Strassen's algorithm is not optimally accurate

Jean-Guillaume Dumas, Clément Pernet, Alexandre Sedoglavic

Strassen's algorithm is not optimally accurate

Jean-Guillaume Dumas
Université Grenoble Alpes
UMR CNRS 5224 LJK
38058 Grenoble, France

Clément Pernet
Univ. Grenoble Alpes, Grenoble INP
UMR CNRS 5224 LJK
38058 Grenoble, France

Alexandre Sedoglavic
Université de Lille
UMR CNRS 9189 CRISTAL
59650 Villeneuve d'Ascq, France

ABSTRACT

We propose a non-commutative algorithm for multiplying 2×2-matrices using 7 coefficient products. This algorithm reaches simultaneously a better accuracy in practice compared to previously known such fast algorithms, and a time complexity bound with the best currently known leading term (obtained via alternate basis sparsification). To build this algorithm, we consider matrix and tensor norms bounds governing the stability and accuracy of numerical matrix multiplication. First, we reduce those bounds by minimizing a growth factor along the unique orbit of Strassen's 2×2-matrix multiplication tensor decomposition. Second, we develop heuristics for minimizing the number of operations required to realize a given bilinear formula, while further improving its accuracy. Third, we perform an alternate basis sparsification that improves on the time complexity constant and mostly preserves the overall accuracy.

CCS CONCEPTS

• **Computing methodologies** → **Linear algebra algorithms**.

1 INTRODUCTION

The first non-commutative algorithm for multiplying 2×2-matrices using 7 coefficient products was discovered by Strassen [20]. It was subsequently proven that all such algorithms with 7 multiplications all lie in a single isotropy orbit on Strassen bilinear tensor decomposition [13]. We here study the numerical accuracy of the recursive application of these 2×2 algorithms.

We first propose a unified accuracy analysis of such recursive algorithms, generalizing some and improving on other state of the art bounds [1, 2, 4, 7, 8, 10]. Following the approach of [4], we then seek to optimize the growth factor, a parameter governing the accuracy in these bounds, over Strassen's orbit. Since the max-norm, producing the sharpest bounds, precludes smooth optimization, we relax the problem to optimizing a weaker growth factor in the Frobenius norm, which will later demonstrate to better reflect the observed practical accuracy.

The most accurate variants are then obtained from these bilinear formulas by minimizing the number of operations required to realize them. Our heuristics for this, make use of common subexpression eliminations with rational coefficients, potential factorization via the kernel of the matrices of the bilinear operators, as well as the Tellegen's transposition principle.

While preserving the complexity bound exponent of Strassen's algorithm, $n^{\log_2(7)}$, those algorithms require slightly more operations, thus worsening the constant factor of the leading term. We therefore finally propose further variants obtained by an alternate basis sparsification, similar to those introduced in [3, 17]. In fine, we obtain variants having a time complexity bound with the best currently known leading term, that simultaneously improve on the accuracy (i.e. mostly preserving in practice the numerical accuracy with or without alternate basis sparsification, again thanks to a minimization of the number of operations required to realize them).

Our C++ tools for the minimization of the number of operations are gathered in the PLinOpt library [11]. We also forked the Matlab framework of [8] to experiment our implementations of the resulting fast and accurate matrix multiplication algorithms [12].

Section 2 presents the symmetries of matrix multiplication tensors that we will use. In Section 3 we propose the unified error bounds on bilinear operators and matrix multiplication algorithms, highlighting how the the growth factor parameter governs accuracy. On a relaxed growth factor in norm 2, we apply, in Section 4, a descent algorithm to reach some local minima and show in Section 5 that it lies within at most 2.6% of the optimal. Finally, Section 6 presents our minimization heuristics and the obtained matrix multiplication algorithms, and their associated accuracy benchmarks.

2 MATRIX PRODUCT SEEN AS TENSOR

We recall there the formalism of tensor decomposition allowing to present clearly the symmetries, later used to search for more numerically accurate fast matrix multiplication algorithms in Section 4. We start by briefly recalling tensorial representation of bilinear maps, through the example introduced by Strassen in [20] of fast 2×2-matrix product and we refer to [18] for this framework.

The product $C = A \cdot B$ of 2×2 matrices could be computed by Strassen algorithm using the following computations:

$$\begin{array}{ll} \rho_1 \leftarrow a_{11}(b_{12}-b_{22}), & \rho_4 \leftarrow (a_{12}-a_{22})(b_{21}+b_{22}), \\ \rho_2 \leftarrow (a_{11}+a_{12})b_{22}, & \rho_5 \leftarrow (a_{11}+a_{22})(b_{11}+b_{22}), \\ \rho_3 \leftarrow (a_{21}+a_{22})b_{11}, & \rho_7 \leftarrow (a_{21}-a_{11})(b_{11}+b_{12}), \\ \rho_6 \leftarrow a_{22}(b_{21}-b_{11}), & \left[\begin{smallmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{smallmatrix}\right] = \left[\begin{smallmatrix} \rho_5+\rho_4-\rho_2+\rho_6 & \rho_6+\rho_3 \\ \rho_2+\rho_1 & \rho_5+\rho_7+\rho_1-\rho_3 \end{smallmatrix}\right]. \end{array} \tag{1}$$

This straight-line program (a.k.a. SLP) encodes the following bilinear map over a field $\mathbb{K}$ with m, k, n equal to 2:

$$\begin{array}{rccc} \beta_{\mathrm{MM}}(A,B): & \mathbb{K}^{m\times k} \times \mathbb{K}^{k\times n} & \to & \mathbb{K}^{m\times n}, \\ & (A, B) & \to & A \cdot B. \end{array} \tag{2}$$

Indices m, k, n are kept in this section for the sake of clarity in order to distinguish easily the different spaces involved in the sequel.

Definition 1. The spaces $\mathbb{K}^{\cdot\times\cdot}$ can be endowed with the classical Frobenius inner product $\langle M, N\rangle = \mathrm{Trace}(M^\intercal \cdot N)$ that establishes an isomorphism between $\mathbb{K}^{\cdot\times\cdot}$ and its dual space $\left(\mathbb{K}^{\cdot\times\cdot}\right)^*$.

Frobenius inner product combines matrix product (2) and the trilinear form $\mathrm{Trace}(C^\intercal \cdot A \cdot B)$ as follow:

$$\begin{array}{rccc} \mathcal{S}|_3: & \mathbb{K}^{m\times k} \times \mathbb{K}^{k\times n} \times (\mathbb{K}^{m\times n})^* & \to & \mathbb{K}, \\ & (A, B, C^\intercal) & \to & \langle C, A\cdot B\rangle. \end{array} \tag{3}$$

As the space of trilinear forms is the canonical dual space of order three tensor products, Strassen algorithm (1) is encoded as the

tensor decomposition $\mathcal{S}$ of the matrix multiplication tensor in sum of seven rank-one tensors defined by the following relations:

$$\begin{aligned}\mathcal{S} = \textstyle\sum_{i=1}^{7} \mathsf{M}_i\otimes\mathsf{N}_i\otimes\mathsf{O}^i &= \left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right]\\ &+\left[\begin{smallmatrix}0&1\\0&-1\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}0&0\\1&1\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right]+\left[\begin{smallmatrix}-1&0\\1&0\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}1&1\\0&0\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}0&0\\0&1\end{smallmatrix}\right]\\ &+\left[\begin{smallmatrix}1&1\\0&0\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}0&0\\0&1\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}-1&0\\1&0\end{smallmatrix}\right]+\left[\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}0&1\\0&-1\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}0&0\\1&1\end{smallmatrix}\right]\\ &+\left[\begin{smallmatrix}0&0\\0&1\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}-1&0\\1&0\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}1&1\\0&0\end{smallmatrix}\right]+\left[\begin{smallmatrix}0&0\\1&1\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right]\otimes\left[\begin{smallmatrix}0&1\\0&-1\end{smallmatrix}\right]\end{aligned} \tag{4}$$

in $(\mathbb{K}^{m\times k})^*\otimes(\mathbb{K}^{k\times n})^*\otimes\mathbb{K}^{m\times n}$ with $m=k=n=2$. In the above tensor decomposition, each summands is a *rank-one tensor* and its *tensor rank* is the number r of such element (7 there). Given Equation (4), one can retrieve a multiplication formula (2) implemented by Eq. (1) using the third 2-contraction of the tensor $\mathcal{S}\otimes\mathsf{A}\otimes\mathsf{B}$ which is defined as the following map:

$$\begin{gathered}\left((\mathbb{K}^{m\times k})^*\otimes(\mathbb{K}^{k\times n})^*\otimes\mathbb{K}^{m\times n}\right)\otimes\left(\mathbb{K}^{m\times k}\otimes\mathbb{K}^{k\times n}\right)\to\mathbb{K}^{m\times n},\\ \left(\textstyle\sum_i \mathsf{M}_i\otimes\mathsf{N}_i\otimes\mathsf{O}^i\right)\otimes(\mathsf{A}\otimes\mathsf{B})\to\textstyle\sum_i\langle\mathsf{M}_i,\mathsf{A}\rangle\langle\mathsf{N}_i,\mathsf{B}\rangle\mathsf{O}^i.\end{gathered} \tag{5}$$

Matrix product tensor decompositions could be represented using other formalisms more adapted to the design of algorithm computing efficiently the matrix product (as shown in Section 6). For example, a nice concise representation was introduced in [16]; it encodes the sum of rank-one tensors by three matrices as done for the Strassen tensor decomposition (4) in the following three matrices $\mathsf{L}_{\mathcal{S}}, \mathsf{R}_{\mathcal{S}}$ and $\mathsf{P}_{\mathcal{S}}$:

$$\left[\begin{smallmatrix}1&0&0&1\\0&1&0&-1\\-1&0&1&0\\1&1&0&0\\1&0&0&0\\0&0&0&1\\0&0&1&1\end{smallmatrix}\right],\left[\begin{smallmatrix}1&0&0&1\\0&0&1&1\\1&1&0&0\\0&0&0&1\\0&1&0&-1\\-1&0&1&0\\1&0&0&0\end{smallmatrix}\right],\left[\begin{smallmatrix}1&0&0&1\\1&0&0&0\\0&0&0&1\\-1&1&0&0\\0&1&0&1\\1&0&1&0\\0&0&1&-1\end{smallmatrix}\right]^{\mathsf{T}}. \tag{6}$$

Notation 2. Given a $m\times k$-matrix A, we denote by A_i the i-th row and by vec$A$ the row-major vectorization of this matrix, i.e. the vector v in $\mathbb{R}^{mk}$ such that $v_{ik+j}=a_{i,j}$. We also denote by $\mathrm{Mat}_{m,k}(v)$ the reciprocal operation, building an $m\times k$ matrix from an mk-dimensional vector. Thus, the ith line $\mathsf{L}_{\mathcal{S}i}$ (resp. $\mathsf{R}_{\mathcal{S}i}$) of matrix $\mathsf{L}_{\mathcal{S}}$ (resp. $\mathsf{R}_{\mathcal{S}}$) is the transposition of the row-major vectorization vecM_i of the first (resp. second vecN_i) component of the ith triad in Equation (4) and the ith column of matrix $\mathsf{P}_{\mathcal{S}}$ is the column-major vectorization vecO^i of its third component.

Definition 3. This encoding of a tensor by three suitable matrices $\mathsf{L},\mathsf{R},\mathsf{P}$ is called a HM representation and is denoted by $[\mathsf{L};\mathsf{R};\mathsf{P}]$.

Equation (11) presented in Section 3 shows that the HM representation allows to construct SLPs for the associated algorithms. We show in Section 6.2 that this could be done efficiently, e.g., using the kernel of L (resp. R) and Tellegen transposition applied to P.

Now we turn to symmetries of matrix product tensor decomposition. Indeed, remark that the matrix product is associated to $\mathrm{Trace}(\mathsf{A}\cdot\mathsf{B}\cdot\mathsf{C})$ by Equation (3) and that, given invertible matrices $\mathsf{U},\mathsf{V},\mathsf{W}$ of suitable sizes and the trace classical properties, this trace is equal to:

$$\begin{aligned}&\mathrm{Trace}\left((\mathsf{A}\cdot\mathsf{B}\cdot\mathsf{C})^{\mathsf{T}}\right)=\mathrm{Trace}(\mathsf{C}\cdot\mathsf{A}\cdot\mathsf{B})=\mathrm{Trace}(\mathsf{B}\cdot\mathsf{C}\cdot\mathsf{A})\\ &\text{and to }\ \mathrm{Trace}\left(\mathsf{U}^{-1}\cdot\mathsf{A}\cdot\mathsf{V}\cdot\mathsf{V}^{-1}\cdot\mathsf{B}\cdot\mathsf{W}\cdot\mathsf{W}^{-1}\cdot\mathsf{C}\cdot\mathsf{U}\right).\end{aligned} \tag{7}$$

These relations illustrate the next theorem and induce the isotropy action on matrix product tensor decomposition presented below:

Theorem 4 ([13, § 2.8]). The isotropy group of the $m\times m$ matrix multiplication tensor is the semidirect product $\mathrm{PSL}^{\pm}(\mathbb{K}^m)^{\times 3}\rtimes\mathfrak{S}_3$, where PSL stands for the group of matrices of determinant ±1 and $\mathfrak{S}_3$ for the symmetric group on 3 elements.

