square of its border rank is 64. We also determine moduli spaces $\underline{VSP}$ for the small Coppersmith–Winograd tensors.

Subjects: **Algebraic Geometry (math.AG)**; Computational Complexity (cs.CC)
MSC classes: 68Q15, 15A69, 14L35
Cite as: arXiv:2009.11391 **[math.AG]**
(or arXiv:2009.11391v1 **[math.AG]** for this version)

Bibliographic data

[Enable Bibex (What is Bibex?)]

Submission history

From: J. M. Landsberg [view email]
[v1] Wed, 23 Sep 2020 21:40:15 UTC (41 KB)

Winograd's algorithm

Naive algorithm for $n = 2$

$$R = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \times \begin{bmatrix} x & y \\ z & t \end{bmatrix} = \begin{bmatrix} ax + bz & ay + bt \\ cx + dz & cy + dt \end{bmatrix}$$

requires $8 \otimes$ and $4 \oplus$

- Naive algorithm for arbitrary n requires $n^3 \otimes$ and $(n^3 - n^2) \oplus$

Winograd's idea (1967): Karatsuba-like scheme

$$R = \begin{bmatrix} (a + z)(b + x) - ab - zx & (a + t)(b + y) - ab - ty \\ (c + z)(d + x) - cd - zx & (c + t)(d + y) - cd - ty \end{bmatrix}$$

- Drawbacks: uses commutativity (e.g., $zb = bz$); not yet profitable for $n = 2$

Winograd's algorithm

Same idea for $n = 2k$: for $\ell := (a_1, \cdots, a_n)$ and $c := (x_1, \cdots, x_n)^T$

$$(\ell|c) = (a_1 + x_2)(a_2 + x_1) + \cdots + (a_{2k-1} + x_{2k})(a_{2k} + x_{2k-1}) - \sigma(\ell) - \sigma(c),$$

where $\sigma(\ell) := a_1a_2 + \cdots + a_{2k-1}a_{2k}$ and $\sigma(c) := x_1x_2 + \cdots + x_{2k-1}x_{2k}$

The element $r_{i,j}$ of $R = AX$ is the scalar product $(\ell_i|c_j)$, where $\ell_1, \ldots, \ell_n$ are the rows of A and $c_1, \ldots, c_n$ are the columns of X

Winograd's algorithm:

- precompute $\sigma(\ell_i)$ for $1 \leq i \leq n \longrightarrow \quad nk = \frac{n^2}{2} \otimes$ and $n(k-1) = \frac{n^2}{2} - n \oplus$
- precompute $\sigma(c_j)$ for $1 \leq j \leq n \longrightarrow \quad nk = \frac{n^2}{2} \otimes$ and $n(k-1) = \frac{n^2}{2} - n \oplus$
- compute all $r_{i,j} := (\ell_i|c_j) \longrightarrow n^2k = \frac{n^3}{2} \otimes$ and $n^2(n+k+1) = \frac{3n^3}{2} + n^2 \oplus$

▷ Total: $\frac{1}{2}n^3 + n^2 \otimes$ and $\frac{3}{2}n^3 + 2n^2 - 2n \oplus$

Waksman's algorithm

Idea for $n = 2$: write

$$R = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \times \begin{bmatrix} x & y \\ z & t \end{bmatrix} = \begin{bmatrix} ax + bz & ay + bt \\ cx + dz & cy + dt \end{bmatrix}$$

as

$$R = \frac{1}{2} \begin{bmatrix} (a+z)(b+x) - (a-z)(b-x) & (a+t)(b+y) - (a-t)(b-y) \\ (c+z)(d+x) - (c-z)(d-x) & (c+t)(d+y) - (c-t)(d-y) \end{bmatrix},$$

and observe that the sum of the 4 products in red is equal to the sum of the 4 products in blue (and equal to $ab + zx + cd + ty$)

- 2×2 matrix product in 7 commutative $\otimes$, when $\mathrm{char}(\mathbb{K}) \neq 2$
- Idea generalizes to $n \times n$ matrices $\longrightarrow \frac{1}{2}n^3 + n^2 - \frac{n}{2}$ $\otimes$ for even n

Winograd/Waksman: summary

▷ They have cubic complexity, but are nevertheless useful in several contexts, e.g. products of small matrices containing large integers

▷ They already show that naive multiplication is not optimal

▷ Their weakness is the use of commutativity of the base ring, which does not allow a recursive use on blocks

▷ Natural question: can we do 7 non-commutative $\otimes$?

