

The exponent of matrix multiplication is less than 2.37116

Mateusz Michalek

5 October 2026

Full version

Abstract

We show that the exponent ω of matrix multiplication satisfies $\omega < 2.37116$. The bound $\omega < 2.371177$ of Dupont, Eisenberger, Kozlovskii, Mehrabian, Ruiz, See, Zhou, Alman, Vassilevska Williams and Balog comes from a feasible point of an optimisation problem. This problem encodes the analysis of Alman, Duan, Vassilevska Williams, Xu, Xu and Zhou, applied to the eighth Kronecker power of the Coppersmith–Winograd tensor CW_5 . We exhibit a better feasible point of the same problem. The point was found by a minimax sequential quadratic programming method. Its feasibility is proved by computation, with two independently written programs: one works in exact rational arithmetic and ball arithmetic, the other in floating-point interval arithmetic. While this note was being written, an author writing under the name BSD announced bounds down to 2.3711694, obtained from other points of the same problem; our work is independent of that work.

1 Introduction

1.1

Let F be a field. The *exponent of matrix multiplication*, denoted by ω , is the infimum of the real numbers τ such that two $n \times n$ matrices over F can be multiplied with $O(n^\tau)$ arithmetic operations. The definition of the product gives $\omega \leq 3$, and $\omega \geq 2$ because the product has n^2 entries. In 1969 Strassen showed that $\omega \leq \log_2 7 < 2.81$ [31]. The following question has been open since then.

What is the value of ω ? Is $\omega = 2$?

During the following two decades the bound was lowered several times, by Pan [27], Bini, Capovani, Romani and Lotti [4], Schönhage [29], Romani [28], Coppersmith and Winograd [11] and Strassen [32]. Two ideas of that period underlie all later work: the asymptotic sum inequality of Schönhage [29], which bounds ω in terms of the border rank of a direct sum of matrix multiplication tensors, and the laser method of Strassen [32, 33], which finds such direct sums inside the Kronecker powers of a suitable tensor of small border rank. Coppersmith and Winograd [12] introduced the tensors that are now denoted by CW_q , applied the laser method to CW_q and to its square, and proved $\omega < 2.376$. Every later bound has been obtained by applying the laser method to higher Kronecker powers of CW_q and by refining the method; see Table 1, and [9, 5] for the background.

year	authors	bound on ω
1969	Strassen [31]	2.8074
1990	Coppersmith, Winograd [12]	2.3755
2010	Stothers [30, 13]	2.3737
2012	Vassilevska Williams [35]	2.3729
2014	Le Gall [23]	2.3728639
2021	Alman, Vassilevska Williams [2]	2.3728596
2023	Duan, Wu, Zhou [15]	2.371866
2024	Vassilevska Williams, Xu, Xu, Zhou [37]	2.371552
2025	Alman, Duan, Vassilevska Williams, Xu, Xu, Zhou [1]	2.371339
2026	Dupont, Eisenberger, Kozlovskii, Mehrabian, Ruiz, See, Zhou, Alman, Vassilevska Williams, Balog [16]	2.371177
2026	BSD [8] (announced; see §1.2)	2.371170
2026	this note	2.37116

Table 1: Upper bounds on ω : Strassen’s, and those obtained from the tensors CW_q . The table lists the bounds that have appeared in journals, in conference proceedings or on arXiv, followed by the two bounds discussed in §1.2. The years are those of publication; the bound of [12] was announced in 1987. The conference paper [35] states 2.3727; the value 2.3729 is the corrected one, from its full version [36]; see also [23].

1.2

The bounds of Table 1 from 2023 onwards rest on one idea. Duan, Wu and Zhou [15] identified a loss in the way the laser method had been applied to higher powers, the *combination loss*, and showed how to compensate for part of it by an asymmetric hashing argument. Vassilevska Williams, Xu, Xu and Zhou [37] refined this argument, and Alman, Duan, Vassilevska Williams, Xu, Xu and Zhou [1] made it more asymmetric. The last four bounds of the table all come from the analysis of [1]. This analysis has a positive integer parameter ℓ^* , the *level*, and treats the tensor $CW_q^{\otimes 2^{\ell^*}-1}$. For each level it produces a finite-dimensional optimisation problem with the property that every feasible point gives an upper bound on ω . The authors of [1] found a feasible point numerically for $q = 5$ and $\ell^* = 3$, the fourth power, and obtained $\omega < 2.371339$. Dupont et al. [16] wrote the problem out in closed form, found a feasible point for $q = 5$ and $\ell^* = 4$, the eighth power, with a gradient method whose code was improved by AlphaEvolve [26], and obtained $\omega < 2.371177$. At the time of writing, this is the best bound in a journal, in conference proceedings or on arXiv.

On 4 October 2026, while this note was being written, an author writing under the name BSD announced [8] three bounds obtained from further points of the problem for $q = 5$ and $\ell^* = 4$, namely 2.371176826..., 2.371174034... and 2.371169352... They are stated for the exponent over the field $\mathbb{Q}$. The manuscript [8] describes the last of them, and all three are listed in the repository that contains it. According to [8], the bound is certified by a program in exact arithmetic and in the proof assistant Lean, where the numerical checks are evaluated by compiled code and the theorem that turns feasible points into bounds is taken as an axiom. At the time of writing, neither the points nor the certificates had been released. The work reported here was done independently of [8], and it gives a smaller bound.

We now state our main result.

Theorem 1.1. *The exponent of matrix multiplication satisfies*

$$\omega \leq \frac{592789949057}{250000000000} = 2.371159796228.$$

In particular, $\omega < 2.37116$.

Theorem 1.1 holds over every field. It is the conjunction of two statements.

- (i) *A theorem of [1]* (Theorem 3.1). For the optimisation problem of [1, 16], at any level, every point that is feasible together with a number Ω , in the sense of Definition 2.1, gives the bound $\omega \leq \Omega$. This is a special case of [16, Theorem 1], which is stated there without proof. It follows from published statements of [1], which we do not prove again; see §3.1.
- (ii) *A feasible point* (Proposition 4.2). For $q = 5$ and $\ell^* = 4$ there is a point that is feasible together with $\Omega = 2.371159796228$. The point is an explicit rational point, given by a file. That it is feasible is a statement about finitely many rational numbers, their logarithms and finitely many maximum entropies, and it is proved by computation.

Thus the mathematics behind Theorem 1.1 is that of [1], and what is new is the point. The bounds of [16] and of [8] have the same structure, with other points at the same level. Our bound is smaller than the bound 2.371177 of [16] by about $1.72 \cdot 10^{-5}$, and smaller than the best bound of [8] by about $9.6 \cdot 10^{-6}$.

1.3

How the point is certified. The optimisation problem is stated in full in §2. Its constraint is an inequality between sums of entropies of probability distributions that are rational functions of the variables. A first program computes all these distributions exactly over $\mathbb{Q}$, encloses every logarithm in ball arithmetic, bounds the one quantity that is not given by a closed formula, a maximum entropy, by weak duality, and checks the final inequality in $\mathbb{Q}$. A second program, written separately, encloses the same quantities by floating-point interval arithmetic with outward rounding and bounds the maximum entropies in a different way. Both accept the point. We also compared the first program, at level 3, with the MATLAB program published by the authors of [1], and at level 4 with a third evaluation, written separately from the text of [16] (§4.4).

How the point was found. The value of a point is a ratio in which twenty minima of three functions each appear. At the points that our optimisers reach, the three functions under each minimum, taken with the weights with which they enter the value, are almost equal; such points lie close to points at which the value is not differentiable. In our runs, gradient methods applied to a smoothed version of the value became very slow near them. We treat the minima exactly, by a sequential quadratic programming method for minimax problems that carries one multiplier for each of the sixty functions (§5). The search started from the published level-3 solution of [1], lifted to level 4.