Lemma 5. Let g denotes $(\mathsf{U}\times\mathsf{V}\times\mathsf{W})$ in $\mathrm{PSL}^{\pm}(\mathbb{K}^m)^{\times 3}$ and $\mathcal{T}$ a rank-one tensor $\mathsf{A}\otimes\mathsf{B}\otimes\mathsf{C}$; the action $\mathsf{g}\diamond\mathcal{T}$ of g on $\mathcal{T}$ is the rank-one tensor $(\mathsf{U}^{-\mathsf{T}}\cdot\mathsf{A}\cdot\mathsf{V}^{\mathsf{T}})\otimes(\mathsf{V}^{-\mathsf{T}}\cdot\mathsf{B}\cdot\mathsf{W}^{\mathsf{T}})\otimes(\mathsf{W}^{-\mathsf{T}}\cdot\mathsf{C}\cdot\mathsf{U}^{\mathsf{T}})$. This action is extended by additivity on higher tensor rank tensors.

Given two isotropies g_1 defined by $(\mathsf{u}_1\times\mathsf{v}_1\times\mathsf{w}_1)$ and g_2 defined by $(\mathsf{u}_2\times\mathsf{v}_2\times\mathsf{w}_2)$ both in $\mathrm{PSL}^{\pm}(\mathbb{K}^m)^{\times 3}$, the composition $g_1\circ g_2$ is given by $(\mathsf{u}_1\cdot\mathsf{u}_2\times\mathsf{v}_1\cdot\mathsf{v}_2\times\mathsf{w}_1\cdot\mathsf{w}_2)$.

The above isotropies action description is based on classical tensor decomposition, the corresponding action on HM representation is a direct consequence and presented in the following lemma.

Lemma 6. Let g be $(\mathsf{U}\times\mathsf{V}\times\mathsf{W})$ in $\mathrm{PSL}^{\pm}(\mathbb{K}^m)^{\times 3}$ and $[\mathsf{L};\mathsf{R};\mathsf{P}]$ be a HM representation of a matrix product tensor decomposition, the action $\mathsf{g}\diamond[\mathsf{L};\mathsf{R};\mathsf{P}]$ of g on $[\mathsf{L};\mathsf{R};\mathsf{P}]$ is another HM representation of a matrix product tensor decomposition defined by:

$$\left[\mathsf{L}\cdot\left(\mathsf{V}^{\mathsf{T}}\otimes\mathsf{U}^{-1}\right);\mathsf{R}\cdot\left(\mathsf{W}^{\mathsf{T}}\otimes\mathsf{V}^{-1}\right);\left(\mathsf{U}\otimes\mathsf{W}^{-\mathsf{T}}\right)\cdot\mathsf{P}\right]. \tag{8}$$

Dealing with a tensor decomposition or with the associated HM representation is not strictly equivalent; In Lemma 5 there is no need to care about the determinant of matrices $(\mathsf{U},\mathsf{V},\mathsf{W})$ while this fact is no more true for Equation (8) as (say) U acts on two different components. The following theorem recalls that in fact all 2×2-matrix product algorithms with 7 coefficient multiplications are obtained by this single orbit of the action of isotropies on Strassen tensor decomposition:

Theorem 7 ([14, § 0.1]). The group $\mathrm{PSL}^{\pm}(\mathbb{K}^m)^{\times 3}$ acts transitively on the variety of optimal algorithms for the computation of 2×2-matrix multiplication.

Thus, isotropy action on Strassen tensor decomposition may define other matrix product algorithm of same tensor rank but with potentially interesting characteristics as shown in Section 4. We explicit these properties in the following section.

3 BILINEAR OPERATOR ACCURACY BOUND

We will consider that any real finite-dimensional vector space $\mathbb{U}$, is equipped with a norm $\|\cdot\|$ and denote by $\|\cdot\|_*$ the related dual norm; for $\phi:\mathbb{U}\to\mathbb{R}$, its norm $\|\phi\|_*$ is $\sup(|\phi(v)|:\|v\|\le1)$. For instance the max-norm $\|\cdot\|_\infty$ and the one-norm $\|\cdot\|_1$ are dual one with the other, while the two-norm $\|\cdot\|_2$ is self-dual. We will also denote the Hamming weight $\#\{i|x_i\neq0\}$ of x by $\|x\|_0$. We will denote by $(x_i)_i$ the vector formed by the coefficients x_i and $\|(x_i)_i\|$ its norm. By extension, we also use $\|x;y\|=\|x\|\cdot\|y\|$ and $\|[\mathsf{L};\mathsf{R};\mathsf{P}]\|=\|\mathsf{L}\|\,\|\mathsf{R}\|\,\|\mathsf{P}\|$.

Lemma 8. For any vectors x and y in $\mathbb{R}^k$ and a matrix A in $\mathbb{R}^{m\times k}$ the following inequalities hold:

$$|x\cdot y| \le \|x\|_*\,\|y\|,\quad \|\mathsf{A}x\|\le\left\|(\|\mathsf{A}_i\|_*)_i\right\|\|x\|, \tag{9}$$

$$\|\mathsf{A}x\|_\infty \le \max_{i=1\ldots m}\left(\textstyle\sum_{j=1}^{k}|a_{i,j}|\right)\|x\|_\infty\le k\|\mathsf{A}\|_\infty\|x\|_\infty. \tag{10}$$

Given an HM representation $[\mathsf{L};\mathsf{R};\mathsf{P}]$ of a matrix multiplication tensor decomposition, one can retrieve the transpose of the multiplication formula (2) implemented by Eq. (1) using the Hadamard product $\mathsf{A}\odot\mathsf{B}$ of matrices A and B with the following map:

$$\begin{array}{ccc}\mathbb{K}^{m\times k}\times\mathbb{K}^{k\times n} & \to & \mathbb{K}^{mn\times 1}\\ (\mathsf{A},\mathsf{B}) & \to & \mathsf{P}^\intercal\cdot((\mathsf{L}\cdot\mathrm{vecA})\odot(\mathsf{R}\cdot\mathrm{vecB}))\,.\end{array} \tag{11}$$

Hence, we express there a bilinear operator $\beta:\mathbb{R}^e\times\mathbb{R}^f\to\mathbb{R}^g$ using its HM representation $[\mathsf{L};\mathsf{R};\mathsf{P}]$ in $\mathbb{R}^{r\times e}\times\mathbb{R}^{r\times f}\times\mathbb{R}^{r\times g}$ as: $\beta(u,v)=\sum_{i=1}^r(\mathsf{L}_i\cdot u)(\mathsf{R}_i\cdot v)(\mathsf{P}^\intercal)_i$. When this operator encodes an $m\times k$ by $k\times n$ matrix multiplication formula, we will thus denote it by β_{MM} and we will have $e=mk, f=kn, g=mn$. We also consider recursive applications of such operators defined as:

$$\begin{array}{rccl}\beta^{(\ell)}: & \mathbb{R}^{e_0e^\ell}\times\mathbb{R}^{f_0f^\ell} & \to\mathbb{R}^{g_0g^\ell} & \\ & (u,v) & \mapsto\sum_{i=1}^r\beta^{(\ell-1)}(\mathsf{L}_i\cdot u,\mathsf{R}_i\cdot v)(\mathsf{P}^\intercal)_i\end{array} \tag{12}$$

and $\beta^0:\mathbb{R}^{e_0}\times\mathbb{R}^{f_0}\to\mathbb{R}^{g_0}$, a bilinear operator which we will assume to be bounded: $\|\beta^{(0)}(u,v)\|\le\gamma_0\|u\|\|v\|\ \forall(u,v)\in\mathbb{R}^{e_0}\times\mathbb{R}^{f_0}$. For convenience, we will define the dimensions $G=g^\ell g_0$, and $K=k^\ell k_0$. Recall that $(\mathsf{P}^\intercal)_i$ is the i-th column of P and remark that $\mathsf{L}_i\cdot u$ is an abuse of notation for the operation where each coefficient $\mathsf{L}_{i,j}$ multiplies a block of $e_0e^{\ell-1}$ contiguous coefficients of u, namely: $\mathsf{L}_i\cdot u=(\mathsf{L}_i\mathrm{Mat}_{\mathsf{e},\mathsf{e}_0\mathsf{e}^{\ell-1}}(\mathsf{u}))^\intercal$.

We will consider the floating point arithmetic in the standard model of [15]: $\widehat{x}$ denotes the computed value for an expression x such that $\widehat{a\ \mathrm{op}\ b}=(a\ \mathrm{op}\ b)(1+\delta)$ for $\mathrm{op}=+,-,\times,/$ where $|\delta|\le\varepsilon$, the unit round off, except when a op b is 0 where δ is -1. We recall in the following Lemma some classical inequalities:

Lemma 9 (see [8, Eq. (3.5)] and [15, Eq. (4.4)]). For any vectors u and v in $\mathbb{R}^n$ the following inequalities hold:

$$|\widehat{u\cdot v}-u\cdot v|\le\|u\|_0\,\|u\|_*\,\|v\|\varepsilon+O(\varepsilon^2). \tag{13}$$

$$\left|\widehat{\textstyle\sum_{i=1}^n u_i}-\sum_{i=1}^n u_i\right|\le(n-1)\left(\textstyle\sum_{i=1}^n|u_i|\right)\varepsilon+O(\varepsilon^2). \tag{14}$$

Proof. Since there are $r=\|u\|_0$ non-zero coefficients in u, the scalar product is actually computed between r-dimensional vectors. The result is then as in [8, Eq. (3.5)]. Applying Eq. (13) with $v=(1,\dots,1)$ and the max-norm gives Eq. (14). □

Definition 10. The *growth factor* of the formula $[\mathsf{L};\mathsf{R};\mathsf{P}]$ computing the bilinear form β is defined by $\gamma=\max_{j=1\dots g}\sum_{i=1}^r\|\mathsf{L}_i\|_*\,\|\mathsf{R}_i\|_*\,|p_{i,j}|$.

The growth factor not only bounds the values of bilinear operators, as show in Lemma 11, but is also central in analyzing their forward numerical error, which will be the focus of Theorem 12.

Lemma 11. For any u,v with adequate dimensions, the following relations hold: $\|\beta(u,v)\|\le\gamma\,\|u\|\,\|v\|$, $\|\beta^{(\ell)}(u,v)\|\le\gamma^\ell\gamma_0\,\|u\|\,\|v\|$ and $\|\beta^{(\ell)}_{\mathrm{MM}}(u,v)\|_\infty\le k^\ell k_0\|u\|_\infty\|v\|_\infty$.

Proof. Let $\mathsf{D_j}$ denotes $\mathrm{Diag}_{i=1\dots r}(p_{i,j})$ and c_j be the j-th coefficient of $\beta(u,v)$. We have that $|c_j|\le\left\|\mathsf{u}^\intercal\mathsf{L}^\intercal\mathsf{D}_j\mathsf{Rv}\right\|\le\left\|\mathsf{u}^\intercal\mathsf{L}\mathsf{D}_j\mathsf{R}\right\|_*\|v\|$, so that $|c_j|\le\left\|\mathsf{L}^\intercal\mathsf{D}_j\mathsf{R}\right\|_*\|u\|\,\|v\|\le\left\|\sum_{i=1}^r(\mathsf{L}_i\otimes\mathsf{R}_i)p_{i,j}\right\|_*\|u\|\,\|v\|$ and $|c_j|\le\left(\sum_{i=1}^r\|\mathsf{L}_i\|_*\,\|\mathsf{R}_i\|_*\,|p_{i,j}|\right)\|u\|\,\|v\|$. Finally, the last inequality follows from (9). □

Theorem 12. Given any choice of norm $\|\cdot\|$, if G denotes g_0g^ℓ and K denotes k_0k^ℓ, the error in computing $\beta^{(\ell)}$ is of the form $\|\widehat{\beta^{(\ell)}}(u,v)-\beta^{(\ell)}(u,v)\|_\infty\le\kappa\,\|u\|\,\|v\|\,\varepsilon+O(\varepsilon^2)$ where either

$$\kappa=(K/k_0)^{\log_k\gamma}\left(k_0^2+Q_0\frac{\gamma}{\gamma-k}k_0\right)-Q_0\frac{\gamma}{\gamma-k}K \tag{15}$$

when $\beta^{(\ell)}$ is an $M\times K$ by $K\times N$ matrix multiplication, or

$$\kappa=(G/g_0)^{\log_g\gamma}\,\gamma_0\left(1+Q_0\left(\log_g(G/g_0)+1\right)\right), \tag{16}$$

otherwise, and $Q_0=\max_j\left(\left\|(\mathsf{P}^\intercal)_j\right\|_0+\max_i(\|\mathsf{L}_i\|_0+\|\mathsf{R}_i\|_0)\mathbb{1}_{p_{i,j}\neq0}\right)$ as in [1, Definition 1].

