Fast and accurate subcubic matrix multiplication

Clément Pernet joint work with Jean-Guillaume Dumas Alexandre Sedoglavic

Workshop on Approximate Computing and Numerical Linear Algebra,
Paris, France. October 9, 2025

Grenoble INP – UGA, Université Grenoble Alpes, Université de Lille, France

Subcubic matrix multiplication algorithms and a long lasting defiance

2×2 matrix multiplication	# Multiplications	# Additions
Classic $O(n^3)$	8	4
[Strassen'69]	7	18
[Winograd'70]	7	15

Subcubic matrix multiplication algorithms and a long lasting defiance

2×2 matrix multiplication	# Multiplications	# Additions	Divide & Conquer
Classic $O(n^3)$	8	4	$2n^3$
[Strassen'69]	7	18	$7n^{2.8074}$
[Winograd'70]	7	15	$6n^{2.8074}$

Subcubic matrix multiplication algorithms and a long lasting defiance

2×2 matrix multiplication	# Multiplications	# Additions	Divide & Conquer
Classic $O(n^3)$	8	4	$2n^3$
[Strassen'69]	7	18	$7n^{2.8074}$
[Winograd'70]	7	15	$6n^{2.8074}$

Speed ? better exponent but worse leading constant $\rightsquigarrow$ only for large n

Subcubic matrix multiplication algorithms and a long lasting defiance

2×2 matrix multiplication	# Multiplications	# Additions	Divide & Conquer
Classic $O(n^3)$	8	4	$2n^3$
[Strassen'69]	7	18	$7n^{2.8074}$
[Winograd'70]	7	15	$6n^{2.8074}$

Speed ? better exponent but worse leading constant $\rightsquigarrow$ only for large n

[Knuth, The Art of Computer Programming vol.2]

advantageous for $n > 20$, and saved 18 percent when $n = 100$. He estimated that Strassen's scheme (36) would not begin to excel over (35) until $n \approx 250$; and such enormous matrices rarely occur in practice unless they are very sparse, when other techniques apply. Furthermore, the known methods of order n^ω

Subcubic matrix multiplication algorithms and a long lasting defiance

2×2 matrix multiplication	# Multiplications	# Additions	Divide & Conquer
Classic $O(n^3)$	8	4	$2n^3$
[Strassen'69]	7	18	$7n^{2.8074}$
[Winograd'70]	7	15	$6n^{2.8074}$

Speed ? better exponent but worse leading constant $\rightsquigarrow$ only for large n

Unstability: aggregation-cancelation, coefficient growth $\rightsquigarrow$ numerically unstable

Subcubic matrix multiplication algorithms and a long lasting defiance

2×2 matrix multiplication	# Multiplications	# Additions	Divide & Conquer
Classic $O(n^3)$	8	4	$2n^3$
[Strassen'69]	7	18	$7n^{2.8074}$
[Winograd'70]	7	15	$6n^{2.8074}$

Speed ? better exponent but worse leading constant $\rightsquigarrow$ only for large n

Unstability: aggregation-cancelation, coefficient growth $\rightsquigarrow$ numerically unstable

[ATLAS BLAS mailing list]

```
>>And anybody knows about other implementations as MKL, ACML or Goto(Open)BLAS?
>
> They should all be using the standard algorithm.  Strassen (and all the other
> fast matmuls) are illegal in standard libraries because they are not as
> numerically stable.  There are some high performance libraries that optionally
> provide fast/unstable multiplies (in particular 3-M for complex), but they
> aren't supposed to do so by default.
```

Subcubic matrix multiplication algorithms and a long lasting defiance

[Huss-Lederman, Jacobson, Johnson, Tsao, Turnbull'96]

Strassen's algorithm has long suffered from the erroneous assumptions that it is not efficient for matrix sizes that are seen in practice and that it is unstable. Both of these assumptions have been questioned in recent work. By stopping the Strassen recursions early

Subcubic matrix multiplication algorithms and a long lasting defiance

[Cormen-Leiserson-Rivest-Stein, Introduction to Algorithms]

From a practical point of view, Strassen's algorithm is often not the method of choice for matrix multiplication, for four reasons:

1. The constant factor hidden in the $\Theta(n^{\lg 7})$ running time of Strassen's algorithm is larger than the constant factor in the $\Theta(n^3)$-time SQUARE-MATRIX-MULTIPLY procedure.
2. When the matrices are sparse, methods tailored for sparse matrices are faster.
3. Strassen's algorithm is not quite as numerically stable as SQUARE-MATRIX-MULTIPLY. In other words, because of the limited precision of computer arithmetic on noninteger values, larger errors accumulate in Strassen's algorithm than in SQUARE-MATRIX-MULTIPLY.
4. The submatrices formed at the levels of recursion consume space.

Subcubic matrix multiplication algorithms and a long lasting defiance

[Cormen-Leiserson-Rivest-Stein, Introduction to Algorithms]

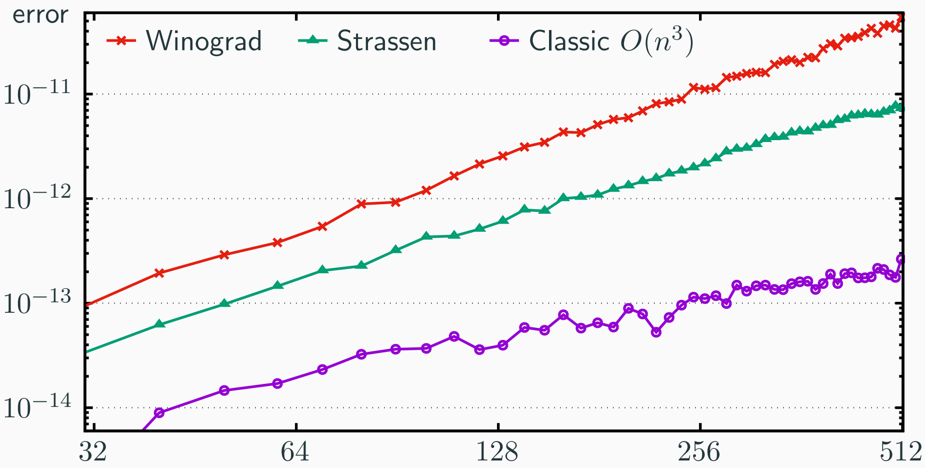
From a practical point of view, Strassen's algorithm is often not the method of choice for matrix multiplication, for four reasons:

1. The constant factor hidden in the $\Theta(n^{\lg 7})$ running time of Strassen's algorithm is larger than the constant factor in the $\Theta(n^3)$-time SQUARE-MATRIX-MULTIPLY procedure.
2. When the matrices are sparse, methods tailored for sparse matrices are faster.
3. Strassen's algorithm is not quite as numerically stable as SQUARE-MATRIX-MULTIPLY. In other words, because of the limited precision of computer arithmetic on noninteger values, larger errors accumulate in Strassen's algorithm than in SQUARE-MATRIX-MULTIPLY.
4. The submatrices formed at the levels of recursion consume space.