What is not claimed. We do not know how far our point is from the optimum of the level-4 problem; when this note was written the search was still running, and the value was decreasing by about 10^{-9} per iteration. The point of [16] has not been released, so we can compare our point with it only through the published bound 2.371177. We make no claim about higher levels.

1.4

Organisation. Section 2 defines the optimisation problem. Section 3 states the theorem that turns feasible points into bounds and lists what it depends on. Section 4 defines the point and proves that it is feasible. Section 5 describes how the point was found, and Section 6 contains remarks. Appendix A lists the files, their checksums and the commands that reproduce the verification. Appendix B explains how the optimisation problem follows from the theorems of [1].

Note on the computations. The search for the point, the certification software, both verifying programs, the cross-checks of §4.4 and the drafting of this note were carried out by AI agents working under the author's direction. Everything can be run again from the files listed in Appendix A.

2 The optimisation problem

We follow [16, §2]. Throughout, $q \geq 1$ and $\ell^* \geq 2$ are fixed integers; in our application $q = 5$ and $\ell^* = 4$. The symbol $\log$ denotes the logarithm to a fixed base greater than 1, and entropies are taken to the same base. The base is 2 in [1, 16]; see Remark 2.3.

2.1 Shapes, regions and the tree

We call X, Y, Z the three *dimensions*. For an integer $\ell \geq 1$, a *level- ℓ shape* is a triple $s = (s_X, s_Y, s_Z)$ of nonnegative integers with $s_X + s_Y + s_Z = 2^\ell$. The set of level- ℓ shapes is denoted by $\mathcal{S}_\ell$. A shape is *positive* if its three coordinates are positive, and it is a *zero shape* otherwise. For $s \in \mathcal{S}_\ell$ with $\ell \geq 2$ we set

$$\text{Split}(s) = \{u \in \mathcal{S}_{\ell-1} : u_W \leq s_W \text{ for } W = X, Y, Z\}.$$

If $u \in \text{Split}(s)$ then $s - u \in \mathcal{S}_1$.

There are six *regions*, numbered $r = 1, \dots, 6$. Let π_r be the r -th permutation of the symbols X, Y, Z in lexicographic order, and write $(X_r, Y_r, Z_r) = (\pi_r(X), \pi_r(Y), \pi_r(Z))$. Thus $(X_1, Y_1, Z_1) = (X, Y, Z)$, $(X_2, Y_2, Z_2) = (X, Z, Y)$, and so on up to $(X_6, Y_6, Z_6) = (Z, Y, X)$. We say that the dimension X_r *plays the role of X in region r* , and similarly for Y_r and Z_r . The numbering of the six permutations is a convention: every node below has one child for each region, so that another numbering only renames the variables of the problem.

The problem is organised by a rooted tree. Its root is denoted by G . Every other node T has a *level* $\ell \in \{2, \dots, \ell^*\}$ and a *shape* $s_T \in \mathcal{S}_\ell$.

- The root has one child $G[s, r]$ for every shape $s \in \mathcal{S}_{\ell^*}$ and every region r . It has level ℓ^* and shape s .
- A node T of level $\ell \geq 3$ with positive shape has one child $T[u, r]$ for every $u \in \text{Split}(s_T)$ and every region r . It has level $\ell - 1$ and shape u .
- All other nodes are *leaves*: the nodes with a zero shape, and the nodes of level 2 with positive shape. The positive shapes of level 2 are $(1, 1, 2)$, $(1, 2, 1)$ and $(2, 1, 1)$.

For $\ell^* = 4$ the tree has 1 552 338 nodes besides the root: 918 of level 4, 64 260 of level 3 and 1 487 160 of level 2.

2.2 The variables

For a finite set D we write $\Delta(D)$ for the set of probability distributions on D , that is, of functions $p: D \rightarrow [0, 1]$ with $\sum_{x \in D} p(x) = 1$. For $\ell \geq 1$ and $0 \leq a \leq 2^\ell$ let

$$\mathcal{C}_{\ell,a} = \left\{ L \in \{0, 1, 2\}^{2^{\ell-1}} : \sum_p L_p = a \right\}.$$

An element of $\Delta(\mathcal{C}_{\ell,a})$ is called a level- ℓ *complete split distribution*. The variables of the problem are the following.

- (V1) At the root and at every node T of level at least 3 with positive shape: a distribution $A_T = (A_T^{(1)}, \dots, A_T^{(6)}) \in \Delta(\{1, \dots, 6\})$ and, for every region r , a distribution $\alpha_T^{(r)} \in \Delta(D_T)$, where $D_G = \mathcal{S}_{\ell^*}$ and $D_T = \text{Split}(s_T)$ for $T \neq G$.
- (V2) At every node T with a zero shape, of level ℓ : a complete split distribution $\beta_T \in \Delta(\mathcal{C}_{\ell,a})$, where $a = s_{T,W}$ and W is the first of the dimensions X, Y, Z with $s_{T,W} \neq 0$.
- (V3) At every node T of level 2 with positive shape: a number $\mu_T \in [0, \frac{1}{2}]$.

A *point* is a choice of values for all variables. The set of points is a product of simplices and intervals.

2.3 Masses and complete split distributions

From a point we derive the following quantities.

Masses. Every node other than the root has a *mass* $m_T \geq 0$, defined from the root downwards by

$$m_{G[s,r]} = A_G^{(r)} \alpha_G^{(r)}(s), \quad m_{T[u,r]} = m_T A_T^{(r)} (\alpha_T^{(r)}(u) + \alpha_T^{(r)}(s_T - u)). \quad (1)$$

Complete split distributions. Every node T other than the root, of level ℓ , has for each dimension W a complete split distribution $\beta_{T,W} \in \Delta(\mathcal{C}_{\ell,s_{T,W}})$, defined from the leaves upwards. We write δ_L for the point mass at L , and $\vec{0}$ and $\vec{2}$ for the vectors with all entries 0 and all entries 2.

- (a) Let T have a zero shape. Let W_0 be the first dimension with $s_{T,W_0} = 0$, let W_1 be the first dimension with $s_{T,W_1} \neq 0$, and let W_2 be the third dimension. Then

$$\beta_{T,W_0} = \delta_{\vec{0}}, \quad \beta_{T,W_1} = \beta_T, \quad \beta_{T,W_2}(\vec{2} - L) = \beta_T(L) \text{ for all } L.$$

- (b) Let T have level 2 and positive shape, and let W be the dimension with $s_{T,W} = 2$. Then

$$\beta_{T,W} = \mu_T \delta_{(0,2)} + \mu_T \delta_{(2,0)} + (1 - 2\mu_T) \delta_{(1,1)}, \quad \beta_{T,W'} = \frac{1}{2} \delta_{(0,1)} + \frac{1}{2} \delta_{(1,0)} \text{ for } W' \neq W.$$

- (c) Let T have level $\ell \geq 3$ and positive shape. For every region r and dimension W put

$$\beta_{T,W}^{(r)} = \sum_{u \in \text{Split}(s_T)} \alpha_T^{(r)}(u) \beta_{T[u,r],W} \times \beta_{T[s_T-u,r],W}, \quad \beta_{T,W} = \sum_{r=1}^6 A_T^{(r)} \beta_{T,W}^{(r)}. \quad (2)$$

Here $\beta \times \beta'$ is the distribution of the concatenation of two independent vectors with distributions β and β' : $(\beta \times \beta')(L \circ L') = \beta(L) \beta'(L')$.