Proof. By induction we will prove that the bound is of the form $\|\Delta_{\beta^{(\ell)}}\|_\infty=\|\widehat{\beta^{(\ell)}(u,v)}-\beta^{(\ell)}(u,v)\|_\infty\le t_\ell\,\|u\|\,\|v\|\,\varepsilon+O(\varepsilon^2)$, clarifying in the process the value for t_ℓ. Consider the block c_j of $G/g=g_0g^{\ell-1}$ consecutive coefficients of the output: $c_j=\sum_{i=1}^r\mathsf{H}_ip_{i,j}$, where $\mathsf{H}_i=\beta^{(\ell-1)}(\mathsf{L}_i\cdot u,\mathsf{R}_i\cdot v)$. Let $d_{i,j}=\mathsf{H}_ip_{i,j}$ and $\Delta_{d_{i,j}}=\widehat{d_{i,j}}-d_{i,j}$. Then, by Eq. (14):

$$\|\widehat{c_j}-c_j\|_\infty\le\|\widehat{\sum_{i=1}^r\widehat{d_{i,j}}}-\sum_{i=1}^r\widehat{d_{i,j}}\|_\infty+\|\sum_{i=1}^r\widehat{d_{i,j}}-\sum_{i=1}^r d_{i,j}\|_\infty \tag{17}$$

$$\le\sum_{i=1}^r\|\widehat{\mathsf{H}_ip_{i,j}}\|_\infty\left(\left\|(\mathsf{P}^\intercal)_j\right\|_0-1\right)\varepsilon+\sum_{i=1}^r\|\Delta_{d_{i,j}}\|_\infty+O(\varepsilon^2) \tag{18}$$

Similarly,

$$\|\Delta_{d_{i,j}}\|_\infty\le\|\widehat{p_{i,j}\widehat{\mathsf{H}_i}}-p_{i,j}\widehat{\mathsf{H}_i}\|_\infty+\|p_{i,j}\widehat{\mathsf{H}_i}-p_{i,j}\mathsf{H}_i\|_\infty \tag{19}$$

$$\le|p_{i,j}|\|\mathsf{H}_i\|_\infty\varepsilon+|p_{i,j}|\|\Delta_{\mathsf{H}_i}\|_\infty+O(\varepsilon^2), \tag{20}$$

and again by bilinearity of $\beta^{(\ell-1)}$, the following equality holds: $\|\Delta_{\mathsf{H}_i}\|_\infty=\|\widehat{\beta^{(\ell-1)}}(\widehat{\mathsf{L}_i\cdot u},\widehat{\mathsf{R}_i\cdot v})-\beta^{(\ell-1)}(\mathsf{L}_i\cdot u,\mathsf{R}_i\cdot v)\|_\infty$, and this quantity is bounded by:

$$\|\Delta_{\beta^{(\ell-1)}}\|_\infty+\|\beta^{(\ell-1)}(\Delta_L,\mathsf{R}_i\cdot v)\|_\infty+\|\beta^{(\ell-1)}(\mathsf{L}_i\cdot u,\Delta_R)\|_\infty \tag{21}$$

By Eq. (13) and the induction hypothesis we have

$$\|\Delta_{\mathsf{M}}\|_\infty\le\|\mathsf{M}_i\|_0\,\|\mathsf{M}_i\|_*\,\|u\|\,\varepsilon+O(\varepsilon^2)\text{ with }\mathsf{M}\in\{\mathsf{L},\mathsf{R}\}, \tag{22}$$

$$\|\Delta_{\beta^{(\ell-1)}}\|_\infty\le t_{\ell-1}\,\|\mathsf{L}_i\cdot u+\Delta_{\mathsf{L}}\|\,\|\mathsf{R}_i\cdot v+\Delta_{\mathsf{R}}\|\,\varepsilon+O(\varepsilon^2), \tag{23}$$

$$\le c_{\ell-1}\,\|\mathsf{L}_i\|_*\,\|u\|\,\|\mathsf{R}_i\|_*\,\|v\|\,\varepsilon+O(\varepsilon^2). \tag{24}$$

By Lemma 11, the following inequality holds

$$\|\beta^{(\ell-1)}(\Delta_{\mathsf{L}},\mathsf{R}_i\cdot v)\|_\infty\le\Theta^{\ell-1}\Theta_0\|\Delta_{\mathsf{L}}\|_\infty\,\|\mathsf{R}_i\|_*\,\|v\| \tag{25}$$

for $(\Theta,\Theta_0)=(k,k_0)$ if $\beta=\beta_{\mathrm{MM}}$ or (γ,γ_0) otherwise (where $\Theta_0=\gamma_0$ comes from the current proof with $\ell=1$ and $g_0=1$). Similarly,

$$\|\beta^{(\ell-1)}(\mathsf{L}_i\cdot u,\Delta_{\mathsf{R}})\|_\infty\le\Theta^{\ell-1}\Theta_0\|\Delta_{\mathsf{R}}\|_\infty\,\|\mathsf{L}_i\|_*\,\|u\|\,. \tag{26}$$

Gathering Eqs. (18), (20) to (22) and (24) to (26) we deduce that

$$\|\widehat{c_j}-c_j\|_\infty\le\textstyle\sum_{i=1}^r(\Theta^{\ell-1}\Theta_0(\|\mathsf{L}_i\|_0+\|\mathsf{R}_i\|_0+\left\|(\mathsf{P}^\intercal)_j\right\|_0)+t_{\ell-1})$$
$$\times\,\|\mathsf{L}_i\|_*\,\|\mathsf{R}_i\|_*\,|p_{i,j}|\,\|u\|\,\|v\|\,\varepsilon+O(\varepsilon^2),$$

and thus that $\|\widehat{c_j}-c_j\|_\infty\le(\Theta^{\ell-1}\Theta_0Q_0+t_{\ell-1})\gamma\,\|u\|\,\|v\|\,\varepsilon+O(\varepsilon^2)$. As in [15], we deduce that t_ℓ must then satisfy:

$$\begin{cases}t_\ell &=& (\Theta^{\ell-1}\Theta_0Q_0+t_{\ell-1})\gamma & \text{for }\ell>0,\\ t_0 &=& k_0^2 & \text{for matrix product,}\\ t_0 &=& (1+Q_0)\gamma_0 & \text{otherwise.}\end{cases} \tag{27}$$

	Formula	Applies to	norm	Q	γ	Winograd Q	Winograd γ	Strassen Q	Strassen γ	Equation (35) Q	Equation (35) γ	Equation (34) Q	Equation (34) γ
Brent [7]	(15)	Strassen only	∞	NA	$\gamma_{1,1,\infty}$			3.67	12				
BL [4] DDHK [10]	(16)	any MM alg.	∞	Q_0'[1]	$\gamma_{0,1,\infty}$[2]	9	18	7	12	9.59	40	9.81	98.54
Higham [15]	(15)	S. & W. only	∞	NA	$\gamma_{1,1,\infty}$	4.94	18	3.83	12				
Ballard et al. [1]	(16)	any MM alg.	∞	Q_0	$\gamma_{1,1,\infty}$	10	18	8	12	12	13	15	17.48
Dai, Lim [8]	[8, Th 3.3]	$\ell = 1$, any alg.	2	$m+n+r$	$\gamma_{2,1}$	15	17.86	15	14.83	15	12.21	15	12.07
			∞	$m+n+r$	$\gamma_{1,\infty,1}$	15	27	15	20	15	22	15	25.14
Here	(15)	any MM alg.	2	Q_0	$\gamma_{2,1,\infty}$	10	8	8	6.83	12	6.05	15	5.97
			∞	Q_0	$\gamma_{1,1,\infty}$	10	18	8	12	12	13	15	17.48
Here	(16)	any alg.	∞	Q_0	$\gamma_{1,1,\infty}$	10	18	8	12	12	13	15	17.48

Table 1: Comparing accuracy formulas for recursive bilinear matrix multiplication operators in the form of Theorem 12.

This recurrence relation solves into $t_\ell = \gamma^\ell t_0 + Q_0\Theta_0\Theta^\ell \sum_{i=1}^{\ell} (\gamma/\Theta)^i$. In the case of a matrix multiplication operator, t_ℓ is equal to: $\gamma^\ell k_0^2 + Q_0 k_0 \gamma \frac{\gamma^\ell - k^\ell}{\gamma - k} = \left(\frac{K}{k_0}\right)^{\log_k \gamma}\left(k_0^2 + \frac{Q_0\gamma}{\gamma-k}k_0\right) - \frac{Q_0\gamma}{\gamma-k}K$. In the general case, the value of t_ℓ becomes: $\gamma^\ell\gamma_0(1+Q_0) + Q_0\gamma_0\gamma^\ell\ell$, that is: $(G/g_0)^{\log_g\gamma}\gamma_0\left(1+Q_0\left(\log_g G/g_0 + 1\right)\right)$. □

Theorem 12 generalizes or improves on previous similar results in [1, 7, 8, 10, 15]. In particular, [8] only considers one recursive level with no base case; [7, 15] have tight bounds but only for Strassen and Winograd's algorithms in max-norm; and lastly [1, 10] has an additional logarithmic factor likely due to a loser bound on each $\|\beta^{(\ell-1)}\|_\infty$, not exploiting the fact that they are matrix products.

Eventhough the choice of the max-norm produces the tightest bounds in Theorem 12, as in most of previous works, the bounds are stated for any choice of norm, as in [8], for alternative norms, such as the 2-norm, may give growth factor expressions more amenable to optimizations, as detailed in the following section.

Table 1 compares the various existing bounds on numerical accuracy of matrix multiplication algorithms. These bounds depend on the following choices made on the norms to define the growth factor γ:

$$\begin{aligned}\gamma_{0,1,\infty} &= \left\|\left(\left\|\left(\|\mathsf{L}_i;\mathsf{R}_i\|_0\,|p_{i,k}|\right)_i\right\|_1\right)_k\right\|_\infty \|\mathsf{L}\|_\infty\|\mathsf{R}\|_\infty\|\mathsf{P}\|_\infty \\ &= \left(\max_{k\in\{1\ldots mn\}} \textstyle\sum_{i=1}^r \|\mathsf{L}_i\|_0\|\mathsf{R}_i\|_0|p_{i,k}|\right)\|\mathsf{L}\|_\infty\|\mathsf{R}\|_\infty\|\mathsf{P}\|_\infty \end{aligned} \tag{28}$$

$$\gamma_{2,1} = \left\|(\|\mathsf{L}_i;\mathsf{R}_i;\mathsf{P}_i\|_2)_i\right\|_1 = \textstyle\sum_{i=1}^r \|\mathsf{L}_i\|_2\|\mathsf{R}_i\|_2\|\mathsf{P}_i\|_2 \tag{29}$$

$$\begin{aligned}\gamma_{q,1,\infty} &= \left\|\left(\left\|\left(\|\mathsf{L}_i;\mathsf{R}_i\|_q\,|p_{i,k}|\right)_i\right\|_1\right)_k\right\|_\infty \\ &= \max_{k\in\{1\ldots mn\}} \textstyle\sum_{i=1}^r \|\mathsf{L}_i\|_q\|\mathsf{R}_i\|_q|p_{i,k}| \text{ with } q\in\{1,2\}.\end{aligned} \tag{30}$$

4 GROWTH FACTOR ALONG ORBITS

In the footstep of [4], we aim to find alternative 2×2 matrix product tensor decomposition, in the orbit of Strassen's one, with improved accuracy, hence minimizing the Growth factor. The use of the maxnorm induces an expression Eq. (30) for $\gamma_{1,1,\infty}$ poorly suited for optimizations. We will instead make two relaxations: first, using the 2-norm, and second, as in [8] bounding $\gamma_{2,1,\infty}$ by $\gamma_{2,1}$:

$$\max_{k\in\{1\ldots mn\}} \sum_{i=1}^r \|\mathsf{L}_i\|_2\,\|\mathsf{R}_i\|_2\,|p_{i,k}| \le \sum_{i=1}^r \|\mathsf{L}_i\|_2\,\|\mathsf{R}_i\|_2\,\|\mathsf{P}_i\|_2 \tag{31}$$

Theorem 7 shows that all fast 2×2 matrix product algorithms are in the same orbit under isotropies action introduced in Lemma 5. While the tensor rank is invariant under this action, growth factor is—generally—not. As its definition is based on Frobenius norm, some isotropies leave it invariant as stated next:

Lemma 13. The growth factor $\gamma_{2,1}$ is invariant under the action of the semidirect product $\mathrm{so}^\pm(\mathbb{K}^n)^{\times 3} \rtimes \mathfrak{S}_3$ induced by the special orthogonal group and the permutation group $\mathfrak{S}_3$.