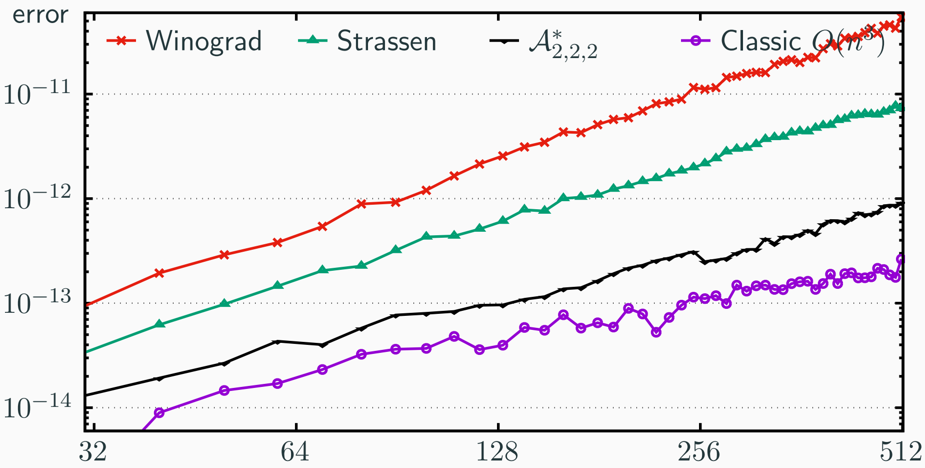
Debunking in progress

1. [Schwarz et al.] and [here] $\rightsquigarrow 5n^{\log_2 7}$ [here] $\frac{189}{16}n^{\log_4 48} \approx 11.9n^{2.793}$
3. [Brent 70], [Bini Lotti'80], [Demmel'93], [Higham'02], [Demmel et al.'07] ... and [here]
4. Pebble games (or [Dumas Grenet'24] $\rightsquigarrow 8n^{\log_2 7}$ fully inplace)

Accuracy of recursive 2×2 matrix multiplication algorithms



Contribution: new algorithms with improved accuracy and leading constant



Fast Matrix Multiplication algorithms and their representation

Bilinear program

$$\rho_1 \leftarrow a_{11} \cdot b_{11}$$
$$\rho_2 \leftarrow a_{12} \cdot b_{21}$$
$$\rho_3 \leftarrow (-a_{11} - a_{12} + a_{21} + a_{22}) \cdot b_{22}$$
$$\rho_4 \leftarrow a_{22} \cdot (-b_{11} + b_{12} + b_{21} - b_{22})$$
$$\rho_5 \leftarrow (a_{21} + a_{22}) \cdot (-b_{11} + b_{12})$$
$$\rho_6 \leftarrow (-a_{11} + a_{21}) \cdot (b_{12} - b_{22})$$
$$\rho_7 \leftarrow (-a_{11} + a_{21} + a_{22}) \cdot (-b_{11} + b_{12} - b_{22})$$

Fast Matrix Multiplication algorithms and their representation

Bilinear program

$$\rho_1 \leftarrow a_{11} \cdot b_{11}$$
$$\rho_2 \leftarrow a_{12} \cdot b_{21}$$
$$\rho_3 \leftarrow (-a_{11} - a_{12} + a_{21} + a_{22}) \cdot b_{22}$$
$$\rho_4 \leftarrow a_{22} \cdot (-b_{11} + b_{12} + b_{21} - b_{22})$$
$$\rho_5 \leftarrow (a_{21} + a_{22}) \cdot (-b_{11} + b_{12})$$
$$\rho_6 \leftarrow (-a_{11} + a_{21}) \cdot (b_{12} - b_{22})$$
$$\rho_7 \leftarrow (-a_{11} + a_{21} + a_{22}) \cdot (-b_{11} + b_{12} - b_{22})$$

Equivalent L, R, P representation

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & -1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 1 & 1 \end{bmatrix} \quad R = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ -1 & 1 & 0 & -1 \end{bmatrix}$$

Fast Matrix Multiplication algorithms and their representation

Bilinear program

$$\rho_1 \leftarrow a_{11} \cdot b_{11}$$
$$\rho_2 \leftarrow a_{12} \cdot b_{21}$$
$$\rho_3 \leftarrow (-a_{11} - a_{12} + a_{21} + a_{22}) \cdot b_{22}$$
$$\rho_4 \leftarrow a_{22} \cdot (-b_{11} + b_{12} + b_{21} - b_{22})$$
$$\rho_5 \leftarrow (a_{21} + a_{22}) \cdot (-b_{11} + b_{12})$$
$$\rho_6 \leftarrow (-a_{11} + a_{21}) \cdot (b_{12} - b_{22})$$
$$\rho_7 \leftarrow (-a_{11} + a_{21} + a_{22}) \cdot (-b_{11} + b_{12} - b_{22})$$

$$\begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \begin{bmatrix} \rho_1 + \rho_2 & \rho_1 - \rho_3 + \rho_5 - \rho_7 \\ \rho_1 + \rho_4 + \rho_6 - \rho_7 & \rho_1 + \rho_5 + \rho_6 - \rho_7 \end{bmatrix}$$

Equivalent L, R, P representation

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & -1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 1 & 1 \end{bmatrix} \quad R = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & 1 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ -1 & 1 & 0 & -1 \end{bmatrix}$$

$$P = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 & -1 \\ 1 & 0 & 0 & 1 & 0 & 1 & -1 \\ 1 & 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix}$$

Transformations of a Fast Matrix Multiplication Algorithm

Isotropies : exploiting symmetries of the matrix multiplication tensor

$$
\begin{array}{c}
[\mathsf{L};\mathsf{R};\mathsf{P}] \text{ an } n \times n \text{ Matrix Multiplication Representation} \\
\Big\downarrow \ \diamond\,(\mathsf{U},\mathsf{V},\mathsf{W}) \in \mathrm{SL}^{\pm}(n,\mathbb{K})^3 \\
\left[\mathsf{L}\cdot\left(\mathsf{V}^\intercal\otimes\mathsf{U}^{-1}\right);\mathsf{R}\cdot\left(\mathsf{W}^\intercal\otimes\mathsf{V}^{-1}\right);\left(\mathsf{U}\otimes\mathsf{W}^{-\intercal}\right)\cdot\mathsf{P}\right]
\end{array}
$$

Transformations of a Fast Matrix Multiplication Algorithm

Isotropies : exploiting symmetries of the matrix multiplication tensor

$[\mathsf{L};\mathsf{R};\mathsf{P}]$ an $n \times n$ Matrix Multiplication Representation

$$\Big\downarrow \diamond\, (\mathsf{U},\mathsf{V},\mathsf{W}) \in \mathrm{SL}^{\pm}(n,\mathbb{K})^3$$

$$\left[\mathsf{L}\cdot\left(\mathsf{V}^{\intercal}\otimes\mathsf{U}^{-1}\right);\mathsf{R}\cdot\left(\mathsf{W}^{\intercal}\otimes\mathsf{V}^{-1}\right);\left(\mathsf{U}\otimes\mathsf{W}^{-\intercal}\right)\cdot\mathsf{P}\right]$$

[de Groote'78]: All 2×2 matrix products with 7 multiplications lie in the same orbit

$\rightsquigarrow$ usefull for exploring matrix product algorithms in 7 multiplications

Transformations of a Fast Matrix Multiplication Algorithm

Sparsification via alternative basis [Karstadt and Schwartz'17]

$[\mathsf{L};\mathsf{R};\mathsf{P}]$ an $n \times n$ Matrix Multiplication Representation

$\diamond\,(\mathsf{U},\mathsf{V},\mathsf{W}) \in \mathrm{GL}(n^2,\mathbb{K})^3$

$[\mathsf{LU};\mathsf{RV};\mathsf{WP}]$

Transformations of a Fast Matrix Multiplication Algorithm

Sparsification via alternative basis [Karstadt and Schwartz'17]

$$\begin{array}{c} [\mathsf{L};\mathsf{R};\mathsf{P}] \text{ an } n \times n \text{ Matrix Multiplication Representation} \\ \Big\downarrow \diamond(\mathsf{U},\mathsf{V},\mathsf{W}) \in \mathrm{GL}(n^2,\mathbb{K})^3 \\ [\mathsf{LU};\mathsf{RV};\mathsf{WP}] \end{array}$$