2.4 Retained exponents

We write $H(p) = -\sum_x p(x) \log p(x)$ for the entropy of a distribution p , with $0 \log 0 = 0$, and $H(p_1, \dots, p_k)$ for the entropy of the distribution with masses $p_1, \dots, p_k$. For a set D of shapes and $\rho \in \Delta(D)$, let ρ_W be the marginal of ρ in the coordinate W , that is, $\rho_W(w) = \sum_{a \in D, a_W=w} \rho(a)$, and let

$$H_D^{\max}(\rho) = \max\{H(\rho') : \rho' \in \Delta(D), \rho'_W = \rho_W \text{ for } W = X, Y, Z\}. \quad (3)$$

Thus $H_D^{\max}(\rho) \geq H(\rho)$.

Let T be the root or a node of level at least 3 with positive shape, and let r be a region. We abbreviate $\alpha = \alpha_T^{(r)}$ and $D = D_T$, and we define a weight w on D and distributions b_W by

$$w(v) = \begin{cases} \alpha(v) & \text{if } T = G, \\ \alpha(v) + \alpha(s_T - v) & \text{if } T \neq G, \end{cases} \quad b_W = \begin{cases} \sum_{v \in D} \alpha(v) \beta_{G[v,r],W} & \text{if } T = G, \\ \beta_{T,W}^{(r)} & \text{if } T \neq G. \end{cases}$$

For $T = G$ the distribution b_W lives on $\{0, 1, 2\}^{2^{\ell^*}-1}$, the disjoint union of the sets $\mathcal{C}_{\ell^*,a}$. For a subset $U \subseteq D$ with $w(U) = \sum_{v \in U} w(v) > 0$ and a dimension W we write

$$\bar{\beta}_{U,W} = \frac{1}{w(U)} \sum_{v \in U} w(v) \beta_{T[v,r],W}$$

for the w -weighted average of the complete split distributions of the children $T[v, r]$, $v \in U$. We now define three numbers. The first is

$$x_T^{(r)} = H(\alpha_{X_r}) + H(\alpha) - H_D^{\max}(\alpha). \quad (4)$$

For the second, let $U_j = \{v \in D : v_{Y_r} = j, v_{Z_r} > 0\}$ for $j \geq 0$, and put

$$y_T^{(r)} = H(b_{Y_r}) - \sum_{v \in D: v_{Z_r}=0} w(v) H(\beta_{T[v,r],Y_r}) - \sum_{j \geq 0} w(U_j) H(\bar{\beta}_{U_j,Y_r}). \quad (5)$$

For the third, let $U'_k = \{v \in D : v_{X_r} > 0, v_{Y_r} > 0, v_{Z_r} = k\}$ for $k \geq 0$, and put

$$z_T^{(r)} = H(b_{Z_r}) - \sum_{v \in D: v_{X_r}=0 \text{ or } v_{Y_r}=0} w(v) H(\beta_{T[v,r],Z_r}) - \sum_{k \geq 0} w(U'_k) H(\bar{\beta}_{U'_k,Z_r}). \quad (6)$$

In (5) and (6) the terms with $w(U_j) = 0$ or $w(U'_k) = 0$ are omitted.

The *retained exponents* of the root, of the levels $\ell = 3, \dots, \ell^*$ and of level 2 are

$$E_G = \sum_{r=1}^6 A_G^{(r)} \min\{x_G^{(r)}, y_G^{(r)}, z_G^{(r)}\}, \quad (7)$$

$$E_\ell = \sum_{r=1}^6 \min\left\{\sum_T m_T A_T^{(r)} x_T^{(r)}, \sum_T m_T A_T^{(r)} y_T^{(r)}, \sum_T m_T A_T^{(r)} z_T^{(r)}\right\}, \quad (8)$$

$$E_2 = \min_{W \in \{X,Y,Z\}} \sum_T m_T e_{T,W}, \quad e_{T,W} = \begin{cases} H(\mu_T, \mu_T, 1 - 2\mu_T) & \text{if } s_{T,W} = 2, \\ \log 2 & \text{if } s_{T,W} = 1. \end{cases} \quad (9)$$

In (8) the sums are over the nodes T of level ℓ with positive shape, and in (9) over the nodes of level 2 with positive shape.

2.5 Matrix sizes and the problem

Every leaf T has three *local matrix sizes* $M_{T,X}, M_{T,Y}, M_{T,Z} \geq 0$. For a distribution β on vectors with entries in $\{0, 1, 2\}$ let $\nu(\beta) = \sum_L \beta(L) \cdot |\{p : L_p = 1\}|$ be the expected number of entries equal to 1.

- If T has a zero shape and W_0 is the first dimension with $s_{T,W_0} = 0$, then

$$M_{T,W_0} = m_T(H(\beta_T) + \nu(\beta_T) \log q), \quad M_{T,W} = 0 \text{ for } W \neq W_0.$$

- If T has level 2 and positive shape, then

$$M_{T,W} = \begin{cases} 2\mu_T m_T \log q & \text{if } s_{T,W} = 2, \\ (1 - 2\mu_T) m_T \log q & \text{if } s_{T,W} = 1. \end{cases}$$

We set

$$E_{\text{total}} = E_G + \sum_{\ell=3}^{\ell^*} E_\ell + E_2, \quad M_W = \sum_{T \text{ leaf}} M_{T,W}, \quad M_{\text{total}} = \min\{M_X, M_Y, M_Z\}. \quad (10)$$

Definition 2.1. A point together with a real number Ω is *feasible* if the following three conditions hold.

(F1) $M_{\text{total}} > 0$.

(F2) Each of the minima in (7), (8) and (9) is positive.

(F3) The inequality

$$E_{\text{total}} + \Omega \cdot M_{\text{total}} \geq 2^{\ell^*-1} \log(q+2) \quad (11)$$

holds.

The *value* of a point that satisfies (F1) and (F2) is the smallest Ω for which (F3) holds, namely $(2^{\ell^*-1} \log(q+2) - E_{\text{total}})/M_{\text{total}}$.

The optimisation problem considered in this note is to minimise the value over the points that satisfy (F1) and (F2). There are $6(\ell^* - 1) + 1$ minima in (F2): six in E_G , six in each E_ℓ , and one in E_2 .

Remark 2.2 (comparison with [16]). Formulas (1)–(11) are those of [16, §§2.1–2.4], in a slightly different notation. Five points should be mentioned. First, for $T \neq G$ the number $m_T A_T^{(r)} x_T^{(r)}$ is denoted there by $E_{T,X_r}^{(r)}$, and similarly for y and z ; the terms subtracted in (5) and (6) are denoted by $\eta_{T,Y_r}^{(r)}$ and $\eta_{T,Z_r}^{(r)}$; and $H_D^{\max}(\alpha) - H(\alpha)$ is the *penalty* $P_D(\alpha)$. Second, for $T \neq G$ the weight of the average in (5) is written in [16] as $\alpha(*, j, +) + \alpha(*, s_Y - j, <) = \sum_{v \in U_j} \alpha(v) + \sum_{v: v_Y = s_Y - j, v_Z < s_Z} \alpha(v)$ (for $r = 1$). This equals $w(U_j)$, because $v \mapsto s_T - v$ maps the second index set bijectively onto U_j ; the same holds for (6). Third, [16] use logarithms to base 2, for which the constant $\log 2$ in (9) is 1. Fourth, in [16] a point and a number Ω are called feasible if (F3) holds, and the problem is to minimise Ω over all feasible pairs; the conditions (F1) and (F2) are not imposed there. Our definition is more restrictive, for the reasons given in Remark 3.4. Fifth, in [16] the masses are described as numbers in $[0, 1]$ attached to the nodes with positive shape. In fact the leaves with a zero shape need masses as well, for their matrix sizes, and masses can exceed 1: by (1) a child

$T[u, r]$ with $u = s_T - u$ has mass $2m_T A_T^{(r)} \alpha_T^{(r)}(u)$, and the point of §4.4(d) has nodes of mass 2 and 4. Nothing depends on this. The remaining differences are notational: [16] writes a supremum in (3), uses the convention $0 \cdot \text{undefined} = 0$ where we omit terms, and gives explicit ranges for the indices j and k in (5) and (6).