Proof. By Definition 1, Frobenius norms are invariant under orthogonal transformations and so is $\gamma_{2,1}$ by Eq. (29). Lemma 13 is then derived from Equations (7) and (8). □

As it is useless to consider isotropies leaving the growth factor invariant, we limit our search to isotropies of the following form:

Lemma 14. The action of $(\mathrm{h}\times\mathrm{p})^{\times 3}$ determines the growth factor $\gamma_{2,1}$ for $\mathrm{h} = \{\mathsf{H}_\rho = \left[\begin{smallmatrix}\rho & 0\\ 0 & 1/\rho\end{smallmatrix}\right] \mid \rho > 0\}$ and $\mathrm{p} = \{\mathsf{P}_\xi = \left[\begin{smallmatrix}1 & \xi\\ 0 & 1\end{smallmatrix}\right] \mid \xi \in \mathbb{R}\}$.

Proof. Considering Eq. (8), we see that the product of any action, say U, by a non-zero scalar will affect the growth factor exactly twice: once in U and once, inverted, in U^{-1}, as norms are absolutely homogeneous. It is therefore sufficient to consider matrices with determinant 1. Then, Lemma 13 states that orthogonal matrices also do not have any effect. From the QR decomposition of any invertible matrices there remains thus just the $(\mathrm{h}\times\mathrm{p})$ part of the Iwasawa decomposition, for each of the three transformations in Theorem 7. □

We should study the action of $(\mathrm{h}\times\mathrm{p})^{\times 3}$ on Strassen tensor decomposition in order to find variants with the smaller possible $\gamma_{2,1}$. Unfortunately, a direct definitive result for this question seems to be out of reach and we present several ersatz. First, we perform numerical minimization on $\gamma(\mathrm{g}\diamond\mathcal{S})$ with a completely generic isotropy g in $\mathrm{psl}^\pm(\mathbb{R}^2)^{\times 3}$ (involving 6 indeterminates by Lemma 14); this experiment suggests that a suitable isotropy to reach a fast matrix product tensor decomposition with minimal $\gamma_{2,1}$ could be of the form $(u\times u\times u)$ (involving only 2 indeterminates). The following

[1] [4, 10] reach an improved value of Q_0 by assuming all additions are performed following a balanced tree, instead of a worst case estimate as done in all other formulas.
[2] We applied the same $\gamma_{0,1,\infty}$ for [4], as it seems to be missing a dependency in the magnitude of the coefficients in L, R, P, which was fixed in [10].

proposition states precisely this possibility (its proof is a simple computation presented in Appendix A.1).

Proposition 15. Consider the matrices $u(\rho,\xi) = \mathsf{H}_\rho \cdot \mathsf{P}_\xi$ and the isotropies $\mathsf{g}_{\rho,\xi}$ defined by $u(\rho,\xi)^{\times 3}$. The minimal value on the orbit $\mathsf{g}_{\rho,\xi} \diamond \mathcal{S}$ of the growth factor $\gamma_{2,1}(\mathsf{g}_{\rho,\xi} \diamond \mathcal{S})$ is reached at the point $(\rho,\xi) = (\sqrt[4]{4/3}, -1/2)$ and equal to $4/\sqrt{2} + 16/\sqrt{3} > 12.06603$.

The algorithm corresponding to the point (ρ,ξ) with minimal $\gamma_{2,1}$ on this restricted orbit is given in Eq. (34). We gather in Table 2 values for $\gamma_{2,1}$ of some matrix product tensor decompositions together with the result obtained in Proposition 15. In Section 6, we compare the implementation of algorithms associated to these tensor decompositions in order to confirm that their numerical accuracy is correlated to their respective $\gamma_{2,1}$ growth factor.

5 UPPER AND LOWER BOUNDS

We explore in this section some bounds on the norm of each component of a HM representation. By the multiplicativity of $L_{p,q}$ norms (even generalized to negative Hölder conjugates), this will always give alternate bounds on the error, a priori less accurate, but potentially easier to apprehend.

Lemma 16. For any HM representation $\mathcal{H}$, with matrices $\mathsf{L}, \mathsf{R}, \mathsf{P}$ in $\mathbb{K}^{r\times n}$, let $\gamma_{\mathcal{H}} = \gamma_{2,1}(\mathcal{H})$ be its $\gamma_{2,1}$ growth factor, as in Eq. (29). Then for any strictly positive y and z, we have both:

$$\gamma_{\mathcal{H}} \le \|\mathcal{H}\|_{2,3} \le \|\mathcal{H}\|_F \qquad \text{and} \tag{32}$$

$$\max\left\{ r^{1+3z}\, \|\mathcal{H}\|_{2,-\frac{1}{z}} ; \|\mathsf{L}\|_{2,-\frac{1}{y}} \cdot \|\mathsf{R}\|_{2,-\frac{1}{z}} \cdot \|\mathsf{P}\|_{2,\frac{1}{1+y+z}} \right\} \le \gamma_{\mathcal{H}} \tag{33}$$

Proof. Let $a_i = \|\mathsf{L}_i\|_2$, $b_i = \|\mathsf{R}_i\|_2$ and $c_i = \|\mathsf{P}_i\|_2$. The first right hand side inequality is Hölder's inequality on a_i, b_i and c_i, with the Hölder conjugates $\frac{1}{3}+\frac{1}{3}+\frac{1}{3} = 1$, that is: $\|(a_i \cdot b_i \cdot c_i)_i\|_1 \le \|(a_i)_i\|_3 \cdot \|(b_i)_i\|_3 \cdot \|(c_i)_i\|_3 = \|\mathcal{H}\|_{2,3}$ The second right hand side inequality is a direct application of the monotonicity of norms. Then, the left hand side inequality is obtained by a reverse Hölder's inequality on the vectors a_i, b_i, c_i and 1, with the Hölder conjugates $\frac{1}{-1/z}+\frac{1}{-1/z}+\frac{1}{-1/z}+(1+3z) = 1$. We have indeed that the $(1+3z)$-norm $\|(1)_i\|_{1/(1+3z)}$ is $\left(\sum_{i=1}^{r} 1^{1/(1+3z)}\right)^{1+3z}$. Combined with: $\|\mathcal{H}\|_{2,-\frac{1}{z}} = \|(a_i)_i\|_{-\frac{1}{z}} \cdot \|(b_i)_i\|_{-\frac{1}{z}} \cdot \|(c_i)_i\|_{-\frac{1}{z}}$, this shows that $r^{1+3z}\,\|\mathcal{H}\|_{2,-\frac{1}{z}} \le \|(a_i \cdot b_i \cdot c_i \cdot 1)_i\|_1$. Finally, for the other lhs, we also use Hölder's inequality on a_i, b_i and c_i, now with Hölder conjugates $(1/-1/y)+(1/-1/z)+(1/1/(1+y+z)) = 1$. □

Table 2 gives the Frobenius and $(2,3)$-norms of each of three matrices defining the HM representation of a matrix product algorithm, as well as their $\gamma_{2,1}$ growth factor. In the following proposition, we show that—up to orthogonal transformations—the minimum of the Frobenius norm of the HM representation defining a fast 2×2-matrix multiplication algorithms is $\sqrt{10}$ and subsequent results.

Proposition 17. The minimal product $\|\mathcal{H}\|_F$ of the three Frobenius norms of the HM representation of any bilinear algorithm for matrix multiplication with 7 multiplications, is $\sqrt{10}^3$.

Algorithm	$\gamma_{2,1}(\mathcal{H})$	$\|\mathcal{H}\|_{2,3}$	$\|\mathcal{H}\|_F$
Winograd	$7+\frac{8}{\sqrt{2}}+\frac{9}{\sqrt{3}} \approx 17.853$	$11+\frac{8}{\sqrt{2}}+\frac{9}{\sqrt{3}}$	$\sqrt{14}^3$
Strassen	$12+\frac{4}{\sqrt{2}} \approx 14.828$	$2+\frac{20}{\sqrt{2}}$	$\sqrt{12}^3$
Eq. (35)	$\frac{75}{8}+\frac{4}{\sqrt{2}} \approx 12.203$	$\frac{125}{32}+\frac{4}{\sqrt{2}}+\frac{25}{2\sqrt{5}}$	$\sqrt{\frac{162}{16}}^3$
Eq. (40)	$\frac{75}{8}+\frac{4}{\sqrt{2}} \approx 12.203$	$\frac{125}{32}+\frac{4}{\sqrt{2}}+\frac{25}{2\sqrt{5}}$	$\sqrt{10}\sqrt{\frac{162}{16}}\frac{810}{80\sqrt{10}}$
Eq. (34)	$\frac{16}{\sqrt{3}}+\frac{4}{\sqrt{2}} \approx 12.066$	$\frac{16}{\sqrt{3}}+\frac{4}{\sqrt{2}}$	$\sqrt{10}^3$
Conv.	8.000	8	$\sqrt{8}^3$

Table 2: Illustration of Eq. (32) on $\mathcal{H} = [\mathsf{L};\mathsf{R};\mathsf{P}]$

This proposition's proof is given in Appendix A.1. Remark that this lower bound is reached, by the algorithm which HM representation is given in Equation (34).

$$\left[\begin{smallmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ 0 & 0 & 1 & -\frac{\sqrt{3}}{3} \\ 0 & 1 & 0 & \frac{\sqrt{3}}{3} \\ 0 & 0 & 0 & -\frac{2}{\sqrt{3}} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{6} \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{6} \end{smallmatrix}\right] ; \left[\begin{smallmatrix} 0 & \frac{2}{\sqrt{3}} & 0 & 0 \\ -1 & \frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 & -1 \\ \frac{1}{2} & -\frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{6} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{smallmatrix}\right] ; \left[\begin{smallmatrix} \frac{\sqrt{3}}{6} & \frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{3} & 0 & -1 & 0 \\ \frac{\sqrt{3}}{3} & -1 & 0 & 0 \\ \frac{\sqrt{3}}{6} & -\frac{1}{2} & -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{\sqrt{3}}{6} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{2}{\sqrt{3}} & 0 & 0 & 0 \end{smallmatrix}\right]^{\mathsf{T}} \tag{34}$$

Remark 18. Similarly, the point $\left(\sqrt[4]{3}/\sqrt{2}, -\sqrt[4]{3}/\sqrt{6}, \sqrt[4]{3}/\sqrt{2}, \sqrt[4]{3}/\sqrt{6}\right)$ is a minimum of the cube of $\|\mathsf{L}\cdot(W\otimes V)\|_{2,3}$. It turns out that this value is $16/\sqrt{3}+4/\sqrt{2}$, the same as the $\gamma_{2,1}$ growth factor at this point, proving that our upper bound is reached.

We now turn to potential lower bounds.

Lemma 19. For $W = \left[\begin{smallmatrix} r & x \\ 0 & r^{-1} \end{smallmatrix}\right]$ and $V = \left[\begin{smallmatrix} s & y \\ 0 & s^{-1} \end{smallmatrix}\right]$, and L the first component of Strassen's HM representation given in Equation (6), and any $z \ge 0.5171$, the point $\left(\sqrt[4]{3}/\sqrt{2}, -\sqrt[4]{3}/\sqrt{6}, \sqrt[4]{3}/\sqrt{2}, \sqrt[4]{3}/\sqrt{6}\right)$ is a local minimum of $\|\mathsf{L}\cdot(W\otimes V)\|_{2,-1/z}$ as a function of r, x, s and y.

Proof. Alike the proof of Proposition 17, we give the explicit expression of $\|\mathsf{L}\cdot(W\otimes V)\|_{2,-1/z}$ as the function:

$$f_z(r,x,s,y) = \Big((r^2s^2+r^2y^2+x^2s^2+(xy+1/rs)^2)^{-1/2z} + (s^2/r^2+(y/r-1/rs)^2)^{-1/2z} + (r^2s^2+r^2y^2+(xs+s/r)^2+(xy+y/r)^2)^{-1/2z} + (r^2s^2+r^2y^2+s^2x^2+x^2y^2)^{-1/2z} + (1/r^2s^2)^{-1/2z} + (r^2/s^2+(x/s+1/rs)^2)^{-1/2z} + (r^2s^2+(r/s-ry)^2+x^2s^2+(x/s-xy)^2)^{-1/2z}\Big)^{-z}.$$

Then the evaluation of its partial derivatives at the given point is zero, by inspection, for any z in $\mathbb{R}$. Now, the roots of the characteristic polynomial of the Hessian of f_z at this point are $3\frac{n}{d}\left(b_1 \pm \sqrt{\delta_1}\right)$ and $\sqrt{3}\frac{n}{d}\left(b_2 \pm \sqrt{\delta_2}\right)$ for $\delta_1 = -16z(24z-1)\tau + 13443^{\frac{1}{z}}(z^2 - 2/7z + 13/448)2^{-\frac{1}{z}} + 48z^2$, $\delta_2 = 24z(272z-237)\tau + 40323^{\frac{1}{z}}(z^2 - 12/7z + 333/448)2^{-\frac{1}{z}} + 2704z^2$, $n = (2^{\frac{-1}{2z}} + 63^{\frac{1}{2z}}2^{-\frac{1}{z}})^{-z}$, $d = 216z\tau + 36z$, $b_1 = ((32z-11)\tau + 4z)\sqrt{3}$, and $b_2 = (96z-63)\tau + 52z$ with $\tau = 3^{\frac{1}{2z}}2^{\frac{-1}{2z}}$. First, δ_1 and δ_2 are never negative, so that the roots are always real. Second, both expressions ${b_i}^2 - \delta_i$ have the same two roots, the lowest one is $1/4$ and the second one is strictly less than 0.5171.