Choose $(\mathsf{U},\mathsf{V},\mathsf{W})$ making $[\mathsf{L};\mathsf{R};\mathsf{P}]$ sparser

- ✓ reduces the number of additions
- ✓ reduces the leading constant
- ✗ No longer a Matrix Multiplication alg.
  $\rightsquigarrow$ apply the inverse change of basis on the input in $O(n^2 \log n)$

Transformations of a Fast Matrix Multiplication Algorithm

Sparsification via alternative basis [Karstadt and Schwartz'17]

$$\begin{array}{c} [\mathsf{L};\mathsf{R};\mathsf{P}] \text{ an } n\times n \text{ Matrix Multiplication Representation} \\ \Big\downarrow \diamond(\mathsf{U},\mathsf{V},\mathsf{W}) \in \mathrm{GL}(n^2,\mathbb{K})^3 \\ [\mathsf{LU};\mathsf{RV};\mathsf{WP}] \end{array}$$

Choose $(\mathsf{U},\mathsf{V},\mathsf{W})$ making $[\mathsf{L};\mathsf{R};\mathsf{P}]$ sparser

- ✓ reduces the number of additions
- ✓ reduces the leading constant
- ✗ No longer a Matrix Multiplication alg.
  $\rightsquigarrow$ apply the inverse change of basis on the input in $O(n^2 \log n)$

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & 1 \\ 0 & 1 & -1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 1 & -1 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 \\ 0 & 0 & -1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

- 12 additions $\rightsquigarrow 5n^{\log_2 7} + O(n^2 \log n)$

Improving accuracy

Accuracy bounds

Notations:

ε: machine precision

$\widehat{C}$: the approximation of $C = A \times B$ computed using floating pt arithmetic

Accuracy bounds

Notations:

ε: machine precision

$\widehat{C}$: the approximation of $C = A \times B$ computed using floating pt arithmetic

A first (strict) definition of accuracy [Miller'75]

For the conventional product:

$$|\widehat{\mathsf{C}} - \mathsf{C}| \leq f_{\mathsf{alg}}(n, \varepsilon)|\mathsf{A}| \times |\mathsf{B}| \qquad \text{(coefficient-wise)}$$

Accuracy bounds

Notations:

ε: machine precision

$\widehat{C}$: the approximation of $C = A \times B$ computed using floating pt arithmetic

A first (strict) definition of accuracy [Miller'75]

For the conventional product:

$$|\widehat{\mathsf{C}} - \mathsf{C}| \leq f_{\mathsf{alg}}(n, \varepsilon)|\mathsf{A}| \times |\mathsf{B}| \qquad \text{(coefficient-wise)}$$

- Conventional $O(n^3)$ product: $f_{\mathsf{alg}}(n, \varepsilon) = \frac{(2n-1)\varepsilon}{1-(2n-1)\varepsilon}$
- Moreover any algorithm matching this accuracy must be $\Omega(n^3)$

$\rightsquigarrow$ Long lasting impression that sub-cubic algorithms were unstable

Accuracy bounds

A second definition (commonly used for forward accuracy)

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_\infty \leq f_{\mathsf{alg}}(n)\|\mathsf{A}\|_\infty\|\mathsf{B}\|_\infty\varepsilon + O(\varepsilon^2)$$

- Classic $O(n^3)$ product: $f_{\mathsf{alg}}(n) = n^2$

Accuracy bounds

A second definition (commonly used for forward accuracy)

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_\infty \leq f_{\mathsf{alg}}(n)\|\mathsf{A}\|_\infty\|\mathsf{B}\|_\infty\varepsilon + O(\varepsilon^2)$$

- Classic $O(n^3)$ product: $f_{\mathsf{alg}}(n) = n^2$
- Strassen: $f_{\mathsf{alg}}(n) = O(n^{\log_2 12}\log n)$ [Bini Lotti'80] [Demmel et al.07], [Ballard et al.'16]

Accuracy bounds

A second definition (commonly used for forward accuracy)

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_\infty \leq f_{\mathsf{alg}}(n)\|\mathsf{A}\|_\infty\|\mathsf{B}\|_\infty\varepsilon + O(\varepsilon^2)$$

- Classic $O(n^3)$ product: $f_{\mathsf{alg}}(n) = n^2$
- Strassen: $f_{\mathsf{alg}}(n) = O(n^{\log_2 12}\log n)$ [Bini Lotti'80] [Demmel et al.07], [Ballard et al.'16]
  $f_{\mathsf{alg}}(n) = O(n^{\log_2 12})$ [Brent'70], [Higham'02]

Accuracy bounds

A second definition (commonly used for forward accuracy)

$$\left\|\widehat{C} - C\right\|_\infty \leq f_{\mathsf{alg}}(n)\|A\|_\infty\|B\|_\infty \varepsilon + O(\varepsilon^2)$$

- Classic $O(n^3)$ product: $f_{\mathsf{alg}}(n) = n^2$
- Strassen: $f_{\mathsf{alg}}(n) = O(n^{\log_2 12} \log n)$ [Bini Lotti'80] [Demmel et al.07], [Ballard et al.'16]
  $f_{\mathsf{alg}}(n) = O(n^{\log_2 12})$ [Brent'70], [Higham'02]
- Winograd: $f_{\mathsf{alg}}(n) = O(n^{\log_2 18} \log n)$ [Bini Lotti'80] [Demmel et al.07], [Ballard et al.'16]
  $f_{\mathsf{alg}}(n) = O(n^{\log_2 18})$ [Higham'02]

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_\infty \leq f_{\mathsf{alg}}(n)\|\mathsf{A}\|_\infty\|\mathsf{B}\|_\infty\varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma})$$

and

$$\gamma = \max_{k=1..4}\sum_{i=1}^{7}\|\mathsf{L}_i\|_1\,\|\mathsf{R}_i\|_1\,|p_{i,k}|$$

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_\infty \leq f_{\mathsf{alg}}(n)\|\mathsf{A}\|_\infty\|\mathsf{B}\|_\infty\varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma})$$

and

$$\gamma = \max_{k=1..4} \sum_{i=1}^{7} \|\mathsf{L}_i\|_1 \, \|\mathsf{R}_i\|_1 \, |p_{i,k}|$$

- ✓ $\|\cdot\|_\infty$ in RHS produces the tightest bounds
- ✓ $\gamma = 12$ (as for Strassen's algorithm) is minimal
- ✓ $\gamma = 12$ is attained by many other variants

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_\infty \leq f_{\mathsf{alg}}(n) \|\mathsf{A}\|_\infty \|\mathsf{B}\|_\infty \varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma})$$

and

$$\gamma = \max_{k=1..4} \sum_{i=1}^{7} \|\mathsf{L}_i\|_1 \|\mathsf{R}_i\|_1 |p_{i,k}|$$

- ✓ $\|\cdot\|_\infty$ in RHS produces the tightest bounds
- ✓ $\gamma = 12$ (as for Strassen's algorithm) is minimal
- ✓ $\gamma = 12$ is attained by many other variants

- ✗ no room for improvement
- ✗ does not seem to fully capture the accuracy of the algorithms

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_\infty \leq f_{\mathsf{alg}}(n) \left\|\mathsf{A}\right\| \left\|\mathsf{B}\right\| \varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma})$$

and

$$\gamma = \max_{k=1..4} \sum_{i=1}^{7} \left\|\mathsf{L}_i\right\|^* \left\|\mathsf{R}_i\right\|^* |p_{i,k}|$$

- Bound holds for any norm $\|\cdot\|$ and related dual norm $\|\cdot\|^*$:

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_\infty \leq f_{\mathsf{alg}}(n)\|\mathsf{A}\|_\infty\|\mathsf{B}\|_\infty\varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma})$$

and

$$\gamma = \max_{k=1..4} \sum_{i=1}^{7} \|\mathsf{L}_i\|_1 \|\mathsf{R}_i\|_1 |p_{i,k}|$$

- Bound holds for any norm $\|\cdot\|$ and related dual norm $\|\cdot\|^*$:
  - $(\|\cdot\|_\infty, \|\cdot\|_1)$

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\| \widehat{\mathsf{C}} - \mathsf{C} \right\|_\infty \leq f_{\mathsf{alg}}(n) \left\| \mathsf{A} \right\|_1 \left\| \mathsf{B} \right\|_1 \varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma})$$

and

$$\gamma = \max_{k=1..4} \sum_{i=1}^{7} \|\mathsf{L}_i\|_\infty \|\mathsf{R}_i\|_\infty |p_{i,k}|$$

- Bound holds for any norm $\|\cdot\|$ and related dual norm $\|\cdot\|^*$:
  - $(\|\cdot\|_\infty, \|\cdot\|_1)$
  - $(\|\cdot\|_1, \|\cdot\|_\infty)$

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_{\infty} \leq f_{\mathsf{alg}}(n) \left\|\mathsf{A}\right\|_2 \left\|\mathsf{B}\right\|_2 \varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma})$$

and

$$\gamma = \max_{k=1..4} \sum_{i=1}^{7} \left\|\mathsf{L}_i\right\|_2 \left\|\mathsf{R}_i\right\|_2 |p_{i,k}|$$

- Bound holds for any norm $\|\cdot\|$ and related dual norm $\|\cdot\|^*$:
  - $(\|\cdot\|_\infty, \|\cdot\|_1)$
  - $(\|\cdot\|_1, \|\cdot\|_\infty)$
  - $(\|\cdot\|_2, \|\cdot\|_2)$

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\|\widehat{\mathsf{C}} - \mathsf{C}\right\|_p \leq f_{\mathsf{alg}}(n) \left\|\mathsf{A}\right\|_q \left\|\mathsf{B}\right\|_q \varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma_{p,q}})$$

and

$$\gamma_{p,q} = \left\| \sum_{i=1}^{7} \left\|\mathsf{L}_i\right\|_q^* \left\|\mathsf{R}_i\right\|_q^* |p_{i,k}| \right\|_p$$

- Bound holds for any norm $\|\cdot\|$ and related dual norm $\|\cdot\|^*$:
- arbitrary $\|\cdot\|_p$ on LHS $\|\cdot\|_q$ on RHS

Generalizing the accuracy bounds for Fast Matrix Multiplication

$$\left\| \widehat{\mathsf{C}} - \mathsf{C} \right\|_p \leq f_{\mathsf{alg}}(n) \left\| \mathsf{A} \right\|_q \left\| \mathsf{B} \right\|_q \varepsilon + O(\varepsilon^2)$$

with

$$f_{\mathsf{alg}}(n) = O(n^{\log_2 \gamma_{p,q}})$$

and

$$\gamma_{p,q} = \left\| \sum_{i=1}^{7} \left\| \mathsf{L}_i \right\|_q^* \left\| \mathsf{R}_i \right\|_q^* |p_{i,k}| \right\|_p$$

- Bound holds for any norm $\|\cdot\|$ and related dual norm $\|\cdot\|^*$:
- arbitrary $\|\cdot\|_p$ on LHS $\|\cdot\|_q$ on RHS
- no log whenever `alg` performs a Matrix Multiplication

Search for algorithms

- among the 2×2 multiplication algorithms in 7 products,
- using Isotropies and Sparsification transformations,
- improving accuracy $\rightsquigarrow$ **optimize** $\gamma_{\infty,2}$ w.r.t. norm 2,
- and with a competitive complexity's leading constant.

Optimizing the growth factor in norm 2

Weaker majorations (to optimize a smoother function):

$$\underbrace{\max_{k=1..4} \sum_{i=1}^{7} \|\mathsf{L}_i\|_2 \, \|\mathsf{R}_i\|_2 \, |p_{i,k}|}_{\gamma_{\infty,2}} \quad \leq \quad \underbrace{\sum_{i=1}^{7} \|\mathsf{L}_i\|_2 \, \|\mathsf{R}_i\|_2 \, \|\mathsf{P^\intercal}_i\|_2}_{\gamma_2}$$

Optimizing the growth factor in norm 2

Weaker majorations (to optimize a smoother function):

$$\underbrace{\max_{k=1..4} \sum_{i=1}^{7} \|\mathsf{L}_i\|_2 \|\mathsf{R}_i\|_2 |p_{i,k}|}_{\gamma_{\infty,2}} \leq \underbrace{\sum_{i=1}^{7} \|\mathsf{L}_i\|_2 \|\mathsf{R}_i\|_2 \|\mathsf{P}^{\intercal}{}_i\|_2}_{\gamma_2}$$

Optimization program over all isotropies (U, V, W)

Numerical optimization: approximate solution showing a structure: $\|\mathsf{L}\| = \|\mathsf{R}\| = \|\mathsf{P}\|$

Exact optimization: solving a polynomial system with Gröbner basis

- 1 isotropy $= 3 \times 4$ variables (up to permutations and rotations)
  $\rightsquigarrow$ too hard to solve
- Optimize on the subvariety where $\|\mathsf{L}\| = \|\mathsf{R}\| = \|\mathsf{P}\|$
  $\rightsquigarrow$ 4 variables $\rightsquigarrow$ 2 variables

Optimizing the growth factor in norm 2

Weaker majorations (to optimize a smoother function):

$$\underbrace{\max_{k=1..4} \sum_{i=1}^{7} \|\mathsf{L}_i\|_2 \, \|\mathsf{R}_i\|_2 \, |p_{i,k}|}_{\gamma_{\infty,2}} \leq \underbrace{\sum_{i=1}^{7} \|\mathsf{L}_i\|_2 \, \|\mathsf{R}_i\|_2 \, \|\mathsf{P^T}_i\|_2}_{\gamma_2}$$

Proposition

The minimum of γ_2 *on the subvariety* $\|\mathsf{L}\| = \|\mathsf{R}\| = \|\mathsf{P}\|$ *is* $\gamma_2^* = \frac{16}{\sqrt{3}} + \frac{4}{\sqrt{2}} \approx 12.066,$

reached by the Algorithm $\mathcal{A}^*_{2,2,2} = \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ 0 & 0 & 1 & -\frac{\sqrt{3}}{3} \\ 0 & 1 & 0 & \frac{\sqrt{3}}{3} \\ 0 & 0 & 0 & -\frac{2}{\sqrt{3}} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{6} \end{bmatrix} ; \begin{bmatrix} 0 & \frac{2}{\sqrt{3}} & 0 & 0 \\ -1 & \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 & -1 \\ \frac{1}{2} & -\frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix} ; \begin{bmatrix} \frac{\sqrt{3}}{6} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{3} & 0 & -1 & 0 \\ \frac{\sqrt{3}}{3} & -1 & 0 & 0 \\ \frac{\sqrt{3}}{6} & -\frac{1}{2} & -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{6} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{2}{\sqrt{3}} & 0 & 0 & 0 \end{bmatrix}^{\mathsf{T}}.$