Matrix multiplication

Strassen's algorithm

Matrix multiplication
Strassen's algorithm

Strassen was attempting to prove, by process of elimination, that such an algorithm did not exist when he arrived at it.

"First I had realized that an estimate tensor rank < 8 for two by two matrix multiplication would give an asymptotically faster algorithm. Then I worked over $\mathbb{Z}/2\mathbb{Z}$ (as far as I remember) to simplify matters."

Strassen's matrix multiplication algorithm

Same idea as for Karatsuba's algorithm: trick in low size + recursion

Additional difficulty: Formulas should be non-commutative

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \times \begin{bmatrix} x & y \\ z & t \end{bmatrix} \iff \begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix}$$

Crucial remark: If $\varepsilon \in \{0, 1\}$ and $\alpha \in \mathbb{K}$, then 1 multiplication suffices for $E \cdot v$, where v is a vector, and E is a matrix of one of the following types:

$$\begin{bmatrix} \alpha & \alpha \\ \varepsilon\alpha & \varepsilon\alpha \end{bmatrix}, \begin{bmatrix} \alpha & -\alpha \\ \varepsilon\alpha & -\varepsilon\alpha \end{bmatrix}, \begin{bmatrix} \alpha & \varepsilon\alpha \\ -\alpha & -\varepsilon\alpha \end{bmatrix}$$

Strassen's matrix multiplication algorithm

Problem: Write

$$M = \begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix}$$

as a sum of less than 8 elementary matrices.

$$M - \underbrace{\begin{bmatrix} a & a & & \\ a & a & & \\ & & & \\ & & & \end{bmatrix}}_{E_1} - \underbrace{\begin{bmatrix} & & & \\ & & & \\ & & d & d \\ & & d & d \end{bmatrix}}_{E_2} = \begin{bmatrix} & b-a & & \\ c-a & d-a & & \\ & & a-d & b-d \\ & & c-d & \end{bmatrix}$$

Strassen's matrix multiplication algorithm

Problem: Write

$$M = \begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix}$$

as a sum of less than 8 elementary matrices.

$$M - E_1 - E_2 = \underbrace{\begin{bmatrix} & & & \\ & d-a & a-d & \\ & d-a & a-d & \\ & & & \end{bmatrix}}_{E_3} + \begin{bmatrix} & b-a & & \\ c-a & & d-a & \\ & a-d & & b-d \\ & & c-d & \end{bmatrix}$$

Strassen's matrix multiplication algorithm

Problem: Write

$$M = \begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix}$$

as a sum of less than 8 elementary matrices.

$$M - E_1 - E_2 - E_3 = \begin{bmatrix} & b-a & & \\ & & & \\ & a-d & & b-d \\ & & & \end{bmatrix} + \begin{bmatrix} c-a & & d-a & \\ & & & \\ & & & c-d \\ & & & \end{bmatrix}$$

Strassen's matrix multiplication algorithm

Problem: Write

$$M = \begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix}$$

as a sum of less than 8 elementary matrices.

$$M - E_1 - E_2 - E_3 = \underbrace{\begin{bmatrix} & b-a & & \\ & & & \\ (b-d)-(b-a) & & & b-d \end{bmatrix}}_{E_5} + \underbrace{\begin{bmatrix} c-a & & (c-a)-(c-d) & \\ & & & \\ & & & c-d \end{bmatrix}}_{E_6 + E_7}$$

Strassen's matrix multiplication algorithm

Problem: Write

$$M = \begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix}$$

as a sum of less than 8 elementary matrices.