Third, all four roots are positive for z equal to this 0.5171, and, by continuity, so are they for $z \geq 0.5171$. □

Corollary 20. $11.7554696 < \frac{28}{9} 2^{\frac{11}{14}} 3^{\frac{5}{7}}$ is a lower bound for the $\gamma_{2,1}$ growth factor of an HM formula using 7 products.

Proof. Following the proof of Lemma 19, we have that equality $f_z\left(\frac{\sqrt[4]{3}}{\sqrt{2}}, \frac{-\sqrt[4]{3}}{\sqrt{6}}, \frac{\sqrt[4]{3}}{\sqrt{2}}, \frac{\sqrt[4]{3}}{\sqrt{6}}\right) = (2^{\frac{-1}{2z}} + 63^{\frac{1}{2z}} 2^{-\frac{1}{z}})^{-z}$ hold. Denoting this quantity by ζ_z we thus have that $7^{1+3z}\zeta_z^3 \leq 7^{1+3z} \|\mathcal{H}\|_{2,-\frac{1}{z}}$. Which lhs limit at $z = \infty$ is $\frac{28}{9} 2^{\frac{11}{14}} 3^{\frac{5}{7}}$. By Eq. (33), this show that $\frac{28}{9} 2^{\frac{11}{14}} 3^{\frac{5}{7}}$ is less or equal to γ_F as announced. Similarly, the limit of $\zeta_z^2 \zeta_{-1-2z} \leq \|\mathsf{L}\|_{2,-\frac{1}{z}} \cdot \|\mathsf{R}\|_{2,-\frac{1}{z}} \cdot \|\mathsf{P}\|_{2,\frac{1}{1+z+z}}$, at $z = \infty$, is also $\frac{28}{9} 2^{\frac{11}{14}} 3^{\frac{5}{7}}$. □

Corollary 20 for instance also shows that the $\gamma_{2,1}$ growth factor of the conventional algorithm (8) can not be attained by such fast algorithms. Let see now how this bound behave in our experiments.

6 ALGORITHMS INTO PRACTICE

In this section, we present several techniques to lower the number of operations used in our algorithms and thus, lower complexity bounds and potentially obtain a better accuracy.

Determining actual complexity bounds requires to estimate the number of operations required to implement a given formula. Considering a HM representation, a direct upper bound can be obtained by first count the number of coefficients different from $-1, 0, 1$ to get an upper bound on the number of multiplications/divisions, second count the number of non-zero coefficients, minus the number of rows, to get an upper bound on the number of additions/subtractions.

To obtain lower operation counts, we use the following techniques: first, we select among equivalently accurate algorithms; this is presented in Section 6.1; second, we factor as much as possible the computations between rows of the HM representations, as in Section 6.2; third, we use dependent rows as more opportunities for factorization, as in Section 6.3. We then present some good candidates (as well as in Appendix A.4) and we eventually look at some potential sparse change of basis in Section 6.4.

6.1 Sparsifying via rotations

We have seen in Lemma 13 that orthogonal transformations leave the Frobenius norm invariant and thus, the $\gamma_{2,1}$ growth factor. Therefore, one can apply 4×4 generic Kronecker products of orthogonal 2×2 (rotation) matrices using Lemma 6 and try to optimize considered HM representation for several possible goals: (1) a smaller number of non-zero coefficients in HM representation components; (2) a non-zero pattern better suited to factorization (see the technique of Section 6.2); (3) a triangular (sparse) subset of independent rows (see the technique of Section 6.3).

For instance, to obtain Eq. (34), we solve for the minimal values of the Frobenius norms as in Proposition 17, and then for orthogonal transformations that produce as many vectors of the canonical basis as possible. Doing so, we found that with $\gamma_{2,1}$ set to $16/\sqrt{3} + 4/\sqrt{2}$ and HM representation component Frobenius norms set to $\sqrt{10}$, the maximal possible number of canonical vectors was 1. Equation (34) is one of those. Similarly, Eq. (40) is an orthogonal optimization of Eq. (35), with one canonical vector in each of components of the HM representation. A C++ implementation of these tools is available in the PLinOpt library [11].

6.2 Factoring heuristics

For the implementation of a given linear operator (one of the matrices in the HM representation) one can try to find the shortest straight-line program for its computation. The problem is NP-hard in general (see e.g. [6]); but for small matrices, and over the field with 2 elements, [6] and references therein, propose several heuristics that potentially reduce the number of operations.

Not all of them are applicable to fields with more elements but we use a kind of common sub-expression eliminations, the "cancellation-free" search, described in Algorithm 6 and implemented in `plinopt/optimizer -D` [11].

6.3 Kernel computation and transposition

If the rank of the linear operator is lower than its number of rows, then an additional strategy has proven useful: compute first some independent rows, then express the dependent ones by their linear relationsFor this, Algorithm 1 computes a left Kernel of the linear operator and uses it to compute the dependent rows via linear combinations of the independent ones. This is sometimes faster than directly computing the dependent rows. Of course, if the matrix's rank is lower than the number of *columns*, one can apply Algorithm 1 to the transposed matrix, and then apply the *Tellegen transposition* principle to recover the transposed linear dependencies (see, e.g., [5] and references therein).

Algorithm 1 Kernel decomposition of a linear operator

Input: M in $\mathbb{K}^{m\times n}$ such that $r = \operatorname{Rank} M$.
Output: A straight line program computing $x \hookrightarrow M \cdot x$.

1: By Gaussian elimination, compute $M = P \cdot L \cdot U \cdot Q$ with P a permutation matrix, L in $\mathbb{K}^{m\times r}$ be $\begin{bmatrix} L_1 \\ L_2 \end{bmatrix}$ unit upper triangular and L_1 in $\mathbb{K}^{r\times r}$; choosing P so that (1) the first r rows of $P^{-1}M$ are sparsest; (2) L_1 is the sparsest; (3) L_2 is the sparsest;
2: Let σ be the permutation represented by P;
3: Apply Alg. 6 to $[\, r_{\sigma(1)} \cdots r_{\sigma(r)} \,]^\intercal \longrightarrow [\, I_r 0 \,] \cdot P \cdot M \cdot \vec{x}$;
▷ $[\, -L_2 \cdot {L_1}^{-1}\ I_{m-r} \,]$ is a (sparse) left kernel of M and provides the linear dependencies of the remaining rows
4: Apply Alg. 6 to $[\, r_{\sigma(r+1)} \cdots r_{\sigma(m)} \,]^\intercal \longrightarrow L_2 \cdot {L_1}^{-1} [\, r_{\sigma(1)} \cdots r_{\sigma(r)} \,]^\intercal$.

Algorithm 1 is implemented in `plinopt/optimizer -K`. The transposition principle applied to such straight-line programs is implemented in `plinopt/transpozer` [11]. These routines have produced the implementations given in the following, for our different HM formulas (for instance the implementation Table 3 of Eq. (34) with only 24 additions and 12 multiplications/divisions).

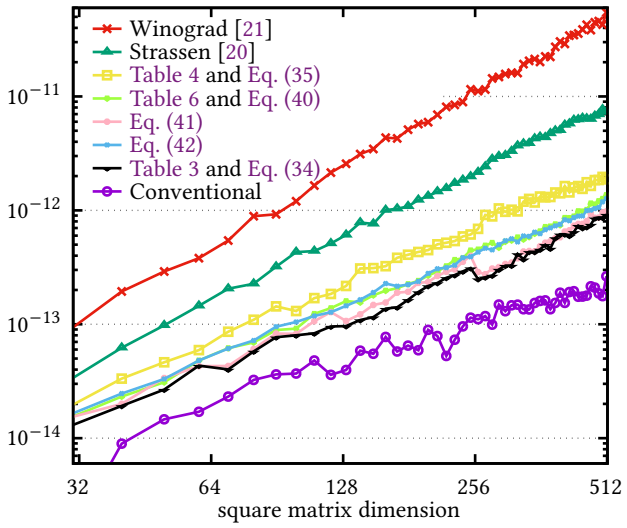
Remark 21. The accuracy obtained with our different fast variants is given in Figure 1 using the Matlab framework of [8], which we forked in [12] and where we have just added the implementations of the variants presented here. Thus, in Figures 1, 2, 3 and 4 we present the error as the infinite norm of the difference between the result of our implementations and the exact matrix multiplication.

$$
\begin{array}{llll}
t_1 = \frac{\sqrt{3}}{3}a_{22} & t_2 = a_{21} + t_1 & s_1 = \frac{\sqrt{3}}{3}b_{21} & s_2 = s_1 - b_{11} \\
t_3 = a_{12} + t_2 & l_1 = \frac{\sqrt{3}}{2}a_{11} + \frac{1}{2}t_3 & s_3 = s_2 + b_{22} & r_1 = 2s_1 \\
l_2 = a_{12} - t_1 & l_3 = t_2 & r_2 = s_2 & r_3 = s_1 - b_{22} \\
l_4 = 2t_1 & l_5 = l_2 - l_1 & r_4 = \frac{1}{2}s_3 - \frac{\sqrt{3}}{2}b_{12} & r_5 = r_3 + r_4 \\
l_6 = l_5 + l_4 & l_7 = l_5 + l_3 & r_6 = r_1 - r_5 & r_7 = r_5 - r_2 \\
\multicolumn{4}{c}{p_1 = l_1 \cdot r_1 \quad p_2 = l_2 \cdot r_2 \quad p_3 = l_3 \cdot r_3 \quad p_4 = l_4 \cdot r_4} \\
\multicolumn{4}{c}{p_5 = l_5 \cdot r_5 \quad p_6 = l_6 \cdot r_6 \quad p_7 = l_7 \cdot r_7} \\
w_2 = p_5 + p_1 + p_6 & w_1 = p_7 + p_6 & w_5 = \frac{p_4 + w_2}{2} & w_3 = w_2 - p_2 \\
c_{12} = p_1 - p_3 - w_5 & c_{21} = w_3 - w_5 & c_{22} = \sqrt{3}w_5 & \\
\multicolumn{4}{l}{c_{11} = \frac{\sqrt{3}}{3}(w_3 - c_{12} - 2w_1)}
\end{array}
$$

Table 3: SLP of Eq. (34) with 24 add. and 12 mul./div.

In Figure 1, all our variants, Tables 3, 4 and 6 and Eqs. (41) and (42) with decreasing $\gamma_{2,1}$, are mostly more and more accurate. Our best algorithm presents an order of magnitude advantage over Strassen's and two orders of magnitude advantage over Winograd's. It is then quite close to the conventional algorithm's accuracy. Figure 1 uses normal distribution, but the same behavior is obtained, e.g., with a uniform distribution in Figure 4.