Optimizing the growth factor in norm 2

Weaker majorations (to optimize a smoother function):

$$\underbrace{\max_{k=1..4} \sum_{i=1}^{7} \|\mathsf{L}_i\|_2 \, \|\mathsf{R}_i\|_2 \, |p_{i,k}|}_{\gamma_{\infty,2}} \leq \underbrace{\sum_{i=1}^{7} \|\mathsf{L}_i\|_2 \, \|\mathsf{R}_i\|_2 \, \|\mathsf{P^\intercal}_i\|_2}_{\gamma_2}$$

Proposition

The minimum of γ_2 *is* $\gamma_2^* = \frac{16}{\sqrt{3}} + \frac{4}{\sqrt{2}} \approx 12.066,$

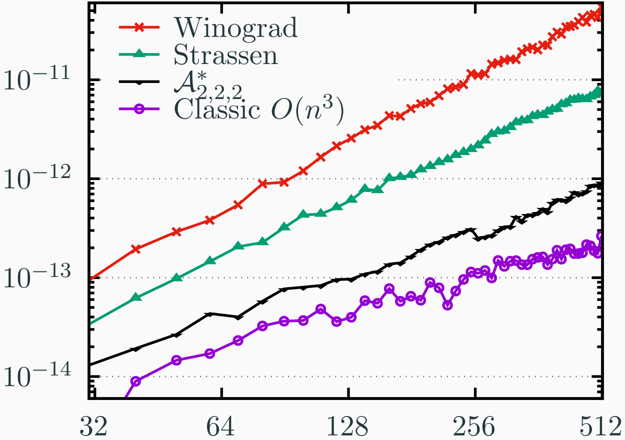
reached by the Algorithm $\mathcal{A}^*_{2,2,2} = \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ 0 & 0 & 1 & -\frac{\sqrt{3}}{3} \\ 0 & 1 & 0 & \frac{\sqrt{3}}{3} \\ 0 & 0 & 0 & -\frac{2}{\sqrt{3}} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{6} \end{bmatrix}; \begin{bmatrix} 0 & \frac{2}{\sqrt{3}} & 0 & 0 \\ -1 & \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 & -1 \\ \frac{1}{2} & -\frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix}; \begin{bmatrix} \frac{\sqrt{3}}{6} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{3} & 0 & -1 & 0 \\ \frac{\sqrt{3}}{3} & -1 & 0 & 0 \\ \frac{\sqrt{3}}{6} & -\frac{1}{2} & -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{6} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{2}{\sqrt{3}} & 0 & 0 & 0 \end{bmatrix}^\intercal.$

A new algorithm with improved accuracy

Algorithm	$\gamma_{\infty,\infty}$	$\gamma_{\infty,2}$	γ_2
Winograd'70	18	8	17.854
Strassen'69	12	6.829	14.829
$\mathcal{A}^*_{2,2,2}$	17.475	5.966	12.066

A new algorithm with improved accuracy

Algorithm	$\gamma_{\infty,\infty}$	$\gamma_{\infty,2}$	γ_2
Winograd'70	18	8	17.854
Strassen'69	12	6.829	14.829
$\mathcal{A}^*_{2,2,2}$	17.475	5.966	12.066



Improving speed while preserving accuracy

Improving speed while preserving accuracy

With isotropies

Any isotropy $(\mathsf{U}, \mathsf{V}, \mathsf{W})$ made of permutations and rotations preserves the growth factor γ.
$\rightsquigarrow$ helped finding the sparse algorithm $\mathcal{A}^{*}_{2,2,2}$ $\rightsquigarrow$ 24 ADD $(+12$ scalar MUL $)$

Improving speed while preserving accuracy

With isotropies

Any isotropy $(\mathsf{U}, \mathsf{V}, \mathsf{W})$ made of permutations and rotations preserves the growth factor γ.
$\rightsquigarrow$ helped finding the sparse algorithm $\mathcal{A}^*_{2,2,2}$ $\rightsquigarrow$ 24 ADD (+12 scalar MUL)

With alternative basis [Karstadt, Schwartz'17]

- Reduces to the optimal 12 ADD $\rightsquigarrow$ $5n^{\log_2 7} + O(n^2 \log n)$ complexity

$$\underbrace{\begin{bmatrix} 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}}_{L^{\Phi}} \times \underbrace{\begin{bmatrix} 0 & 0 & 0 & \frac{2}{\sqrt{3}} \\ 0 & 1 & 0 & \frac{\sqrt{3}}{3} \\ 0 & 0 & 1 & -\frac{\sqrt{3}}{3} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}}_{\Phi} \; ; \; \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 1 \end{bmatrix}}_{R^{\Psi}} \times \underbrace{\begin{bmatrix} 0 & \frac{2}{\sqrt{3}} & 0 & 0 \\ 1 & -\frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 & -1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix}}_{\Psi} \; ; \; \underbrace{\begin{bmatrix} 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}^{\intercal}}_{P^{\nu}} \times \underbrace{\begin{bmatrix} -\frac{2}{\sqrt{3}} & 0 & 0 & 0 \\ \frac{\sqrt{3}}{3} & -1 & 0 & 0 \\ -\frac{\sqrt{3}}{3} & 0 & -1 & 0 \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}^{\intercal}}_{\nu}$$

Accuracy of the alternative basis variant

- [Schwartz, Toledo, Vaknim and Wiernik'24]: claims accuracy is preserved $O(n^{\log_2 \gamma} \log n)$
- Unable to verify the claim, instead best proven bound

$$\gamma^{(AltBasisMatMul)} = \gamma^{(AltBasisTensor)} \left\|\Phi\right\|_q \left\|\Psi\right\|_q \left\|\nu^T\right\|_p$$

Accuracy of the alternative basis variant

- [Schwartz, Toledo, Vaknim and Wiernik'24]: claims accuracy is preserved $O(n^{\log_2 \gamma} \log n)$
- Unable to verify the claim, instead best proven bound

$$\gamma^{(AltBasisMatMul)} = \gamma^{(AltBasisTensor)} \left\|\Phi\right\|_q \left\|\Psi\right\|_q \left\|\nu^T\right\|_p$$

		Standard	Alternative basis
Strassen	$\gamma_{\infty,\infty}$	12	80
	$f_{\text{alg}}(n)$	$10.6n^{3.69}$	$(1+14\log n)n^{6.33}$
Winograd	$\gamma_{\infty,\infty}$	18	270
	$f_{\text{alg}}(n)$	$12.25n^{4.17}$	$(1+15\log n)n^{8.08}$
$\mathcal{A}^*_{2,2,2}$	$\gamma_{\infty,\infty}$	17.48	184
	$f_{\text{alg}}(n)$	$17.94n^{4.13}$	$(1+20\log n)n^{7.52}$

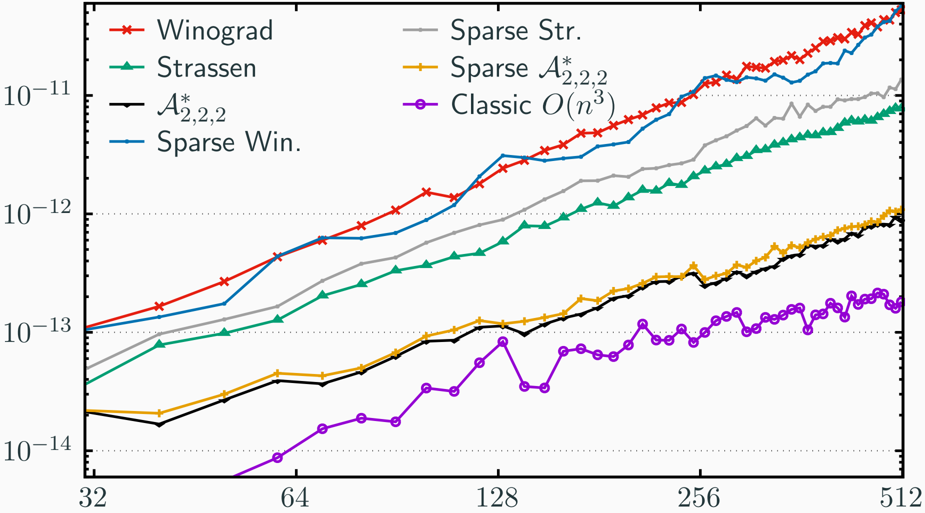
Accuracy of the alternative basis variant