Conclusion

$$M = E_1 + E_2 + E_3 + E_4 + E_5 + E_6 + E_7$$

$\Longrightarrow$ one can multiply 2×2 matrices using 7 non-comm products instead of 8

DAC Theorem:

$\mathsf{MM}(r) = 7 \cdot \mathsf{MM}(r/2) + O(r^2) \quad \Longrightarrow \quad \mathsf{MM}(r) = O(r^{\log_2(7)}) = O(r^{2.81})$

$$\underbrace{\begin{bmatrix} a & a & \cdot & \cdot \\ a & a & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}}_{E_1} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix} = \begin{bmatrix} a(x+z) \\ a(x+z) \\ \cdot \\ \cdot \end{bmatrix}, \underbrace{\begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & d & d \\ \cdot & \cdot & d & d \end{bmatrix}}_{E_2} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix} = \begin{bmatrix} \cdot \\ \cdot \\ d(y+t) \\ d(y+t) \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & d-a & a-d & \cdot \\ \cdot & d-a & a-d & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}}_{E_3} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix} = \begin{bmatrix} \cdot \\ (d-a)(z-y) \\ (d-a)(z-y) \\ \cdot \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & b-d & \cdot & b-d \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}}_{E_4} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix} = \begin{bmatrix} \cdot \\ \cdot \\ (b-d)(z+t) \\ \cdot \end{bmatrix}, \underbrace{\begin{bmatrix} \cdot & b-a & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & -(b-a) & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}}_{E_5} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix} = \begin{bmatrix} (b-a)z \\ \cdot \\ -(b-a)z \\ \cdot \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ c-a & \cdot & c-a & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}}_{E_6} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix} = \begin{bmatrix} \cdot \\ (c-a)(x+y) \\ \cdot \\ \cdot \end{bmatrix}, \underbrace{\begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & -(c-d) & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & c-d & \cdot \end{bmatrix}}_{E_7} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix} = \begin{bmatrix} \cdot \\ -(c-d)y \\ \cdot \\ (c-d)y \end{bmatrix}$$

▷ In summary, 7 $\otimes$ (non-comm.) and 18 $\oplus$:

$$\begin{bmatrix} a & b & 0 & 0 \\ c & d & 0 & 0 \\ 0 & 0 & a & b \\ 0 & 0 & c & d \end{bmatrix} \times \begin{bmatrix} x \\ z \\ y \\ t \end{bmatrix} = \begin{bmatrix} a(x+z)+(b-a)z \\ a(x+z)+(d-a)(z-y)+(c-a)(x+y)-(c-d)y \\ d(y+t)+(d-a)(z-y)+(b-d)(z+t)-(b-a)z \\ d(y+t)+(c-d)y \end{bmatrix}$$

▷ Extension: $n^3 - n(n-1)/2$ non-comm. $\otimes$ for $n \times n$ [Fiduccia'72]

▷ 7 non-comm. $\otimes$ and 15 $\oplus$ [Winograd'71] (instead of 18 $\oplus$ for [Strassen'69])

▷ Optimality: [Winograd'71], [Hopcroft & Kerr'71] (7 $\otimes$); [Probert'73] (15 $\oplus$)

Input Two matrices $A, X \in \mathcal{M}_n(\mathbb{K})$, with $n = 2^k$.

Output The product AX.

1. If $n = 1$, return AX.
2. Write $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, $X = \begin{bmatrix} x & y \\ z & t \end{bmatrix}$, with $a, b, c, d, x, y, z, t \in \mathcal{M}_{n/2}(\mathbb{K})$.
3. Compute recursively the products

$$\begin{aligned} q_1 &= a(x+z), & q_2 &= d(y+t), \\ q_3 &= (d-a)(z-y), & q_4 &= (b-d)(z+t) \\ q_5 &= (b-a)z, & q_6 &= (c-a)(x+y), \quad q_7 = (c-d)y. \end{aligned}$$