Figure 1: Numerical accuracy vs size (normal distribution)



Remark 22. In [4] the authors consider all bilinear algorithms using 7 multiplications with constants of the form $\pm 2^i$; they shown that Strassen's original method [20] reaches in this class the minimum value 12 of their $\gamma_{0,1\infty}$ factor error bound (while for instance that of Winograd [21] is 18, see also Table 1). In Eq. (35) and Table 4 we propose an algorithm in this class that has a worse $\gamma_{0,1\infty}$ of 40, but a $\gamma_{2,1}$ of $4/\sqrt{2} + 75/8 \approx 12.2034$, better than those of Strassen 14.8284 or Winograd 17.8530 (see Table 2). Figure 1 shows that this algorithm is also more accurate in practice than both Strassen's and Winograd's variants.

$$
\begin{bmatrix} 0 & -1 & 1 & 0 \\ 1 & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{4} \\ 0 & 0 & 1 & -\frac{1}{2} \\ 0 & 1 & 0 & -\frac{1}{2} \\ 0 & 0 & 1 & \frac{1}{2} \\ 1 & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{4} \\ 0 & 1 & 0 & \frac{1}{2} \end{bmatrix}
;
\begin{bmatrix} 1 & 0 & 0 & -1 \\ 1 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 0 & -1 \\ \frac{1}{2} & \frac{1}{4} & -1 & -\frac{1}{2} \\ 0 & \frac{1}{2} & 0 & 1 \\ 1 & -\frac{1}{2} & 0 & 0 \\ \frac{1}{2} & -\frac{1}{4} & 1 & -\frac{1}{2} \end{bmatrix}
;
\begin{bmatrix} 0 & 1 & 1 & 0 \\ \frac{1}{2} & 1 & 0 & 0 \\ \frac{1}{4} & -\frac{1}{2} & -\frac{1}{2} & 1 \\ -\frac{1}{2} & 0 & 1 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{2} & 1 \\ \frac{1}{2} & -1 & 0 & 0 \\ \frac{1}{2} & 0 & 1 & 0 \end{bmatrix}^{\mathsf{T}} \tag{35}
$$

$$
\begin{array}{llll}
r_1 = \frac{1}{2}a_{22} & t_2 = a_{21} - r_1 & u_1 = \frac{1}{2}b_{12} & s_1 = b_{11} + u_1 \\
t_3 = a_{12} + r_1 & t_0 = t_2 - t_3 & s_2 = u_1 - b_{22} & u_2 = s_1 - b_{22} \\
t_4 = a_{21} + r_1 & r_2 = t_2 - a_{12} & s_4 = b_{22} + u_1 & s_0 = s_1 - s_4 \\
t_5 = a_{11} + \frac{1}{2}r_2 & t_1 = t_5 - t_0 & s_3 = \frac{1}{2}u_2 - b_{21} & s_5 = s_0 - s_2 \\
\multicolumn{4}{c}{p_1 = t_0 \cdot s_0 \quad p_2 = t_1 \cdot s_1 \quad p_3 = t_2 \cdot s_2 \quad p_4 = t_3 \cdot s_3} \\
\multicolumn{4}{c}{p_5 = t_4 \cdot s_4 \quad p_6 = t_5 \cdot s_5 \quad p_7 = (t_4 - t_0) \cdot (s_0 - s_3)} \\
\multicolumn{4}{l}{v_1 = p_5 - p_3 \quad v_2 = p_1 + \frac{1}{2}v_1 \quad v_3 = p_7 + p_6 - p_4 + p_2} \\
\multicolumn{4}{l}{c_{22} = p_5 + p_3 \quad c_{12} = p_2 - p_6 + v_2 \quad c_{11} = \frac{1}{2}(v_3 + \frac{1}{2}c_{22}) \quad c_{21} = p_4 + p_7 + v_2}
\end{array}
$$

Table 4: SLP of Eq. (35) with 27 add., 7 div. by 2 and $\gamma_F \approx 12.2034$

Remark 23. Eq. (35) was obtained by approximating the minimal point of the $\gamma_{2,1}$ growth factor taken from Proposition 15 with the smallest powers of 2. Further rational higher-order approximations are obtained in the same vein, giving for instance Eqs. (40) to (42) and Table 6, as shown in Appendix A.2.

6.4 Alternate basis sparsification

The technique of [3, 17] reduces the number of operations by factoring each matrix in the HM decomposition into a sparser one via a 4×4 change of basis (CoB). In a recursive version, the left and right hand sides (resp. result) CoB can be recursively precomputed (resp. post-computed), for a total cost in $O(n^2 \log n)$. In the meantime the sparsest 7×4 matrices are applied reducing the dominant term of the computation. The optimal decomposition of Winograd's algorithm in [17, § 3.3] reduces the number of intermediate additions from 15 to 12. For a fully recursive version, this gives a constant factor reduction of the complexity bound from $6n^{\log_2(7)}$ to $5n^{\log_2(7)}$.

This is also the case for the algorithm of Eq. (34), for instance, that can use the CoB of Eq. (36) to obtain the sparse recursion of Eq. (37). There the constant factor reduction of complexity bound is from $13n^{\log_2(7)}$ for Table 3 to $5n^{\log_2(7)}$. To obtain this sparse CoB, the generic technique of [3] can be used. In our case, for 4×4 CoB, the following heuristic was sufficient to obtain optimal (12-additions) sparse $-1, 0, 1$ intermediate matrices: (1) Find independent columns of each CoB one at a time; (2) For this, alternatively factor-out common coefficients in the resulting columns and find a linear combination minimizing the density of the resulting column, using as coefficients of the combination only $-1, 0, 1$ and some of the values of the coefficients of the input; (3) Until this alternation does not sparsify anymore. This heuristic is implemented in `plinopt/sparsifier` [11] and resulting sparse implementation is shown in Algorithms 2 to 5 and Table 5.

$$\begin{bmatrix} 0 & 0 & 0 & \frac{2}{\sqrt{3}} \\ 0 & 1 & 0 & \frac{\sqrt{3}}{3} \\ 0 & 0 & 1 & -\frac{\sqrt{3}}{3} \\ -\frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}; \begin{bmatrix} 0 & \frac{2}{\sqrt{3}} & 0 & 0 \\ 1 & -\frac{\sqrt{3}}{3} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 & -1 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} & -\frac{1}{2} \end{bmatrix}; \begin{bmatrix} -\frac{2}{\sqrt{3}} & 0 & 0 & 0 \\ \frac{\sqrt{3}}{3} & -1 & 0 & 0 \\ -\frac{\sqrt{3}}{3} & 0 & -1 & 0 \\ \frac{\sqrt{3}}{2} & -\frac{1}{2} & \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix}^{\mathsf{T}} \tag{36}$$

$$\begin{bmatrix} 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}; \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 1 \end{bmatrix}; \begin{bmatrix} 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}^{\mathsf{T}} \tag{37}$$

Algorithm 2 LCoB(A, ℓ) left change-of-basis of Eq. (36)

if $\ell \leq 0$ **then return** A. **end if**
$m_1 = \text{LCoB}(a_{11}, \ell-1)$; $m_2 = \text{LCoB}(a_{21}, \ell-1)$;
$m_3 = \text{LCoB}(a_{12}, \ell-1)$; $m_4 = \text{LCoB}(a_{22}, \ell-1)$;
$t_1 = (m_3 - m_2)/2$; $t_2 = m_4\sqrt{3}/3$; $t_3 = t_1 + m_1\sqrt{3}/2$;
return $[m_4\sqrt{3}/2 + t_3, m_2 - t_2, m_3 + t_2, m_4\sqrt{3}/6 - t_3]$.

Algorithm 3 RCoB(A, ℓ) right change-of-basis of Eq. (36)

if $\ell \leq 0$ **then return** A. **end if**
$m_1 = \text{RCoB}(a_{11}, \ell-1)$; $m_2 = \text{RCoB}(a_{21}, \ell-1)$;
$m_3 = \text{RCoB}(a_{12}, \ell-1)$; $m_4 = \text{RCoB}(a_{22}, \ell-1)$;
$t_1 = (m_1 - m_4)/2$; $t_2 = m_3\sqrt{3}/6 - m_2\sqrt{3}/2$;
return $[t_1 + t_2, (m_1 + m_4)/2 + (m_2 - m_3)\sqrt{3}/2, m_3\sqrt{3}/3, t_2 - t_1]$.

$$\begin{gathered} s_1 = a_{11} + a_{12} \quad s_2 = a_{11} + a_{22} \quad s_3 = a_{11} - a_{21} \\ t_1 = b_{12} + b_{22} \quad t_2 = b_{11} + b_{12} \quad t_3 = b_{12} + b_{21} \\ p_1 = a_{11}{\cdot}b_{12} \quad p_2 = s_1{\cdot}b_{21} \quad p_3 = a_{21}{\cdot}t_1 \quad p_4 = a_{12}{\cdot}t_2 \\ p_5 = s_2{\cdot}b_{22} \quad p_6 = a_{22}{\cdot}t_3 \quad p_7 = s_3{\cdot}b_{11} \\ c_{11}{=}p_7{-}p_6 \quad c_{12}{=}p_2{+}p_3 \quad c_{21}{=}p_4{-}p_5 \quad c_{22}{=}p_1{+}p_2{+}p_5{+}p_6 \end{gathered}$$

Table 5: SLP of Eq. (37) with 12 additions

Algorithm 4 CoBP(A, ℓ) product change-of-basis of Eq. (36)

if $\ell \leq 0$ **then return** A. **end if**
$m_1 = \text{CoBP}(a_{11}, \ell-1)$; $m_2 = \text{CoBP}(a_{21}, \ell-1)$;
$m_3 = \text{CoBP}(a_{12}, \ell-1)$; $m_4 = \text{CoBP}(a_{22}, \ell-1)$;
$t_1 = m_4/2$; $t_2 = m_4\sqrt{3}/2$;
return $[m_1 2/\sqrt{3} + (m_3 - m_2)\sqrt{3}/3 + t_2, m_2 - t_1, m_3 + t_1, t_2]$.

Remark 24. With Algorithm 5, we can get the best of both worlds: the best known dominant term of the complexity bound, while mostly preserving the best known numerical accuracy. The former property comes from the fact that Eq. (37) requires only 12 additions. The latter is shown in Figure 2 but seems more complex to prove.

Indeed, the sparsest 7×4 matrices have a slightly reduced $\gamma_{2,1}$ growth factor. For the algorithm of [17], this growth factor is indeed reduced from $7 + 8/\sqrt{2} + 9/\sqrt{3} \approx 17.853$, for [21], to $4 + 12/\sqrt{2} \approx 12.486$. But the three CoB and each ℓ recursive calls add some terms

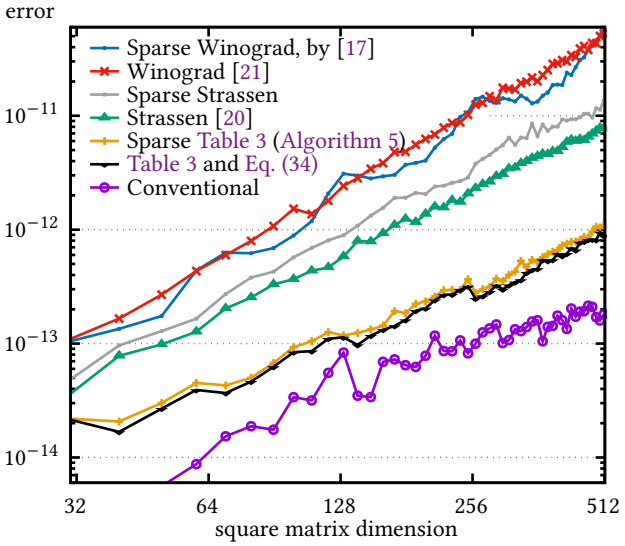
Algorithm 5 Sparsification of Eq. (34)

Input: $A, B \in \mathbb{K}^{n_0 2^\ell \times n_0 2^\ell}$.
Output: $C = A{\cdot}B$.
$\bar{A} \leftarrow \text{LCoB}(A, \ell)$; $\bar{B} \leftarrow \text{RCoB}(B, \ell)$ ▷ Via Algorithms 2 and 3
$\bar{C} \leftarrow \bar{A} \cdot \bar{B}$; ▷ Via Table 5 with ℓ recursive calls
return $C \leftarrow \text{CoBP}(\bar{C}, \ell)$. ▷ Via Algorithm 4

in the bounds and has then to be taken into account. From Lemmas 8 and 11, even with the infinite operator norm, we thus have to consider the maximum absolute row sum of the CoB. It is 3, for that of [17]. Thus the error upper bound grows at least to $\left(\gamma{\cdot}3^3\right)^\ell$. Similarly, the $\gamma_{2,1}$ growth factor of Eq. (37) is lower than that of Eq. (34): from $4/\sqrt{2} + 16/\sqrt{3} \approx 12.066$ to $7 + 6/\sqrt{2} \approx 11.243$. But again, the three CoB in Eq. (36) have maximum absolute row sum $1 + \sqrt{3} \approx 2.73$. The error upper bound then grows at least to $\left(\gamma{\cdot}(10 + 18/\sqrt{3})\right)^\ell$.

In all cases though, these large upper bounds do not reflect the experiments shown in Figure 2. They show, on the contrary, that sparsification seems to mostly preserve accuracy. The accuracy of the sparse Winograd variant, by [17], is around that of the original; the accuracy of the sparse Strassen variant is slightly worse; and the accuracy of Algorithm 5 is only barely inferior to that of Table 3 and Eq. (34), despite their better time complexity.

Figure 2: Numerical effect of sparsification (normal distribution)



Following [19, § 3.2], we can also confirm our algorithms' accuracy on badly conditioned matrices (see Figure 3).

Remark 25. To further improve their practical behavior, as done in [9, § 4.3], [2, § 6.1] or [1, § 6], some diagonal scaling adapted to specific input matrices can also be added to any algorithms. The idea is to precondition the input matrices with well suited D_1, D_2, D_3 thanks to the relation $\mathsf{A} \cdot \mathsf{B} = {D_1}^{-1}\big((D_1 \cdot AD_2){\cdot}(D_2^{-1} \cdot BD_3)\big){D_3}^{-1}$. This can be still be applied to any of the variants presented here.