- [Schwartz, Toledo, Vaknim and Wiernik'24]: claims accuracy is preserved $O(n^{\log_2 \gamma} \log n)$
- Unable to verify the claim, instead best proven bound

$$\gamma^{(AltBasisMatMul)} = \gamma^{(AltBasisTensor)} \left\|\Phi\right\|_q \left\|\Psi\right\|_q \left\|\nu^T\right\|_p$$

		Standard	Alternative basis
Strassen	$\gamma_{\infty,2}$	6.83	84
	$f_{\mathsf{alg}}(n)$	$10.38n^{2.78}$	$(1+14\log n)n^{6.40}$
Winograd	$\gamma_{\infty,2}$	8	108
	$f_{\mathsf{alg}}(n)$	$12.43n^{3}$	$(1+15\log n)n^{6.76}$
$\mathcal{A}^*_{2,2,2}$	$\gamma_{\infty,2}$	5.97	95.6
	$f_{\mathsf{alg}}(n)$	$17.74n^{2.58}$	$(1+20\log n)n^{6.58}$

Accuracy of Alternative Basis variants



$4 \times 4 \times 4$ Matrix Multiplication schemes: complexity

$N \times N \times N$ Algorithm	N	field	# Mul	# Add	Recursively
Classic $O(n^3)$	2	any	8	4	$2n^3$
[Strassen'69]	2	any	7	18	$7n^{2.8074}$
[Winograd'70]	2	any	7	15	$6n^{2.8074}$

$4 \times 4 \times 4$ Matrix Multiplication schemes: complexity

$N \times N \times N$ Algorithm	N	field	# Mul	# Add	Recursively
Classic $O(n^3)$	2	any	8	4	$2n^3$
[Strassen'69]	2	any	7	18	$7n^{2.8074}$
[Winograd'70]	2	any	7	15	$6n^{2.8074}$
Winograd$^{\otimes 2}$	4	any	49	165	$6n^{2.8074}$

$4 \times 4 \times 4$ Matrix Multiplication schemes: complexity

$N \times N \times N$ Algorithm	N	field	# Mul	# Add	Recursively
Classic $O(n^3)$	2	any	8	4	$2n^3$
[Strassen'69]	2	any	7	18	$7n^{2.8074}$
[Winograd'70]	2	any	7	15	$6n^{2.8074}$
Winograd$^{\otimes 2}$	4	any	49	165	$6n^{2.8074}$
[Fawzi et al. 22]	4	char$= 2$	47		$O(n^{2.7773})$

$4 \times 4 \times 4$ Matrix Multiplication schemes: complexity

$N \times N \times N$ Algorithm	N	field	# Mul	# Add	Recursively
Classic $O(n^3)$	2	any	8	4	$2n^3$
[Strassen'69]	2	any	7	18	$7n^{2.8074}$
[Winograd'70]	2	any	7	15	$6n^{2.8074}$
Winograd$^{\otimes 2}$	4	any	49	165	$6n^{2.8074}$
[Fawzi et al. 22]	4	char$= 2$	47		$O(n^{2.7773})$
[Novikov et al. 25]	4	$\mathbb{C}$	48		$O(n^{2.7925})$

$4 \times 4 \times 4$ Matrix Multiplication schemes: complexity

$N \times N \times N$ Algorithm	N	field	# Mul	# Add	Recursively
Classic $O(n^3)$	2	any	8	4	$2n^3$
[Strassen'69]	2	any	7	18	$7n^{2.8074}$
[Winograd'70]	2	any	7	15	$6n^{2.8074}$
Winograd$^{\otimes 2}$	4	any	49	165	$6n^{2.8074}$
[Fawzi et al. 22]	4	char$= 2$	47		$O(n^{2.7773})$
[Novikov et al. 25]	4	$\mathbb{C}$	48		$O(n^{2.7925})$
$\mathcal{A}^*_{4,4,4}$	4	char$\neq 2$	48	346	$11.9n^{2.7925}$

$4 \times 4 \times 4$ Matrix Multiplication schemes: complexity

$N \times N \times N$ Algorithm	N	field	# Mul	# Add	Recursively	Alt-basis
Classic $O(n^3)$	2	any	8	4	$2n^3$	
[Strassen'69]	2	any	7	18	$7n^{2.8074}$	$5n^{2.8074}$
[Winograd'70]	2	any	7	15	$6n^{2.8074}$	$5n^{2.8074}$
Winograd$^{\otimes 2}$	4	any	49	165	$6n^{2.8074}$	
[Fawzi et al. 22]	4	char$= 2$	47		$O(n^{2.7773})$	
[Novikov et al. 25]	4	$\mathbb{C}$	48		$O(n^{2.7925})$	
$\mathcal{A}^*_{4,4,4}$	4	char$\neq 2$	48	346	$11.9n^{2.7925}$	$1.19n^{2.7925}$

$4 \times 4 \times 4$ Matrix Multiplication schemes: accuracy

$N \times N \times N$ Algorithm	N	# Mul	Recursively	$\gamma_{\infty,2}$	$f_{\mathsf{alg}}(n)$
Classic $O(n^3)$	2	8	$2n^3$	2	$O(n)$
[Strassen'69]	2	7	$7n^{2.8074}$	6.83	$O(n^{2.7716})$
[Winograd'70]	2	7	$6n^{2.8074}$	8	$O(n^3)$
$\mathcal{A}^*_{2,2,2}$	2	7	$13n^{2.8074}$	5.97	$O(n^{2.58})$

$4 \times 4 \times 4$ Matrix Multiplication schemes: accuracy

$N \times N \times N$ Algorithm	N	# Mul	Recursively	$\gamma_{\infty,2}$	$f_{\text{alg}}(n)$
Classic $O(n^3)$	2	8	$2n^3$	2	$O(n)$
[Strassen'69]	2	7	$7n^{2.8074}$	6.83	$O(n^{2.7716})$
[Winograd'70]	2	7	$6n^{2.8074}$	8	$O(n^3)$
$\mathcal{A}^*_{2,2,2}$	2	7	$13n^{2.8074}$	5.97	$O(n^{2.58})$
Strassen$^{\otimes 2}$	4	49	$7n^{2.8074}$	46.65	$O(n^{2.7716})$

$4 \times 4 \times 4$ **Matrix Multiplication schemes: accuracy**

$N \times N \times N$ Algorithm	N	# Mul	Recursively	$\gamma_{\infty,2}$	$f_{\mathsf{alg}}(n)$
Classic $O(n^3)$	2	8	$2n^3$	2	$O(n)$
[Strassen'69]	2	7	$7n^{2.8074}$	6.83	$O(n^{2.7716})$
[Winograd'70]	2	7	$6n^{2.8074}$	8	$O(n^3)$
$\mathcal{A}^*_{2,2,2}$	2	7	$13n^{2.8074}$	5.97	$O(n^{2.58})$
Strassen$^{\otimes 2}$	4	49	$7n^{2.8074}$	46.65	$O(n^{2.7716})$
$\mathcal{A}^*_{4,4,4}$	4	48	$11.9n^{2.7925}$	27.314	$O(n^{2.3857})$

Thank you

$\mathcal{A}^*_{2,2,2}$ in [Dumas, Sedoglavic, P. 2025]
Towards automated generation of fast and accurate algorithms for recursive matrix multiplication.
JSC in revision. `https://hal.science/hal-04995684v1`

$\mathcal{A}^*_{4,4,4}$ in [Dumas, Sedoglavic, P. 2025]
A non-commutative algorithm for multiplying 4x4 matrices using 48 non-complex multiplications.
`https://hal.science/hal-05112145`

All MatMul algorithms (LRP representations and schedule), maple optimization programs, PlinOpt scripts and Matlab benchmarks are available on

- `https://github.com/jgdumas/Fast-Matrix-Multiplication`.
- `https://github.com/jgdumas/plinopt`.