4. Compute the sums

$$\begin{aligned} r_{1,1} &= q_1 + q_5, & r_{1,2} &= q_2 + q_3 + q_4 - q_5, \\ r_{2,1} &= q_1 + q_3 + q_6 - q_7, & r_{2,2} &= q_2 + q_7. \end{aligned}$$

5. Return $\begin{bmatrix} r_{1,1} & r_{1,2} \\ r_{2,1} & r_{2,2} \end{bmatrix}$.

In practice

▷ in a good implementation, Winograd & Waksman algorithms are interesting for small sizes

▷ Strassen's algorithm then becomes the best for $n \approx 64$

▷ Kaporin's algorithm becomes the best for $n \approx 500$

▷ best practical algorithm is [Kaporin'04]: it uses $n^3/3 + 4n^2 + 8n$ non-comm. $\otimes$ in size n. Choosing $n = 48$ leads to $O(n^{\log_{48}(46464)}) = O(n^{2.776})$

▷ the vast majority of the other algorithms rely on techniques that are two complex, and that implies very big constants in the $O(\cdot) \longrightarrow$ interesting for sizes over millions or billions

▷ magma is one of the few CAS that uses fast matrix multiplication

Other linear algebra problems

Complexity of linear algebra: main results

Theorem 2 ["Gaussian elimination is not optimal"]

Let θ be a feasible exponent for matrix multiplication in $\mathcal{M}_n(\mathbb{K})$. Then, one can compute:

1. the inverse A^{-1} and the determinant $\det(A)$ of $A \in \mathrm{GL}_n(\mathbb{K})$ [Strassen'69]
2. the solution of $Ax = b$ for any $A \in \mathrm{GL}_n(\mathbb{K})$ and $x \in \mathbb{K}^n$ [Strassen'69]
3. the LUP and LDU decompositions of A [Bunch & Hopcroft'74]
4. the rank $\mathrm{rk}(A)$ and an echelon form [Schönhage'72]
5. the characteristic polynomial $\chi_A(x)$ and the minimal polynomial $\mu_A(x)$ [Keller-Gehrig'85]

using $\tilde{O}(n^\theta)$ operations in $\mathbb{K}$.

Complexity of linear algebra: main results

Theorem 3 ["equivalence of linear algebra problems"]

The following problems on matrices in $\mathcal{M}_n(\mathbb{K})$

- multiplication
- inversion
- determinant
- characteristic polynomial
- LUP decomposition for matrices of full rank

all have the same asymptotic complexity, up to logarithmic factors.

In other words, the exponent ω controls the complexity of all these problems:

$$\omega = \omega_{\mathsf{inv}} = \omega_{\mathsf{det}} = \omega_{\mathsf{charpoly}} = \omega_{\mathsf{LUP}}$$

▷ Open: are ω_{solve} and ω_{rank} and $\omega_{\mathsf{isinvertible}}$ also equal to ω?

Inversion is not harder than multiplication

▷ [Strassen'69] showed how to reduce matrix inversion (and also linear system solving) to matrix multiplication

▷ His result is: one can invert a (generic) $n \times n$ matrix in $O(n^\theta)$ ops.

$\longrightarrow$ "Gauss elimination is not optimal"

▷ [Klyuyev & Kokovkin-Shcherbak'65] had previously proven that Gaussian elimination *is optimal* if one restricts to row and column operations.

▷ Strassen's method is a *Gaussian elimination by blocks*, applied recursively

▷ His algo requires 2 inversions,6 multiplications and 2 additions, in size $\frac{n}{2}$:

$$\mathsf{I}(n) \leq 2\mathsf{I}(n/2) + 6\mathsf{MM}(n/2) + n^2/2 \leq 3\sum_i 2^i \cdot \mathsf{MM}(n/2^i) + O(n^2) = O(\mathsf{MM}(n))$$

Inversion of dense matrices

- Starting point is the (non commutative!) identity ($a, b, c, d \in \mathbb{K}^\star$)

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ ca^{-1} & 1 \end{bmatrix} \times \begin{bmatrix} a & 0 \\ 0 & z \end{bmatrix} \times \begin{bmatrix} 1 & a^{-1}b \\ 0 & 1 \end{bmatrix},$$

where $z = d - ca^{-1}b$ is the *Schur complement* of a in M.