Remark 2.3 (base of the logarithm). If the base of the logarithm is changed, then E_{total} , M_{total} and $\log(q+2)$ are multiplied by the same positive constant. Hence feasibility and the value of a point do not depend on the base. The certificate of §4 uses natural logarithms, for which the right-hand side of (11) is $8 \ln 7$ when $q = 5$ and $\ell^* = 4$.

Remark 2.4 (meaning). The following dictionary is not needed for the proofs of this note, but it explains the problem; it is made precise in Appendix B. The tensor $\text{CW}_q^{\otimes 2^{\ell-1}}$ is a sum of *constituent tensors* $T_{i,j,k}$ indexed by the level- ℓ shapes (i, j, k) . A node T of level ℓ stands for the tensor power $T_{s_T}^{\otimes m_T n}$, with $n \rightarrow \infty$, restricted to the variables whose finest index sequences have the statistics $\beta_{T,X}$, $\beta_{T,Y}$, $\beta_{T,Z}$. At a node with positive shape the laser method is applied once more: the power is divided into six regions with proportions $A_T^{(r)}$, in region r the three dimensions are treated asymmetrically in the order X_r, Y_r, Z_r , and $\alpha_T^{(r)}$ is the distribution according to which the constituent tensor is split into two constituent tensors of the level below. The three numbers under each minimum are the exponential rates at which blocks of the three dimensions survive the hashing and the zeroing-out steps. A leaf with a zero shape is a matrix multiplication tensor in which two of the three sizes are equal to 1, and its local matrix size is the logarithm of the third size per unit of n . In the end one has, with logarithms to base 2, at least $2^{E_{\text{total}}n - o(n)}$ independent copies of a matrix multiplication tensor of format $\langle 2^{M_Y n}, 2^{M_Z n}, 2^{M_X n} \rangle$ up to factors $2^{o(n)}$, and (11) is what Schönhage's asymptotic sum inequality requires of them.

3 The theorem behind the problem

Theorem 3.1 (Alman, Duan, Vassilevska Williams, Xu, Xu, Zhou [1]; see [16, Theorem 1]). *Let $q \geq 1$ and $\ell^* \geq 2$ be integers. If a point and a real number Ω are feasible in the sense of Definition 2.1, then $\omega \leq \Omega$.*

3.1

We explain where Theorem 3.1 is proved. The paper [1] shows how a direct sum of copies of $\text{CW}_q^{\otimes 2^{\ell^*-1}n}$ degenerates into a direct sum of matrix multiplication tensors, in $\ell^* + 1$ stages: a *global stage* [1, Theorem 5.3], *constituent stages* for the levels $\ell^*, \dots, 2$ [1, Theorem 6.4], and the identification of the terms that have a zero coordinate with matrix multiplication tensors [1, Theorem 6.2]. Each stage has parameters, and the number of copies and the sizes that it produces are explicit functions of them. The stages are combined in [1, Algorithm 1], and Schönhage's asymptotic sum inequality [29] then gives a bound on ω in terms of the parameters. The search for the best parameters is formulated in [1, §7] as an optimisation problem, whose constraints are those of the three theorems together with the inequality at the end of [1, Algorithm 1].

In [16, §2] the parameters of all stages are arranged on the tree of §2.1, the consistency conditions between consecutive stages are solved, and the result is the closed form reproduced in §2. The statement [16, Theorem 1] is Theorem 3.1 without the conditions (F1) and (F2); it is given there with attribution to [1] and without proof. Three of the authors of [16] are authors of [1], and the current version of [1] records that the parameters of [16], substituted into its analysis, give $\omega < 2.371177$. Since no derivation of the closed form is in print, we give one in Appendix B.

We show there that a point which is feasible together with Ω , in the sense of Definition 2.1, supplies admissible parameters for every stage of [1, Algorithm 1], and that the asymptotic sum inequality then gives $\omega \leq \Omega$.

Remark 3.2 (what Theorem 1.1 depends on). Theorem 1.1 depends on Theorem 3.1 in the same way as the bounds of [1] and [16] do. Theorem 3.1 in turn rests on the following statements, which we use as published; the derivation in Appendix B uses the statements in (b) only for parameters at which every region of positive weight has a positive exponent.

- (a) The bound on the asymptotic rank of the Coppersmith–Winograd tensor, $\tilde{R}(CW_q) \leq q + 2$ [12], and the asymptotic sum inequality of Schönhage [29], in the form [1, Theorem 3.2].
- (b) The global stage and the constituent stage, [1, Theorems 5.3 and 6.4]. In [1] these are deduced from [1, Propositions 5.1 and 6.3], the corresponding statements in which the interface tensors produced have $\varepsilon = 0$. The proofs of the propositions are given there, with several steps referred to [37]; the deduction of the theorems from the propositions is omitted as being similar to the proofs of [37, Theorems 5.3 and 6.3].
- (c) The theorem on terms with a zero coordinate, [1, Theorem 6.2]. It is quoted there from [37], where it is Theorem 6.1 of the arXiv version. The proofs of the statements in (b) also use the lemma on fixing holes, [1, Theorem 4.2], which is Corollary 4.2 of the arXiv version of [37] (it is cited in [1] as Corollary 3.2).

We use Theorem 3.1 only for the exponent ω of square matrix multiplication; in the notation of [1], which treats rectangular matrix multiplication as well, this is the case $\kappa = 1$.

A formalisation in the proof assistant Lean of the bound $\omega < 2.371339$ of [1], that is, of the case $\ell^* = 3$ with the solution published there, has been made public by Zuiddam [38]. We have not examined it, and we do not know whether its general theorems cover the case $\ell^* = 4$.

Remark 3.3. In [16, §2] the tensor underlying the problem is written as $CW_q^{\otimes 2^{\ell^*}}$. The right-hand side $2^{\ell^* - 1} \log(q + 2)$ of (11), which is also the one of [16], and the conventions of [1] correspond to $CW_q^{\otimes 2^{\ell^* - 1}}$: a node of level ℓ corresponds to a constituent tensor of $CW_q^{\otimes 2^{\ell - 1}}$, whose shape sums to 2^ℓ because every factor CW_q contributes indices that sum to 2. For $\ell^* = 4$ this is the eighth power, as stated in [1] and in the abstract of this note.

Remark 3.4 (the conditions (F1) and (F2)). The theorem of [16] is stated for all points and numbers Ω that satisfy (F3). We have added two conditions.

Condition (F1). The argument that leads from (F3) to $\omega \leq \Omega$ compares two exponential rates and then divides by M_{total} (Lemma B.1). It gives nothing when $M_{\text{total}} = 0$; in that case (F3) does not even involve Ω .

Condition (F2). The numbers under the minima are exponents: up to a nonnegative weight, each minimum is the exponential rate of the number of copies of a tensor that one stage of [1] produces in one region. The numbers $x_T^{(r)}$ are nonnegative at every point, because a shape is determined by two of its coordinates and the entropy is subadditive; so are $y_G^{(r)}$ and $z_G^{(r)}$ at the root, because the entropy is concave. At the nodes of level at least 3, however, the numbers $y_T^{(r)}$ can be negative, and so can the numbers $z_T^{(r)}$ when the level is at least 4. There are points at which a minimum in (8) is negative.¹ The statements of [1] carry no condition on the signs.