From a bilinear operator to a program : `https://github.com/jgdumas/plinopt`

Find the best operation schedule from a matrix representation ?

Common Subexpression Elimination

$$\begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ 0 & 0 & 1 & -\frac{\sqrt{3}}{3} \\ 0 & 1 & 0 & \frac{\sqrt{3}}{3} \\ 0 & 0 & 0 & -\frac{2}{\sqrt{3}} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{6} \end{bmatrix};\begin{bmatrix} 0 & \frac{2}{\sqrt{3}} & 0 & 0 \\ -1 & \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 & -1 \\ \frac{1}{2} & -\frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix};\begin{bmatrix} \frac{\sqrt{3}}{6} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{3} & 0 & -1 & 0 \\ \frac{\sqrt{3}}{3} & -1 & 0 & 0 \\ \frac{\sqrt{3}}{6} & -\frac{1}{2} & -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{6} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{2}{\sqrt{3}} & 0 & 0 & 0 \end{bmatrix}^{\intercal} \rightsquigarrow$$

$$\begin{array}{llll} t_1 = \frac{\sqrt{3}}{3}a_{22} & t_2 = a_{21} + t_1 & s_1 = \frac{\sqrt{3}}{3}b_{21} & s_2 = s_1 - b_{11} \\ t_3 = a_{12} + t_2 & l_1 = \frac{\sqrt{3}}{2}a_{11} + \frac{1}{2}t_3 & s_3 = s_2 + b_{22} & r_1 = 2s_1 \\ l_2 = a_{12} - t_1 & l_3 = t_2 & r_2 = s_2 & r_3 = s_1 - b_{22} \\ l_4 = 2t_1 & l_5 = l_2 - l_1 & r_4 = \frac{1}{2}s_3 - \frac{\sqrt{3}}{2}b_{12} & r_5 = r_3 + r_4 \\ l_6 = l_5 + l_4 & l_7 = l_5 + l_3 & r_6 = r_1 - r_5 & r_7 = r_5 - r_2 \end{array}$$

$$\begin{array}{llll} p_1 = l_1{\cdot}r_1 & p_2 = l_2{\cdot}r_2 & p_3 = l_3{\cdot}r_3 & p_4 = l_4{\cdot}r_4 \\ p_5 = l_5{\cdot}r_5 & p_6 = l_6{\cdot}r_6 & p_7 = l_7{\cdot}r_7 & \end{array}$$

$$\begin{array}{llll} w_2 = p_5 + p_1 + p_6 & w_1 = p_7 + p_6 & w_3 = w_2 - p_2 & w_5 = \frac{p_4 + w_2}{2} \\ c_{12} = p_1 - p_3 - w_5 & c_{21} = w_3 - w_5 & c_{22} = \sqrt{3}w_5 & \end{array}$$

$$c_{11} = \frac{\sqrt{3}}{3}(w_3 - c_{12} - 2w_1)$$

From a bilinear operator to a program : `https://github.com/jgdumas/plinopt`

Find the best operation schedule from a matrix representation ?

Common Subexpression Elimination

$$
\begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ 0 & 0 & 1 & -\frac{\sqrt{3}}{3} \\ 0 & 1 & 0 & \frac{\sqrt{3}}{3} \\ 0 & 0 & 0 & -\frac{2}{\sqrt{3}} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{6} \end{bmatrix};
\begin{bmatrix} 0 & \frac{2}{\sqrt{3}} & 0 & 0 \\ -1 & \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 & -1 \\ \frac{1}{2} & -\frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix};
\begin{bmatrix} \frac{\sqrt{3}}{6} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{3} & 0 & -1 & 0 \\ \frac{\sqrt{3}}{3} & -1 & 0 & 0 \\ \frac{\sqrt{3}}{6} & -\frac{1}{2} & -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{6} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{2}{\sqrt{3}} & 0 & 0 & 0 \end{bmatrix}^{\intercal}
\rightsquigarrow
$$

$$
\begin{array}{llll}
t_1 = \frac{\sqrt{3}}{3} a_{22} & t_2 = a_{21} + t_1 & s_1 = \frac{\sqrt{3}}{3} b_{21} & s_2 = s_1 - b_{11} \\
t_3 = a_{12} + t_2 & l_1 = \frac{\sqrt{3}}{2} a_{11} + \frac{1}{2} t_3 & s_3 = s_2 + b_{22} & r_1 = 2 s_1 \\
l_2 = a_{12} - t_1 & l_3 = t_2 & r_2 = s_2 & r_3 = s_1 - b_{22} \\
l_4 = 2 t_1 & l_5 = l_2 - l_1 & r_4 = \frac{1}{2} s_3 - \frac{\sqrt{3}}{2} b_{12} & r_5 = r_3 + r_4 \\
l_6 = l_5 + l_4 & l_7 = l_5 + l_3 & r_6 = r_1 - r_5 & r_7 = r_5 - r_2
\end{array}
$$

$$
\begin{array}{llll}
p_1 = l_1 \cdot r_1 & p_2 = l_2 \cdot r_2 & p_3 = l_3 \cdot r_3 & p_4 = l_4 \cdot r_4 \\
p_5 = l_5 \cdot r_5 & p_6 = l_6 \cdot r_6 & p_7 = l_7 \cdot r_7 &
\end{array}
$$

$$
\begin{array}{llll}
w_2 = p_5 + p_1 + p_6 & w_1 = p_7 + p_6 & w_3 = w_2 - p_2 & w_5 = \frac{p_4 + w_2}{2} \\
c_{12} = p_1 - p_3 - w_5 & c_{21} = w_3 - w_5 & c_{22} = \sqrt{3} w_5 &
\end{array}
$$

$$
c_{11} = \frac{\sqrt{3}}{3} (w_3 - c_{12} - 2 w_1)
$$

$2 \times 2 \times 2$ ADD : 45 $\rightsquigarrow$ 24 scalar MUL: 57 $\rightsquigarrow$ 12,

$4 \times 4 \times 4$ ADD : 960 $\rightsquigarrow$ 304 scalar MUL: 280 $\rightsquigarrow$ 42,

From a bilinear operator to a program : https://github.com/jgdumas/plinopt

Find the best operation schedule from a matrix representation ?