- This identity (LDU decomposition) is a consequence of Gauss pivoting on M
- It follows the matrix factorization of the inverse of M:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \begin{bmatrix} 1 & -a^{-1}b \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} a^{-1} & 0 \\ 0 & z^{-1} \end{bmatrix} \times \begin{bmatrix} 1 & 0 \\ -ca^{-1} & 1 \end{bmatrix}$$

$$= \begin{bmatrix} a^{-1} + a^{-1}bz^{-1}ca^{-1} & -a^{-1}bz^{-1} \\ -z^{-1}ca^{-1} & z^{-1} \end{bmatrix}.$$

- This identity being non-commutative, it holds for matrices a, b, c, d

Inversion of dense matrices

[Strassen, 1969]

To invert a dense matrix $M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \in \mathcal{M}_n(\mathbb{K})$, with $A, B, C, D \in \mathcal{M}_{\frac{n}{2}}(\mathbb{K})$

0. If $n = 1$, then return M^{-1}.
1. Invert A (recursively): $E := A^{-1}$.
2. Compute the Schur complement: $Z := D - CEB$.
3. Invert Z (recursively): $T := Z^{-1}$.
4. Recover the inverse of M as

$$M^{-1} := \begin{bmatrix} E + EBTCE & -EBT \\ -TCE & T \end{bmatrix}.$$

DAC Theorem: $\mathsf{I}(n) = 2 \cdot \mathsf{I}\left(\frac{n}{2}\right) + O(\mathsf{MM}(n)) \quad \Longrightarrow \quad \mathsf{I}(n) = O(\mathsf{MM}(n))$

Corollary: inversion M^{-1} and system solving $M^{-1}b$ in time $O(\mathsf{MM}(n))$

Determinant of dense matrices

[Strassen, 1969]

To compute $\det(M)$ for $M = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \in \mathcal{M}_n(\mathbb{K})$, with $A, B, C, D \in \mathcal{M}_{\frac{n}{2}}(\mathbb{K})$

0. If $n = 1$, then return M.
1. Compute $E := A^{-1}$ and (recursively) $d_A := \det(A)$.
2. Compute the Schur complement: $Z := D - CEB$.
3. Compute $T := Z^{-1}$ and (recursively) $d_Z := \det(Z)$.
4. Recover the determinant $\det(M)$ as $d_A \cdot d_Z$.

DAC Theorem:

$$\mathsf{D}(n) = 2 \cdot \mathsf{D}\left(\tfrac{n}{2}\right) + 2 \cdot \mathsf{I}\left(\tfrac{n}{2}\right) + O(\mathsf{MM}(n)) \quad \Longrightarrow \quad \mathsf{C}(n) = O(\mathsf{MM}(n))$$

Corollary: Determinant $\det(M)$ in time $O(\mathsf{MM}(n))$

Multiplication is not harder than inversion

[Munro, 1973]

Let A and B two $n \times n$ matrices. To compute $C = AB$, set

$$D = \begin{bmatrix} I_n & A & 0 \\ 0 & I_n & B \\ 0 & 0 & I_n \end{bmatrix}.$$

Then the following identity holds:

$$D^{-1} = \begin{bmatrix} I_n & -A & AB \\ 0 & I_n & -B \\ 0 & 0 & I_n \end{bmatrix}$$

Thus $n \times n$ multiplication reduces to inversion in size $3n \times 3n$: $\omega_{\text{mul}} \leq \omega_{\text{inv}}$.

Exercise. Let $\mathsf{T}(n)$ be the complexity of multiplication of $n \times n$ lower triangular matrices. Show that one can multiply $n \times n$ matrices in $O(\mathsf{T}(n))$ ops.