¹For example, let $\ell^* = 3$, and let all the mass be at the node $T = G[(1, 4, 3), 1]$, that is, $A_G = (1, 0, \dots, 0)$ and $\alpha_G^{(1)} = \delta_{(1, 4, 3)}$. Let $A_T = (1, 0, \dots, 0)$, $\alpha_T^{(1)} = \frac{1}{2}\delta_{(1, 2, 1)} + \frac{1}{2}\delta_{(0, 2, 2)}$, $\mu = \frac{2}{5}$ at the child of shape $(1, 2, 1)$ and $\beta = \delta_{(1, 1)}$ at the child of shape $(0, 2, 2)$. Then $y_T^{(1)} = \frac{6}{5} \log \frac{3}{5} - \frac{1}{5} \log \frac{1}{5} < 0$, so the minimum in (8) for $\ell = 3$ and $r = 1$ is negative. At this point $M_{\text{total}} = \frac{1}{5} \log q$, which is positive for $q \geq 2$, and (F3) holds when Ω is large.

We have not examined whether their proofs cover a region whose weight is positive and whose exponent is negative or zero, and with (F2) we do not need to: in Appendix B the stages are used only when every region of positive weight has a positive exponent.

3.2

The maximum entropy (3) is the only quantity of §2 that is not given by a closed formula. It enters E_{total} only through the numbers $x_T^{(r)}$ in (4), with a negative sign, and these numbers enter (7) and (8) under minima and with nonnegative weights. We record the consequence.

Lemma 3.5 (see [16, §2.5]). *Suppose that every $H_D^{\max}(\alpha)$ in (4) is replaced by a number that is at least as large. Then none of the numbers $x_T^{(r)}$ increases, none of the minima in (7), (8) and (9) increases, and E_{total} does not increase. In particular, if (F2) and (F3) hold after the replacement, then they hold before it.*

Proof. The numbers $x_T^{(r)}$ do not increase by (4). They enter the minima in (7) directly and those in (8) with the weights $m_T A_T^{(r)} \geq 0$, and the minima enter E_{total} with the weights $A_G^{(r)} \geq 0$ and 1. The numbers $y_T^{(r)}$, $z_T^{(r)}$, $e_{T,W}$ and M_{total} do not involve maximum entropies. $\square$

Upper bounds for $H_D^{\max}(\rho)$ come from duality. Let $D' \subseteq D$ be the set of shapes $a \in D$ such that $\rho_W(a_W) > 0$ for $W = X, Y, Z$. Every $\rho' \in \Delta(D)$ with the marginals of ρ vanishes outside D' .

Lemma 3.6 (weak duality). *Let $\rho \in \Delta(D)$, and let $\lambda_X, \lambda_Y, \lambda_Z$ be real-valued functions defined on the values of the three coordinates of the elements of D' . Then, with natural logarithms,*

$$H_D^{\max}(\rho) \leq \ln \left(\sum_{a \in D'} e^{\lambda_X(a_X) + \lambda_Y(a_Y) + \lambda_Z(a_Z)} \right) - \sum_W \sum_w \lambda_W(w) \rho_W(w).$$

Moreover $H_D^{\max}(\rho) \leq \ln |D'|$.

Proof. Put $g(a) = \lambda_X(a_X) + \lambda_Y(a_Y) + \lambda_Z(a_Z)$, $Z = \sum_{a \in D'} e^{g(a)}$ and $Q(a) = e^{g(a)}/Z$ for $a \in D'$. Let $\rho' \in \Delta(D)$ have the marginals of ρ . Then ρ' is supported on D' , and the inequality $\sum_a \rho'(a) \ln(\rho'(a)/Q(a)) \geq 0$ gives $H(\rho') \leq -\sum_a \rho'(a) \ln Q(a) = \ln Z - \sum_a \rho'(a) g(a)$. The last sum equals $\sum_W \sum_w \lambda_W(w) \rho'_W(w) = \sum_W \sum_w \lambda_W(w) \rho_W(w)$. The second bound is the case $\lambda = 0$. $\square$

A second bound is used in [16].

Lemma 3.7 ([16, Lemma 1]). *Let $\rho \in \Delta(D)$, and let $y \in \Delta(D')$ satisfy $y(a) > 0$ for all $a \in D'$ and $y_W = \rho_W$ for $W = X, Y, Z$. Suppose that real numbers λ_0 , $\lambda_W(w)$ and $\varepsilon \geq 0$ satisfy*

$$|\ln y(a) - (\lambda_0 + \lambda_X(a_X) + \lambda_Y(a_Y) + \lambda_Z(a_Z))| \leq \varepsilon \quad \text{for all } a \in D'.$$

Then, with natural logarithms, $H(y) \leq H_D^{\max}(\rho) \leq H(y) + 2\varepsilon$.

This is [16, Lemma 1], which is stated there with logarithms to base 2, applied to the set D' , together with the equality $H_D^{\max}(\rho) = H_{D'}^{\max}(\rho)$. A distribution y as in the lemma need not exist. When none is available, verifier B of §4.3 uses the bound $H_D^{\max}(\rho) \leq \ln |D'|$ of Lemma 3.6 instead; this does not happen at the point of §4.

4 A feasible point at level four

In this section $q = 5$, $\ell^* = 4$, and logarithms are natural.

4.1 The point

The point is stored in the file

CERT_it2312.npz (33.9 MB; SHA-256 1070d16b6d7c84c4700651baedf8cf31
c0d8f80910fd95ac158b6f646209018e),

a NumPy archive, which is part of the archive that accompanies this note (Appendix A). Its arrays with names beginning with P/ hold the variables (V1)–(V3) as double-precision floating-point numbers, one row for each distribution; their names, their shapes and the assignment of rows to nodes are described in Appendix A. There are 2917500 stored numbers. Every finite double-precision number is a rational number with a power of two as denominator, and we read it as that rational number.

Definition 4.1. The point p is defined from the file as follows.

- (i) Each distribution $A_T, \alpha_T^{(r)}, \beta_T$ at a node of level at least 3 or at the root is the corresponding stored row, with one change: the entry at the first position at which the row attains its maximum is replaced by 1 minus the sum of the other entries.
- (ii) Each μ_T is the stored number.
- (iii) At the nodes T of level 2 with a zero shape the file follows the format of the solutions published with [1]. If s_T is a permutation of $(0, 2, 2)$, the file stores a number μ_T , and $\beta_T = \mu_T \delta_{(0,2)} + (1 - 2\mu_T) \delta_{(1,1)} + \mu_T \delta_{(2,0)}$. If s_T is a permutation of $(0, 1, 3)$, then β_T is the uniform distribution on the two elements of its domain. If s_T is a permutation of $(0, 0, 4)$, the domain of β_T has one element.

The definition presupposes that the file contains the arrays described in Appendix A, with the shapes given there, that the stored numbers are finite, that the numbers defined in (i)–(iii) are nonnegative, and that every μ_T lies in $[0, \frac{1}{2}]$. This is the first statement of the following proposition.