REFERENCES

[1] Grey Ballard, Austin R. Benson, Alex Druinsky, Benjamin Lipshitz, and Oded Schwartz. 2016. Improving the Numerical Stability of Fast Matrix Multiplication. *SIAM J. Matrix Anal. Appl.* 37, 4 (2016), 1382–1418. https://doi.org/10.1137/15M1032168

[2] Grey Ballard, James Demmel, Olga Holtz, Benjamin Lipshitz, and Oded Schwartz. 2012. Communication-Optimal Parallel Algorithm for Strassen's Matrix Multiplication. In *Proceedings of the Twenty-Fourth Annual ACM Symposium on Parallelism in Algorithms and Architectures* (Pittsburgh, Pennsylvania, USA) *(SPAA '12)*. Association for Computing Machinery, New York, NY, USA, 193–204. https://doi.org/10.1145/2312005.2312044

[3] Gal Beniamini and Oded Schwartz. 2019. Fast Matrix multiplication via sparse decomposition. In *SPAA '19: Proceedings of the 31st annual ACM symposium on Parallel algorithms and architectures* (Phoenix, Arizona, USA), Petra Berenbrink and Christian Scheideler (Eds.). Association for Computing Machinery, Phoenix, Arizona, USA, 11–22. https://doi.org/10.1145/3323165.3323188

[4] Dario Andrea Bini and Grazia Lotti. 1980. Stability of fast algorithms for matrix multiplication. *Numer. Math.* 36, 1 (March 1980), 63–72. https://doi.org/10.1007/BF01395989

[5] Alin Bostan, Grégoire Lecerf, and Éric Schost. 2003. Tellegen's principle into practice. In *ACM International Symposium on Symbolic and Algebraic Computation, Philadelphia, USA*, Rafael Sendra (Ed.). ACM Press, New York, 37–44. https://doi.org/10.1145/860854.860870

[6] Joan Boyar, Philip Matthews, and René Peralta. 2013. Logic Minimization Techniques with Applications to Cryptology. *Journal of Cryptology* 26, 2 (2013), 280–312. https://doi.org/10.1007/s00145-012-9124-7

[7] R. P. Brent. 1970. *Algorithms for matrix multiplication.* Technical Report STAN-CS-70-157. C.S. Dpt. Standford University.

[8] Zhen Dai and Lek-Heng Lim. 2023. Numerical stability and tensor nuclear norm. *Numer. Math.* 155, 3-4 (Nov. 2023), 345–376. https://doi.org/10.1007/s00211-023-01377-5

[9] James Demmel, Ioana Dumitriu, and Olga Holtz. 2007. Fast linear algebra is stable. *Numer. Math.* 108, 1 (2007), 59–91. https://doi.org/10.1007/S00211-007-0114-X

[10] James Demmel, Ioana Dumitriu, Olga Holtz, and Robert Kleinberg. 2007. Fast matrix multiplication is stable. *Numer. Math.* 106, 2 (Feb. 2007), 199–224. https://doi.org/10.1007/s00211-007-0061-6

[11] Jean-Guillaume Dumas, Bruno Grenet, Clément Pernet, and Alexandre Sedoglavic. 2024. PLinOpt, a collection of C++ routines handling linear & bilinear programs. https://github.com/jgdumas/plinopt 1.7 kSLOC.

[12] Jean-Guillaume Dumas, Clément Pernet, and Alexandre Sedoglavic. 2024. Matlab accurate fast matrix multiplications via 2x2 recursion. https://github.com/jgdumas/Fast-Matrix-Multiplication 1.6 kSLOC.

[13] Hans-Friedich de Groote. 1978. On varieties of optimal algorithms for the computation of bilinear mappings I. The isotropy group of a bilinear mapping. *Theoretical Computer Science* 7, 2 (1978), 1–24. https://doi.org/10.1016/0304-3975(78)90038-5

[14] Hans-Friedich de Groote. 1978. On varieties of optimal algorithms for the computation of bilinear mappings II. Optimal algorithms for 2 × 2-matrix multiplication. *Theoretical Computer Science* 7, 2 (1978), 127–148. https://doi.org/10.1016/0304-3975(78)90045-2

[15] Nicholas J. Higham. 2002. *Accuracy and Stability of Numerical Algorithms* (second ed.). Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898718027

[16] John Edward Hopcroft and Jean Musinski. 1973. Duality Applied to the Complexity of Matrix Multiplications and other Bilinear Forms. In *5th Annual ACM Symposium on the theory of computing* (Austin, TX, USA), Alfred V. Aho, Allan Borodin, Robert L. Constable, Robert W. Floyd, Michael A. Harrison, Richard D. Karp, and Raymond H. Strong (Eds.). 73–87. https://doi.org/10.1137/0202013

[17] Elaye Karstadt and Oded Schwartz. 2017. Matrix Multiplication, a Little Faster. In *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures* (Washington, DC, USA) *(SPAA '17)*, Christian Scheideler and Mohammad Hajiaghayi (Eds.). ACM, New York, NY, USA, 101–110. https://doi.org/10.1145/3087556.3087579

[18] Joseph M. Landsberg. 2016. *Geometry and complexity theory.* Cambridge Studies in Advanced Mathematics, Vol. 169. Cambrigde University Press. https://doi.org/10.1017/9781108183192

[19] Katsuhisa Ozaki, Takeshi Ogita, Shin'ichi Oishi, and Siegfried M. Rump. 2012. Error-free transformations of matrix multiplication by using fast routines of matrix multiplication and its applications. *Numer. Algorithms* 59, 1 (2012), 95–118. https://doi.org/10.1007/S11075-011-9478-1

[20] Volker Strassen. 1969. Gaussian elimination is not optimal. *Numer. Math.* 13, 4 (Aug. 1969), 354–356. https://doi.org/10.1007/BF02165411

[21] Shmuel Winograd. 1977. La complexité des calculs numériques. *La Recherche* 8 (1977), 956–963.

A SUPPLEMENTARY MATERIALS

A.1 Computational proofs

We gather in this section proofs of several proposition that are simple computations on objects presented in our work.

Proof of Proposition 15. To simplify our computations, we use the following coordinates $\rho = \sqrt[4]{4/r}$ and $\xi = (x-1)/2$, the matrix $u(\rho,\xi)$ and the associated isotropy $\mathrm{g}_{\rho,\xi} \diamond \mathcal{S}$. In that case, the explicit expression of the $\gamma_{2,1}$ growth factor $\gamma\left(\mathrm{g}_{r,x} \diamond \mathcal{S}\right)$ along this orbit is $2\sqrt{2} + 3\,\mathcal{A}$ with $\mathcal{A}(r,x)$ equal to $\left((1+x)^2 + r\right)\left((x-1)^2 + r\right)/r\sqrt{r}$. To conclude, we prove that the minimum of $\mathcal{A}(r,x)$ in $\mathbb{R}^+ \times \mathbb{R}$ is $16/3\sqrt{3}$. The partial derivatives of $\mathcal{A}(r,x)$ w.r.t. r and x are

$$\frac{\partial \mathcal{A}}{\partial x} = 4\frac{(x^2+r-1)x}{r^{3/2}},\ \frac{\partial \mathcal{A}}{\partial r} = \frac{r^2 - 2\left(x^2+1\right)r - 3\left(x^2-1\right)^2}{2r^{5/2}}. \tag{38}$$

First, notice that $\frac{\partial \mathcal{A}}{\partial x}\left(1-x^2, x\right)$ is 0 and that $\frac{\partial \mathcal{A}}{\partial r}\left(1-x^2, x\right)$ is equal to $2/\left((x-1)\,(1+x)\right)^{3/2}$. The only critical point is x equal to 0 and, as r is positive, it only could be equal to 3. The Hessian matrix is:

$$H(\mathcal{A}(r,x)) = \frac{1}{r^{\frac{3}{2}}}\begin{bmatrix} 4\left(3x^2+r-1\right) & -\frac{2}{r}\left(3x^2+r-3\right) \\ -\frac{2}{r}\left(3x^2+r-3\right) & -\frac{r^2-6\left(x^2+1\right)r-15\left(x^2-1\right)^2}{4r^2} \end{bmatrix}, \tag{39}$$

one can notice that $H(\mathcal{A}(3,0))$ is equal to $\begin{bmatrix} 2^3 & 0 \\ 0 & 2/3 \end{bmatrix}/3\sqrt{3}$. Hence, the second partial derivative test states that $\left(\sqrt[4]{4/3}, -1/2\right)$ is a local minimum of $\gamma\left(\mathrm{g}_{r,x} \diamond \mathcal{S}\right)$ equal to $2\sqrt{2} + \frac{16}{\sqrt{3}} \approx 12.06603143$. To conclude, the $\gamma_{2,1}$ growth factor reaches its global minimal at this point on the considered orbit because it is its only critical point. □

Proof of Proposition 17. Recall that any of the three matrices in the HM representation of such an algorithm is obtained by row and column permutations of one of these matrices, multiplied by the Kronecker product of two invertible 2×2 matrices W and V (see Lemma 6 and Theorem 7). From the analysis of Section 4, we only need to consider the case where-W and V are each of the form $\left[\begin{smallmatrix} r & x \\ 0 & r^{-1} \end{smallmatrix}\right]$, with strictly positive r (matrices W and V are taken in a simpler form for the sake of simplicity). For this, we let W be $\left[\begin{smallmatrix} r & x \\ 0 & r^{-1} \end{smallmatrix}\right]$ and V be $\left[\begin{smallmatrix} s & y \\ 0 & s^{-1} \end{smallmatrix}\right]$, and choose L the first component of Strassen's HM representation given in Eq. (6). We then obtain the square of the Frobenius norm of $\mathsf{L} \cdot (\mathsf{W} \otimes \mathsf{V})$ as a function of r, x, s and y: $f(r,x,s,y) = 4r^2s^2 + 3r^2y^2 + 3s^2x^2 + x^2y^2 + r^2/s^2 + s^2/r^2 + 1/r^2s^2$ $+ (x/s + 1/rs)^2 + (y/r - 1/rs)^2 + (x/s - xy)^2 + (y/r + xy)^2$ $+ (xy + 1/rs)^2 + (s/r + xs)^2 + (r/s - ry)^2$. Solving the four partial derivatives of the gradient $\left[\frac{\partial f}{\partial r}, \frac{\partial f}{\partial s}, \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$, for a simultaneous zero, we obtain that the only real extrema of f are at the four points: $r = \pm\sqrt[4]{3/4}, s = \pm r, x = -2/3r^3, y = 2/3s^3$, for which its value is always 10. The additional constraint that r and s are positive, thus gives a single extremum. Now, the Hessian matrix at that point is computed as: $\mathrm{H}_f\left(\frac{\sqrt[4]{3}}{\sqrt{2}}, -\frac{\sqrt[4]{3}}{\sqrt{6}}, \frac{\sqrt[4]{3}}{\sqrt{2}}, \frac{\sqrt[4]{3}}{\sqrt{6}}\right) = \frac{4}{9}\begin{bmatrix} \frac{195}{\sqrt{3}} & \frac{69}{\sqrt{3}} & -15 & 15 \\ \frac{69}{\sqrt{3}} & \frac{195}{\sqrt{3}} & -15 & 15 \\ -15 & -15 & \frac{45}{\sqrt{3}} & \frac{9}{\sqrt{3}} \\ 15 & 15 & \frac{9}{\sqrt{3}} & \frac{45}{\sqrt{3}} \end{bmatrix}$.