Computation of characteristic polynomial

[Keller-Gehrig, 1985]

▷ Assume $A \in \mathcal{M}_n(\mathbb{K})$ generic, in particular $\chi_A := \det(xI_n - A)$ irred. in $\mathbb{K}[x]$

▷ This implies $\chi_A(x) = \mu_A(x)$ and $\mathcal{B} := \{A, Av, \ldots, A^{n-1}v\}$ basis of $\mathbb{K}^n$

Lemma. If $v \in \mathbb{K}^n \setminus \{0\}$, then $P := \left[v|Av|\cdots|A^{n-1}v\right]$ is invertible and $C := P^{-1}AP$ is in companion form

Proof. If $\chi_A(x) = x^n - p_{n-1}x^{n-1} - \cdots - p_1x - p_0$, then the matrix C of $f : w \mapsto Aw$ w.r.t. $\mathcal{B}$ is companion, with last column $[p_0, \ldots, p_{n-1}]^T$.

Algorithm.

- Compute the matrix $P := [A|Av|\cdots|A^{n-1}v]$ $O(n^?)$
- Compute the inverse $M := P^{-1}$ $O(n^\theta)$
- Return the last column of MAP $O(n^\theta)$

Computation of characteristic polynomial

[Keller-Gehrig, 1985]

▷ Remaining task: fast computation of the *Krylov sequence*

$$\{v, Av, \ldots, A^{n-1}v\}$$

▷ Naive algorithm: $v \xrightarrow{A\cdot} Av \xrightarrow{A\cdot} A^2v \xrightarrow{A\cdot} \cdots \xrightarrow{A\cdot} A^{n-1}v$ $O(n^3)$

▷ Keller-Gehrig algorithm:

1. Compute $A_0 := A$, $A_k := A_{k-1}^2$ for $k \geq 1$ by binary powering $O(n^\theta \log(n))$
2. Compute $[A^{2^k}v|\cdots|A^{2^{k+1}-1}v] := A_k \times [A^{2^k}v|\cdots|A^{2^k-1}v]$ $O(n^\theta \log(n))$

▷ Conclusion: Krylov sequence and thus $\chi_A(x)$ in $O(n^\theta \log(n))$

The Keller-Gehrig algorithm

Input A matrix $A \in \mathcal{M}_n(\mathbb{K})$, with $n = 2^k$.

Output Its characteristic polynomial $\chi_A(x) = \det(xI_n - A)$.

1. Choose v in $\mathbb{K}^n \setminus \{0\}$.
2. Set $M := A$ and $P := v$.
3. For i from 1 to k, replace P by the horizontal concatenation of P and MP, then M by M^2.
4. Compute $C := P^{-1}AP$ and let $[p_0, \ldots, p_{n-1}]^T$ be its last column.
5. Return $x^n - p_{n-1}x^{n-1} - \cdots - p_0$.

Two exercises for next time

(1) Let $\mathsf{T}(n)$ be the complexity of multiplication of $n \times n$ lower triangular matrices. Show that one can multiply $n \times n$ matrices in $O(\mathsf{T}(n))$ ops.

(2) Let $P \in \mathbb{K}[x]$ be of degree at most n and $\theta > 2$ be a feasible exponent for matrix multiplication in $\mathcal{M}_n(\mathbb{K})$.

(a) Find an algorithm for the simultaneous evaluation of P in $\lceil \sqrt{n} \rceil$ elements of $\mathbb{K}$ using $O(n^{\theta/2})$ in $\mathbb{K}$.

(b) If $A \in \mathcal{M}_n(\mathbb{K})$, show that one can compute $P(A)$ in $O(n^{\theta+1/2})$ ops. in $\mathbb{K}$.

(c) If $Q \in \mathbb{K}[X]$ is another polynomial of degree at most n, show that one can compute the first n coefficients of $P(Q(x))$ using $O(n^{\frac{\theta+1}{2}})$ ops. in $\mathbb{K}$.

▷ Hint: Write $P(x)$ as $P_0(x) + P_1(x)x^d + P_2(x)(x^d)^2 + \cdots$, where d is well-chosen and the $P_i(x)$'s have degrees less than d.