Proposition 4.2. *Let p be the point of Definition 4.1.*

(i) *The file defines a point p of the problem of §2 for $q = 5$ and $\ell^* = 4$: it contains the required arrays, its stored numbers are finite, the distributions of Definition 4.1 have nonnegative entries with sum 1, and all μ_T lie in $[0, \frac{1}{2}]$.*

(ii) *The following inequalities hold:*

$$E_{\text{total}}(p) \geq E_{\text{lo}} = \frac{6326568006967359}{2^{50}}, \quad M_{\text{total}}(p) \geq M_{\text{lo}} = \frac{4723693635163093}{2^{50}}. \quad (12)$$

(iii) *Each of the 19 minima in (7), (8) and (9) is positive at p .*

Proof of Theorem 1.1, given Proposition 4.2. Let $\Omega = 592789949057/250000000000$ and $\Lambda = 1095450027772747/2^{49}$. Then $\ln 7 < \Lambda$: indeed $\ln 7 = 1.94591014905531330510\dots$ and $\Lambda = 1.94591014905531345391\dots$ By (12) and $\Omega > 0$,

$$E_{\text{total}}(p) + \Omega M_{\text{total}}(p) - 8 \ln 7 \geq E_{\text{lo}} + \Omega M_{\text{lo}} - 8\Lambda = \frac{56793356553301}{2^{50} \cdot 250000000000} > 0.$$

The equality is an identity between integers: the numerator is $6326568006967359 \cdot 250000000000 + 4723693635163093 \cdot 592789949057 - 2^4 \cdot 1095450027772747 \cdot 250000000000$. This is (F3) in natural logarithms (Remark 2.3). Condition (F1) holds because $M_{\text{total}}(p) \geq M_{\text{lo}} > 0$, and condition (F2) is Proposition 4.2(iii). Hence p and Ω are feasible in the sense of Definition 2.1, and Theorem 3.1 gives $\omega \leq \Omega = 2.371159796228 < 2.37116$. $\square$

4.2 Proof of Proposition 4.2

The proof is a computation. We describe it as it is carried out by the first of our two programs, which we call *verifier A*; the second is described in §4.3.

Step 1: the domain. The program checks that the file contains exactly the arrays $P/$ listed in Appendix A, with the shapes given there and with double-precision entries. It checks that every stored number is finite and nonnegative, that every replaced entry in Definition 4.1(i) is nonnegative, and that $0 \leq \mu_T \leq \frac{1}{2}$ for all stored μ_T . The comparisons are between rational numbers and are done exactly. This proves (i). All nodes turn out to have positive mass.

Step 2: exact evaluation. The masses (1), the complete split distributions of all nodes (§2.3), the weights w , the marginals of every $\alpha_T^{(r)}$ and all the mixtures b_W and $\bar{\beta}_{U,W}$ of §2.4 are rational functions of the variables. The program computes all of them exactly in $\mathbb{Q}$. As a safeguard it checks that the complete split distributions $\beta_{T,W}^{(r)}$ and $\beta_{T,W}$ which it computes at the nodes of level at least 3 with positive shape have sum exactly 1.

Step 3: logarithms. After Step 2, each of the quantities in (4)–(9) and in §2.5, apart from the maximum entropies, is a finite sum of terms $c \cdot p \ln p$ and $c \ln 5$, $c \ln 2$ with rational c and p . The program evaluates these sums in ball arithmetic with a precision of 192 bits [21]: every real number is represented by a midpoint and a radius, and each operation returns a ball that contains the exact result.

Step 4: maximum entropies. There are 207 906 maximum entropies: six at the root and six at each of the 34 650 nodes of level 3 or 4 with positive shape. For each of them the program computes multipliers λ by floating-point iteration and replaces $H_D^{\max}(\alpha)$ by the smaller of the two upper bounds of Lemma 3.6, evaluated in ball arithmetic at these λ and at the exact marginals. The lemma holds for every λ , so no property of the iteration is used. By Lemma 3.5 the replacement can only decrease the 19 minima and $E_{\text{total}}(p)$.

Step 5: the minima. A minimum of numbers with known lower bounds is bounded below by the minimum of the lower bounds. The program so obtains a rational lower bound for each of the 19 minima, and rational numbers $E_{\text{lo}}^A \leq E_{\text{total}}(p)$ and $M_{\text{lo}}^A \leq M_{\text{total}}(p)$, namely the lower end points of the final balls. The last two are dyadic rationals with long numerators; to twenty digits,

$$E_{\text{lo}}^A = 5.61912117459265324278\dots, \quad M_{\text{lo}}^A = 4.19548274813843494366\dots$$

Since $E_{\text{lo}}^A \geq E_{\text{lo}}$ and $M_{\text{lo}}^A \geq M_{\text{lo}}$, this proves (ii). The lower bounds for the minima are as follows: at least 1.813 for each of the six minima in (7), at least 0.161 for each of the twelve minima in (8), and at least 1.169 for E_2 . This proves (iii). $\square$

Table 2 shows the values of the individual terms. Verifier A proves $\omega \leq 2.371159796155502$ by itself; the slightly larger number in Theorem 1.1 is the one for which both programs agree.

4.3 Two programs

A proof by computation is as reliable as its program. We therefore use two programs, which were written separately and compute in different ways.

Verifier A (directory `verifier_A` of the archive) is the program of §4.2. It was written from scratch, from the text of [16, §2] and a description of the file format. It is short, about 980 lines of Python, it works node by node along the tree, and its rigorous arithmetic is exact rational arithmetic and ball arithmetic from the library FLINT [17, 21]; only the multipliers of Step 4 are computed in floating-point arithmetic. It reads the arrays $P/$ of the file and the entries `meta/` (the level, the number q , the number $\kappa = 1$ and the number Ω); it does not read the arrays $H/$.

quantity	verifier A	verifier B
E_G	1.841753136049	[1.841753136049, 1.841753136050]
E_4	1.141654465053	[1.141654465053, 1.141654465054]
E_3	1.466568743885	[1.466568743885, 1.466568743887]
E_2	1.169144829604	[1.169144829565, 1.169144829642]
E_{total}	5.619121174592	[5.619121174553, 5.619121174632]
M_X	4.195482748147	[4.195482748035, 4.195482748259]
M_Y	4.195482748138	[4.195482748026, 4.195482748250]
M_Z	4.195482748405	[4.195482748294, 4.195482748517]
M_{total}	4.195482748138	[4.195482748026, 4.195482748250]

Table 2: The terms of (10) at the point p , in natural logarithms. Verifier A: lower bounds for the retained exponents, in which the maximum entropies have been replaced by upper bounds, and the values of the matrix sizes; all rounded down to twelve decimals. Verifier B: intervals that contain the true values; this program encloses each maximum entropy from both sides. Lower end points are rounded down and upper end points are rounded up. For comparison, $8 \ln 7 = 15.567281192442506 \dots$

Verifier B (directory `verifier_B` of the archive) treats all nodes of the same level and shape at once, as arrays of floating-point intervals. Every arithmetic operation is rounded outwards, sums and products of many terms are enclosed by the standard a priori bounds for floating-point sums and inner products [20, Chapters 3 and 4], and every logarithm is enclosed with MPFR [18] under directed rounding. For the maximum entropies it uses Lemma 3.7: the file contains, for each of the 207 906 problems, a floating-point approximation of the maximiser and of its multipliers; the program corrects the approximate maximiser over $\mathbb{Q}$ so that it is positive and has exactly the required marginals, bounds the residual ε from above, and uses the bound $H(y) + 2\varepsilon$ or $\ln |D'|$, whichever is smaller. The correction is found by exact linear algebra, in the first instance by multiplying the approximate maximiser by a factor $1 + h(a)$ with $h(a) = \eta_X(a_X) + \eta_Y(a_Y) + \eta_Z(a_Z)$ and rational numbers η ; whatever its origin, the corrected distribution is accepted only after its positivity and its marginals have been checked exactly. For the file of §4.1 all 207 906 corrections succeed, and all residuals ε are at most $2.2 \cdot 10^{-14}$. The numbers E_{lo} , M_{lo} and Λ of §4.1 are the bounds computed by verifier B; they are double-precision numbers, which is why they can be printed in full. The rational number Ω of Theorem 1.1 is the smallest number with twelve decimals that satisfies $E_{\text{lo}} + \Omega M_{\text{lo}} \geq 8\Lambda$; it was computed by the companion program `verifier_B/certify.py`, which wrote the file. Verifier B checks the positivity of the minima in the same way as verifier A, with one difference: for the six minima in (7) it bounds the products $A_G^{(r)} \min\{x_G^{(r)}, y_G^{(r)}, z_G^{(r)}\}$ from below.