Leaving out the factor $4/9$, the characteristic polynomial of this matrix is $X^4 - 160\sqrt{3}X^3 + 22536X^2 - 362880\sqrt{3}X + 5143824$ whose all four roots $\sqrt{7500} \pm \sqrt{5232}$, $(30 \pm 12\sqrt{3}$ are positive. Therefore the point is a local minimum. From this, we see that $\sqrt{10}^3$ is a lower

bound on the product of their Frobenius norms. Furthermore, remark that this lower bound is reached, by the algorithm which HM representation is given in Equation (34). □

A.2 Rational approximations

As stated in Remark 23, by approximating the minimal point of the $\gamma_{2,1}$ growth factor presented in Proposition 15, we could construct further algorithms presented in this section.

$$\begin{bmatrix} \frac{4}{9} & -\frac{8}{9} & -\frac{8}{9} & -\frac{4}{9} \\ 0 & \frac{5}{9} & 0 & \frac{10}{9} \\ \frac{8}{9} & -\frac{2}{3} & 0 & 0 \\ \frac{4}{9} & \frac{2}{9} & \frac{8}{9} & \frac{4}{9} \\ 0 & -\frac{10}{9} & 0 & 0 \\ \frac{4}{9} & -\frac{1}{3} & -\frac{8}{9} & \frac{2}{3} \\ -\frac{4}{9} & -\frac{2}{9} & \frac{8}{9} & \frac{4}{9} \end{bmatrix} ; \begin{bmatrix} -\frac{3}{5} & \frac{4}{5} & -\frac{4}{5} & -\frac{3}{5} \\ 0 & \frac{1}{2} & 0 & -1 \\ -1 & \frac{1}{2} & 0 & 0 \\ 0 & \frac{5}{4} & 0 & 0 \\ \frac{3}{5} & -\frac{3}{10} & \frac{4}{5} & -\frac{2}{5} \\ \frac{2}{5} & \frac{3}{10} & -\frac{4}{5} & -\frac{3}{5} \\ -\frac{3}{5} & -\frac{9}{20} & -\frac{4}{5} & -\frac{3}{5} \end{bmatrix} ; \begin{bmatrix} \frac{9}{20} & \frac{9}{10} & \frac{9}{10} & -\frac{9}{20} \\ 0 & 0 & \frac{27}{40} & -\frac{9}{10} \\ -\frac{9}{8} & 0 & -\frac{9}{16} & 0 \\ \frac{9}{20} & \frac{9}{10} & \frac{9}{40} & \frac{9}{20} \\ -\frac{27}{40} & \frac{9}{10} & \frac{27}{80} & -\frac{9}{20} \\ 0 & 0 & -\frac{9}{8} & 0 \\ \frac{9}{20} & \frac{9}{10} & -\frac{9}{40} & -\frac{9}{20} \end{bmatrix}^{\mathsf{T}} \tag{40}$$

First, Eq. (40) is an orthogonal optimization of Eq. (35), with one canonical vector in each of components of the HM representation. Unfortunately some small non-powers of 2 are then unavoidable, but this gives in Table 6 an algorithm realizing the formula with less additions than that of Table 4.

$$\begin{array}{llll}
u_1=a_{12}+\frac{1}{2}a_{22} & t_2=\frac{8}{9}a_{11}-\frac{2}{3}a_{12} & t_1=\frac{5}{9}u_1 & t_4=\frac{10}{9}a_{12} \\
t_3=\frac{4}{9}a_{11}+\frac{8}{9}a_{21}+\frac{2}{9}u_1 & t_0=t_2-t_3 & t_5=t_1+t_0 & t_6=t_4+t_0 \\
v_1=\frac{1}{2}b_{12} & s_1=v_1-b_{22} & s_2=v_1-b_{11} & s_3=\frac{5}{4}b_{12} \\
s_4=\frac{2}{5}b_{22}-\frac{4}{5}b_{21}+\frac{3}{5}s_2 & s_0=s_1+s_4 & s_5=s_0-s_2 & s_6=s_3-s_0 \\
\multicolumn{4}{c}{p_0=t_0\cdot s_0 \quad p_1=t_1\cdot s_1 \quad p_2=t_2\cdot s_2 \quad p_3=t_3\cdot s_3} \\
\multicolumn{4}{c}{p_4=t_4\cdot s_4 \quad p_5=t_5\cdot s_5 \quad p_6=t_6\cdot s_6} \\
w_1=p_6+p_0+p_4 & w_2=p_5+p_6 & w_3=p_3+w_1 & w_4=p_2+p_4 \\
w_5=p_1+w_1 & w_6=\frac{9}{20}w_3 & & c_{11}=w_6-\frac{9}{8}w_4 \\
c_{12}=\frac{9}{10}w_3 & c_{21}=\frac{27}{40}w_5-\frac{9}{8}w_2+\frac{1}{2}c_{11} & & c_{22}=w_6-\frac{9}{10}w_5
\end{array}$$

Table 6: SLP of Eq. (40), $\gamma_F \approx 12.2034$, with 24 add. and 20 mul.

Finally, we present in Eqs. (41) and (42), successive higher-order rational approximations of the point $(\sqrt[4]{4/3}, -1/2)$ reducing the $\gamma_{2,1}$ growth factor to 12.0695 (resp. 12.0661), approaching 12.06603. They then provide rational algorithms whose accuracy is pretty close to our best one, as shown in Figure 1.

$$\begin{bmatrix} -\frac{167042}{345665} & \frac{295936}{345665} & -\frac{295936}{345665} & -\frac{167042}{345665} \\ -\frac{178623}{345665} & -\frac{51622047}{176980480} & \frac{295936}{345665} & \frac{167042}{345665} \\ 0 & -\frac{51622047}{88490240} & 0 & \frac{334084}{345665} \\ -1 & \frac{289}{512} & 0 & 0 \\ 0 & \frac{289}{256} & 0 & 0 \\ -\frac{167042}{345665} & -\frac{24137569}{88490240} & -\frac{295936}{345665} & -\frac{167042}{345665} \\ -\frac{167042}{345665} & \frac{24137569}{88490240} & -\frac{295936}{345665} & \frac{167042}{345665} \end{bmatrix} ; \begin{bmatrix} -\frac{256}{289} & -\frac{1}{2} & \frac{1}{2} & -\frac{256}{289} \\ -\frac{345665}{295936} & 0 & 0 & 0 \\ -\frac{345665}{591872} & 0 & \frac{345665}{334084} & 0 \\ -\frac{178623}{295936} & -1 & 0 & 0 \\ \frac{178623}{591872} & \frac{1}{2} & \frac{178623}{334084} & \frac{256}{289} \\ -\frac{289}{1024} & \frac{1}{2} & -\frac{1}{2} & \frac{256}{289} \\ -\frac{289}{1024} & \frac{1}{2} & \frac{1}{2} & -\frac{256}{289} \end{bmatrix} \tag{41}$$

$$\begin{bmatrix} \frac{295936}{345665} & \frac{295936}{345665} & 0 & 0 & \frac{295936}{345665} & \frac{295936}{345665} & 0 \\ \frac{178623}{345665} & -\frac{167042}{345665} & 1 & 0 & \frac{178623}{345665} & \frac{178623}{345665} & 0 \\ -\frac{178623}{345665} & -\frac{178623}{345665} & 0 & 1 & \frac{167042}{345665} & -\frac{178623}{345665} & 0 \\ \frac{295936}{345665} & \frac{51622047}{176980480} & \frac{289}{512} & -\frac{289}{512} & \frac{51622047}{176980480} & -\frac{31906176129}{102294717440} & -\frac{345665}{295936} \end{bmatrix}$$

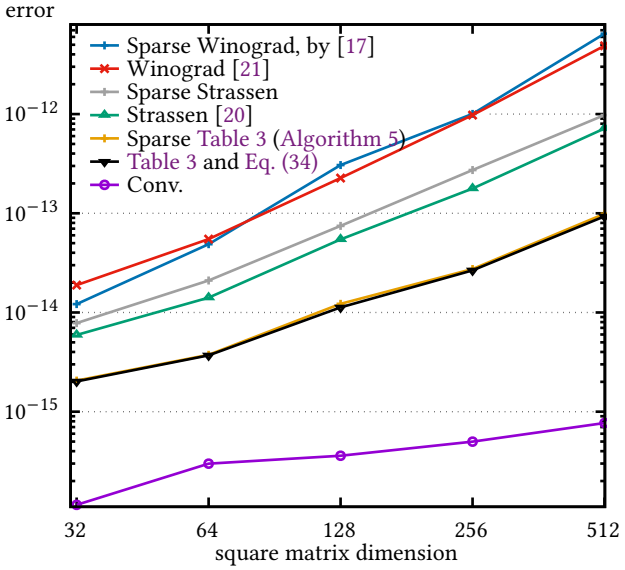
$$\begin{bmatrix} \frac{33124}{38165} & \frac{19208}{38165} & -\frac{19208}{38165} & \frac{33124}{38165} \\ \frac{33124}{38165} & \frac{19208}{38165} & \frac{18957}{38165} & \frac{1857786}{6449885} \\ 0 & \frac{38416}{38165} & 0 & \frac{3715572}{6449885} \\ 0 & 0 & 1 & -\frac{98}{169} \\ 0 & 0 & 0 & \frac{196}{169} \\ \frac{33124}{38165} & \frac{19208}{38165} & -\frac{19208}{38165} & -\frac{1882384}{6449885} \\ \frac{33124}{38165} & -\frac{19208}{38165} & -\frac{19208}{38165} & \frac{1882384}{6449885} \end{bmatrix} ; \begin{bmatrix} -\frac{169}{196} & -\frac{1}{2} & \frac{1}{2} & -\frac{169}{196} \\ \frac{38165}{33124} & 0 & 0 & 0 \\ \frac{38165}{66248} & 0 & -\frac{38165}{38416} & 0 \\ \frac{18957}{33124} & 1 & 0 & 0 \\ \frac{18957}{66248} & \frac{1}{2} & \frac{18957}{38416} & \frac{169}{196} \\ -\frac{49}{169} & \frac{1}{2} & -\frac{1}{2} & \frac{169}{196} \\ -\frac{49}{169} & \frac{1}{2} & \frac{1}{2} & -\frac{169}{196} \end{bmatrix} \tag{42}$$

$$\begin{bmatrix} -\frac{18957}{38165} & \frac{19208}{38165} & -1 & 0 & -\frac{18957}{38165} & -\frac{18957}{38165} & 0 \\ -\frac{33124}{38165} & -\frac{1857786}{6449885} & -\frac{98}{169} & \frac{98}{169} & -\frac{1857786}{6449885} & \frac{359367849}{1264177460} & \frac{38165}{33124} \\ \frac{33124}{38165} & \frac{33124}{38165} & 0 & 0 & \frac{33124}{38165} & \frac{33124}{38165} & 0 \\ -\frac{18957}{38165} & -\frac{18957}{38165} & 0 & 1 & \frac{19208}{38165} & -\frac{18957}{38165} & 0 \end{bmatrix}$$

A.3 Further numerical experiments

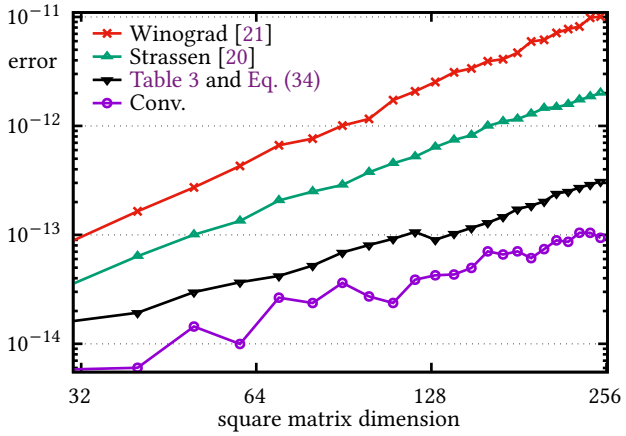
Following [19, § 3.2], we study in Figure 3 the effect of sparsification on random matrix with preassigned singular values and large condition number $\approx 10^{12}$ given by the Matlab function gallery 'randsvd'. The fast variants behavior is unchanged while only the conventional algorithm performs better.

Figure 3: Numerical Effect of Sparsification (large conditioning)



We now show more evidence on the practical accuracy of the algorithms, with respect to their $\gamma_{2,1}$ growth factor. Figure 4 compares the main possibilities on a uniform $[-1, 1]$ distribution, while Figure 1 was using a normal distribution. The behavior is similar, with again our best variant one or two orders of magnitude more accurate, and being quite close to that of the conventional algorithm.

Figure 4: Numerical accuracy for uniform [-1,1] distribution



A.4 Cancellation-free search

Algorithm 6 describes a common sub-expression elimination heuristic that reduced the number of operations in our algorithms.

Algorithm 6 Cancellation-free optimization of a linear operator

Input: $M \in \mathbb{K}^{m \times n}$.
Output: A straight-line program computing $x \longrightarrow M \cdot x$.

repeat ▷ Precomputing all repeated pairs
    In each row list all pairs of indices of non-zero coefficients;
    Among all the rows, find the pair(s) with the maximal number of co-linear representatives;
    In case of ties, exhaust all the possibilities with maximal pairs (or choose one using a score like that of [6, § 3.2]);
    Precompute the chosen pair (in a temporary variable);
    Factor this pair out of all the rows: that is remove the pair from all rows but add a new column to the matrix (representing that pair) with the co-linear multiple of that temporary variable;
until no pair has more than 1 representative
▷ Multipliers by columns:
for all equal coefficients in a column (up to sign) **do**
    Compute the product by the absolute value in a temporary variable;
    Factor this coefficient out: remove it from the column, add a new column (representing that product) with a ±1 in the corresponding row(s);
▷ Multipliers by rows:
for all equal coefficients in a row (up to sign) **do**
    Compute the sum (or subtraction) of variables with that same coefficients in a temporary variable;
    Factor the coefficient out: remove it from the row, but add a new column (representing that sum/subtraction) with the coefficient in the same row;
▷ Now the matrix has been simplified
Apply the remaining linear operations of the matrix.