Common Subexpression Elimination

$$\begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ 0 & 0 & 1 & -\frac{\sqrt{3}}{3} \\ 0 & 1 & 0 & \frac{\sqrt{3}}{3} \\ 0 & 0 & 0 & -\frac{2}{\sqrt{3}} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{6} \end{bmatrix} ; \begin{bmatrix} 0 & \frac{2}{\sqrt{3}} & 0 & 0 \\ -1 & \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 & -1 \\ \frac{1}{2} & -\frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix} ; \begin{bmatrix} \frac{\sqrt{3}}{6} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{3} & 0 & -1 & 0 \\ \frac{\sqrt{3}}{3} & -1 & 0 & 0 \\ \frac{\sqrt{3}}{6} & -\frac{1}{2} & -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{6} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{2}{\sqrt{3}} & 0 & 0 & 0 \end{bmatrix}^{\intercal} \rightsquigarrow$$

$$\begin{array}{llll} t_1 = \frac{\sqrt{3}}{3}a_{22} & t_2 = a_{21} + t_1 & s_1 = \frac{\sqrt{3}}{3}b_{21} & s_2 = s_1 - b_{11} \\ t_3 = a_{12} + t_2 & l_1 = \frac{\sqrt{3}}{2}a_{11} + \frac{1}{2}t_3 & s_3 = s_2 + b_{22} & r_1 = 2s_1 \\ l_2 = a_{12} - t_1 & l_3 = t_2 & r_2 = s_2 & r_3 = s_1 - b_{22} \\ l_4 = 2t_1 & l_5 = l_2 - l_1 & r_4 = \frac{1}{2}s_3 - \frac{\sqrt{3}}{2}b_{12} & r_5 = r_3 + r_4 \\ l_6 = l_5 + l_4 & l_7 = l_5 + l_3 & r_6 = r_1 - r_5 & r_7 = r_5 - r_2 \end{array}$$

$$\begin{array}{llll} p_1 = l_1 \cdot r_1 & p_2 = l_2 \cdot r_2 & p_3 = l_3 \cdot r_3 & p_4 = l_4 \cdot r_4 \\ p_5 = l_5 \cdot r_5 & p_6 = l_6 \cdot r_6 & p_7 = l_7 \cdot r_7 & \end{array}$$

$$\begin{array}{llll} w_2 = p_5 + p_1 + p_6 & w_1 = p_7 + p_6 & w_3 = w_2 - p_2 & w_5 = \frac{p_4 + w_2}{2} \\ c_{12} = p_1 - p_3 - w_5 & c_{21} = w_3 - w_5 & c_{22} = \sqrt{3}w_5 & \end{array}$$

$$c_{11} = \frac{\sqrt{3}}{3}(w_3 - c_{12} - 2w_1)$$

$2 \times 2 \times 2$ ADD : 45 $\rightsquigarrow$ 24 scalar MUL: 57 $\rightsquigarrow$ 12,

$4 \times 4 \times 4$ ADD : 960 $\rightsquigarrow$ 304 scalar MUL: 280 $\rightsquigarrow$ 42,

Optimizing memory placement: $\rightsquigarrow$ Pebble games

Rational approximations of $\mathcal{A}^*_{2,2,2}$

$$\mathcal{A}^* = \mathsf{argmin}(\gamma_2(\mathcal{A}(x, y))) = \mathcal{A}\left(\frac{\sqrt{2}}{\sqrt[4]{3}}, -\frac{1}{2}\right)$$

Rational approximations of $\mathcal{A}^*_{2,2,2}$

$$\mathcal{A}^* = \mathrm{argmin}(\gamma_2(\mathcal{A}(x,y))) = \mathcal{A}\left(\frac{\sqrt{2}}{\sqrt[4]{3}}, -\frac{1}{2}\right)$$

Rational approximations of this minimal point $\rightsquigarrow$ Rational approximations of $\mathcal{A}^*$

- $\frac{\sqrt{2}}{\sqrt[4]{3}} \approx 1$ (first convergent) $\rightsquigarrow$ $\gamma_{2,1} \approx 12.203$ and $\gamma_{2,1,\infty} \approx 6.046$

$$\mathcal{A}(1,-\frac{1}{2}) = \begin{bmatrix} 0&0&0&1\\ 1&0&0&0\\ 0&0&1&0\\ 0&0&1&-1\\ 0&1&0&0\\ 1&0&0&1\\ 0&1&0&-1 \end{bmatrix} \times \begin{bmatrix} 1&\frac{1}{2}&-\frac{1}{2}&-\frac{1}{4}\\ 0&0&1&\frac{1}{2}\\ 0&0&1&-\frac{1}{2}\\ 0&-1&1&0 \end{bmatrix} \quad ; \quad \begin{bmatrix} 0&0&0&1\\ 0&0&1&0\\ 0&-1&0&0\\ -1&0&0&0\\ 0&0&1&-1\\ 0&1&0&1\\ 1&0&0&1 \end{bmatrix} \times \begin{bmatrix} -\frac{1}{2}&-\frac{1}{4}&1&\frac{1}{2}\\ 0&-\frac{1}{2}&0&1\\ 1&\frac{1}{2}&0&0\\ 1&0&0&-1 \end{bmatrix} \quad ; \quad \begin{bmatrix} 0&0&0&1\\ -1&0&0&0\\ 0&0&1&0\\ 1&0&0&1\\ 0&0&1&1\\ 0&1&0&0\\ 0&1&0&1 \end{bmatrix}^{\mathsf{T}} \times \begin{bmatrix} -\frac{1}{2}&-1&0&0\\ \frac{1}{2}&-1&0&0\\ \frac{1}{4}&-\frac{1}{2}&-\frac{1}{2}&1\\ 0&1&1&0 \end{bmatrix}^{\mathsf{T}}$$

Rational approximations of $\mathcal{A}^*_{2,2,2}$

$$\mathcal{A}^* = \operatorname{argmin}(\gamma_2(\mathcal{A}(x,y))) = \mathcal{A}\left(\frac{\sqrt{2}}{\sqrt[4]{3}}, -\frac{1}{2}\right)$$

Rational approximations of this minimal point $\rightsquigarrow$ Rational approximations of $\mathcal{A}^*$

- $\frac{\sqrt{2}}{\sqrt[4]{3}} \approx 1$ (first convergent) $\rightsquigarrow \gamma_{2,1} \approx 12.203$ and $\gamma_{2,1,\infty} \approx 6.046$

$$\mathcal{A}(1,-\frac{1}{2}) = \begin{bmatrix} 0&0&0&1\\ 1&0&0&0\\ 0&0&1&0\\ 0&0&1&-1\\ 0&1&0&0\\ 1&0&0&1\\ 0&1&0&-1 \end{bmatrix} \times \begin{bmatrix} 1&\frac{1}{2}&-\frac{1}{2}&-\frac{1}{4}\\ 0&0&1&\frac{1}{2}\\ 0&0&1&-\frac{1}{2}\\ 0&-1&1&0 \end{bmatrix} \quad ; \quad \begin{bmatrix} 0&0&0&1\\ 0&0&1&0\\ 0&-1&0&0\\ -1&0&0&0\\ 0&0&1&-1\\ 0&1&0&1\\ 1&0&0&1 \end{bmatrix} \times \begin{bmatrix} -\frac{1}{2}&-\frac{1}{4}&1&\frac{1}{2}\\ 0&-\frac{1}{2}&0&1\\ 1&\frac{1}{2}&0&0\\ 1&0&0&-1 \end{bmatrix} \quad ; \quad \begin{bmatrix} 0&0&0&1\\ -1&0&0&0\\ 0&0&1&0\\ 1&0&0&1\\ 0&0&1&1\\ 0&1&0&0\\ 0&1&0&1 \end{bmatrix}^{\intercal} \times \begin{bmatrix} -\frac{1}{2}&-1&0&0\\ \frac{1}{2}&-1&0&0\\ \frac{1}{4}&-\frac{1}{2}&-\frac{1}{2}&1\\ 0&1&1&0 \end{bmatrix}^{\intercal}$$

- $\frac{\sqrt{2}}{\sqrt[4]{3}} \approx \frac{14}{13}$ $\rightsquigarrow \gamma_{2,1} \approx 12.0662$ and $\gamma_{2,1,\infty} \approx 6.000043$
- ...

Accuracy of rational approximations