The two verifiers share the input file and the conventions by which its rows are assigned to the nodes of the tree (Appendix A). These conventions are part of the definition of the point p . Theorem 1.1 does not depend on them: under any assignment of the rows to the nodes that respects their lengths, the file defines a point of the problem, and the theorem needs only one feasible point. The two programs share no code of their own. Both are written in Python and use NumPy and FLINT; verifier A uses FLINT for all of its rigorous arithmetic, and verifier B only for the exact correction of the maximisers. Their results agree: every number in the first column of Table 2 lies in the corresponding interval of the second.

4.4 Further checks

The following checks are not part of the proof. They address the possibility that both programs implement the formulas of §2 incorrectly in the same way.

- (a) *The program of the authors of [1].* The authors of [1] make available a MATLAB program that evaluates the constraints of their optimisation problem [1, §7]; they used it for $\ell^* = 3$. We ran this program, unchanged, on six random points of the problem for $\ell^* = 3$. Each point has 42 candidates: the three numbers under each of the 13 minima, with their weights, and the three matrix sizes. The values of the MATLAB program differ from those of verifier A by less than $2.2 \cdot 10^{-15}$. On the solution published with [1] the two programs differ by $3.3 \cdot 10^{-12}$. This difference comes from the stored solution and not from the formulas: its rows have sum 1 only up to $8 \cdot 10^{-13}$, and the maximising distributions of (3), which the MATLAB program takes from its input, are approximate in it. When both are corrected, the difference is $2.2 \cdot 10^{-15}$. Verifier A has no code that is specific to one level, apart from the leaves of level 2. At level 4 itself we have no comparison with the MATLAB program: one evaluation of the point p by a variant of it without derivatives had not finished after two and a half hours when this note was completed.
- (b) *A third evaluation.* A third program was written from the text of [16, §2] and a description of the layout of the file; the two verifiers were not consulted. It evaluates the quantities of §2 in extended-precision floating-point arithmetic, and again with thirty significant digits. At the point p its values of E_G , E_4 , E_3 , E_2 , M_X , M_Y , M_Z and of the 18 minima in (7) and (8) differ from those of verifier A by less than 10^{-14} , and it finds the value 2.37115979615550116 for p ; the smallest Ω certified by verifier A is 2.37115979615550116 when rounded up to the same number of digits.
- (c) *The published solution for $\ell^* = 3$.* For the solution published with [1], verifier A and verifier B both accept it with $\Omega = 2.371338900719$. This agrees with the published bound 2.371339.
- (d) *A point with known value.* Consider the point for $\ell^* = 4$ whose mass is concentrated on one path of the tree: the root chooses region 1 and the shape (4, 4, 8), that node chooses region 1 and the split (2, 2, 4) + (2, 2, 4), its child chooses region 1 and the split (1, 1, 2) + (1, 1, 2), and $\mu = \frac{1}{4}$ at the resulting node of level 2. The masses along the path are 1, 2 and 4, and a computation by hand gives $E_G = E_4 = E_3 = 0$, $E_2 = 4 \ln 2$ and $M_X = M_Y = M_Z = 2 \ln 5$. The bounds of verifier A for E_G , E_4 , E_3 , E_{total} and M_{total} differ from 0, 0, 0, $4 \ln 2$ and $2 \ln 5$ by less than 10^{-50} . The lower bounds of verifier B for E_{total} and M_{total} lie below these two numbers, by less than 10^{-9} . Several minima vanish at this point, so it does not satisfy (F2), and both verifiers reject it, as they should.
- (e) *Rejections.* Both verifiers reject the file of §4.1 when the number Ω stored in it is lowered below the smallest number that verifier A can certify (Appendix A). On a certificate for $\ell^* = 3$ we made changes that violate the first statement of Proposition 4.2: a negative entry; a row in which the entries other than the largest have sum greater than 1; a value $\mu_T > \frac{1}{2}$; an entry that is not a number; a missing row; a missing array; an array with an additional row or column. Both verifiers reject every such file. They also reject the point of the footnote to Remark 3.4, taken with $q = 5$ and $\Omega = 100$. At that point (F1) and (F3) hold, and the minimum in (8) for $\ell = 3$ and $r = 1$ is $\frac{6}{5} \ln \frac{3}{5} - \frac{1}{5} \ln \frac{1}{5} = -0.2911031660\dots$; verifier A reproduces this number.

4.5 What is verified by computer and what is not

Verified by computer, twice: the three statements of Proposition 4.2, for the quantities defined in §2. Verified by hand: the last step, which is the proof of Theorem 1.1 given in §4.1. Not verified by computer: Theorem 3.1, that is, the statements listed in Remark 3.2 and the derivation in Appendix B. As in every proof by computation, we rely on the correctness of the programs, of Python and the libraries NumPy, FLINT, GMP and MPFR, and of the machine.

5 How the point was found

The contents of this section are not used in the proof of Theorem 1.1.

5.1

As in [16, §3.1], each distribution of (V1) and (V2) is parametrised as the softmax of a vector of real parameters, and the maximum entropies are computed by an inner solver instead of being treated as variables. Each μ_T is one half of a logistic function of a real parameter. For the nodes of level 2 with a zero shape we use the restricted family of Definition 4.1(iii). This gives 2 917 500 real parameters. The value of a point is computed by a program written with JAX [6]; our inner solver works on the dual problem. By Lemma 3.6 an inexact dual solution can only increase the computed value.

5.2

The starting point. The published solution of [1] for $\ell^* = 3$ gives the bound 2.371339. We lift it to a point for $\ell^* = 4$. Let A and $\alpha^{(r)}$ be the region weights and the shape distributions at the root of the level-3 solution. The root of the lifted point has $A_G = A$ and the convolutions $\alpha_G^{(r)}(s) = \sum_{u+v=s} \alpha^{(r)}(u)\alpha^{(r)}(v)$. A node T of level 4 with positive shape s has $A_T = A$ and the conditional distributions $\alpha_T^{(r)}(u) = \alpha^{(r)}(u)\alpha^{(r)}(s-u)/\alpha_G^{(r)}(s)$, and its child $T[u, r]$ receives the variables of the child with shape u and region r of the level-3 root, together with the whole subtree below it. A node of level 4 with a zero shape receives the corresponding mixture of products of level-3 complete split distributions. The lifted point has value 2.371909 in floating-point evaluation. It is worse than the solution it comes from and serves only as a start.

5.3

Smoothing. The quantity E_{total} contains 19 minima of three functions each: one for every region in E_G , E_4 and E_3 , and one in E_2 . A twentieth minimum is M_{total} . A natural first approach replaces each minimum by a smooth lower bound, $-\tau \ln \sum_i e^{-c_i/\tau}$ with a small $\tau > 0$, and minimises the resulting upper bound of the value by a gradient method. With Adam [22] and then L-BFGS [25, 24] this brought the value to 2.371214 in about four days, at the end of which it was decreasing by between 10^{-7} and $3 \cdot 10^{-7}$ per hour. At the points reached, the three numbers under each of the twenty minima, taken with their weights, are almost equal. We attribute the slow progress to this: near such a point the smoothed minima have second derivatives of order $1/\tau$ in the directions that separate these numbers.

5.4

Minimax sequential quadratic programming. We therefore keep the minima. Let $c(x) \in \mathbb{R}^{60}$ be the vector of the 60 *candidates* at the parameter vector x : the three numbers under each of the 19 minima, those of (7) multiplied by $A_G^{(r)}$, and the three matrix sizes M_X, M_Y, M_Z . Thus $E_{\text{total}} = \sum_{j=1}^{19} \min_{i \in I_j} c_i$ and $M_{\text{total}} = \min_{i \in I_{20}} c_i$, where $I_1, \dots, I_{20}$ are the twenty groups of three candidates and I_{20} is the group of the matrix sizes. In this section the *value* of x means the number $(8 \ln 7 - E_{\text{total}})/M_{\text{total}}$; the search does not impose the condition (F2), which is checked afterwards by the verifiers. Let Ω_0 be the value of the current point x_0 . By Dinkelbach’s transformation of fractional programmes [14], a point x with $M_{\text{total}}(x) > 0$ has value less than Ω_0 if and only if

$$\varphi(x) = \sum_{j=1}^{19} \min_{i \in I_j} c_i(x) + \Omega_0 \min_{i \in I_{20}} c_i(x) - 8 \ln 7 > 0.$$

Note that $\varphi(x_0) = 0$. We linearise the candidates at x_0 , with Jacobian J , and choose the step d that maximises

$$\sum_{j=1}^{19} \min_{i \in I_j} (c_i + J_i d) + \Omega_0 \min_{i \in I_{20}} (c_i + J_i d) - \frac{1}{2t} d^\top B d,$$

where B is a positive definite matrix and $t > 0$ a step length. The dual of this problem is a convex quadratic programme in 60 variables a , one multiplier for each candidate, over a product of twenty simplices (nineteen standard ones and one scaled by Ω_0); its solution gives the step $d = tB^{-1}J^\top a$. The matrix B^{-1} is a limited-memory quasi-Newton approximation built from gradients of the function $a^\top c$, and the quadratic programme is solved with Clarabel [19]. The step length t is chosen by backtracking on the true value, and the quadratic programme is solved again for each trial t . The Jacobian costs 60 reverse-mode derivative evaluations.

One further device helped. A quasi-Newton approximation built only from the steps actually taken turned out to be a poor model of the curvature along the next step. In each iteration we therefore add three fresh secant pairs to it, obtained from finite differences of gradients along the current step and along two correction directions, and solve the quadratic programme again before the choice of the step length.

The search described in [8], which was carried out independently of ours, rests on similar ideas: it also works with the unsmoothed value, uses the transformation of [14], and builds second-order models on a small number of nested directions.

5.5

The runs. All runs were made on one machine with a 64-core AMD EPYC 7713 processor. Adam on the smoothed value took the lifted point from 2.371909 to 2.371266, and L-BFGS on the smoothed value reached 2.371214 after about four days. The minimax method without fresh secant pairs, started from that point, reached 2.371183 after 467 iterations of about two minutes each. The minimax method with fresh secant pairs was started from that iterate. It used eight cores on average, and it passed the bound 2.371177 of [16] after 3.4 hours. The point of Theorem 1.1 is the iterate number 2312 in the consecutive numbering of the two minimax runs; it was reached 90 hours after the start of the second one, on 5 October 2026. At that time the value was decreasing by about 10^{-9} per iteration. The programs of the search use JAX 0.11.2 and CVXPY 1.9.3 with Clarabel 0.11.1. Table 3 lists some of the points along the way that were certified in the same manner as p .

point	level	certified Ω
published solution of [1]	3	2.371338900719
minimax method from that solution	3	2.371275696559
minimax method, iterate 520	4	2.371176844590
minimax method, iterate 806	4	2.371163870436
minimax method, iterate 2312	4	2.371159796228

Table 3: Some certified points. In every row, verifier A and verifier B accept the point together with the number Ω of the last column. Iterates are numbered consecutively through the two minimax runs. The files are listed in Appendix A. The last row is the point of Theorem 1.1.

6 Remarks

6.1

The level-3 problem. The same method, applied to the problem for $\ell^* = 3$ and started from the published solution of [1], gives a point that is certified with $\Omega = 2.371275696559$ (Table 3), whereas the published solution gives 2.371339. That the published level-3 solution is not optimal was observed before: Townsend [34] reports the value 2.371301, checked in floating-point arithmetic, and BSD [7] the bound 2.371281376..., checked in Lean with the numerical checks evaluated by compiled code and the theorem of [1] taken as an axiom. All these level-3 values are weaker than the bound of [16].

6.2

Optimality. We do not know the optimal value of the problem for $\ell^* = 4$. Along our last run the value kept decreasing, at a rate that fell from about 10^{-7} per iteration at the start to about 10^{-9} per iteration at the iterate used here. The bound of this note is smaller than that of [16] by about $1.72 \cdot 10^{-5}$, and the bound of [16] is smaller than that of [1] by about $1.6 \cdot 10^{-4}$. Better optimisation at a fixed level therefore moved the bound in the fifth decimal place. This suggests that substantially better bounds on ω need a change of the analysis rather than of the optimiser; the same expectation is expressed in [16, §5]. Limitations of approaches based on the tensors CW_q are studied in [3, 10].

6.3

Higher levels. The problem for $\ell^* = 5$ concerns $CW_5^{\otimes 16}$. Its nodes of level 5 with a zero shape carry distributions on sets $\mathcal{C}_{5,a}$ with up to 5 196 627 elements, and the tree is much larger. We experimented with a restricted family of points for $\ell^* = 5$ and did not reach the values obtained at $\ell^* = 4$. We make no claim about it.

Acknowledgements

The author thanks the authors of [1] for publishing their solutions and their MATLAB program, which provided the starting point of the search and a reference for the comparison of §4.4, and the authors of [16] for the closed form of the optimisation problem.

A Files and reproduction

The point, the two verifiers, the outputs of our runs and the material of the checks of §4.4 are in the archive

`omega-2.37116.tar.gz` (34.1 MB)
SHA-256 `c723a4c7df5ffeeca803b2a7254b9d03d5a753cbcbaa6e42a4866e0eb816fc8c`,

which accompanies this note. All paths below are relative to its root directory. Its file `README.md` describes the contents in more detail, and its file `SHA256SUMS` lists the checksums of all its files.

A.1 The file

The point of §4.1 is in `certificate/CERT_it2312.npz`. This file contains 332 arrays with names beginning with `P/`, which define the point:

- `P/AG`, of shape (6): the distribution A_G ; `P/dist`, of shape (6, 153): the distributions $\alpha_G^{(r)}$. Shapes $s \in \mathcal{S}_\ell$ are ordered by s_X and then by s_Y , increasingly.
- For every positive shape s of level $l \in \{3, 4\}$: `P/Ll_sX_sY_sZ/A`, of shape $(n, 6)$, and `.../alpha`, of shape $(n, 6, |\text{Split}(s)|)$, where n is the number of nodes of that level and shape and `Split(s)` is ordered as a subset of $\mathcal{S}_{l-1}$.
- For every zero shape s of level $l \in \{3, 4\}$: `P/Ll_sX_sY_sZ/beta`, of shape $(n, |\mathcal{C}_{l,a}|)$, with a as in (V2) and $\mathcal{C}_{l,a}$ in lexicographic order.
- For level 2: `P/L2_sX_sY_sZ/mu`, of shape (n) , for the three positive shapes and for the three permutations of $(0, 2, 2)$.

Within one array the nodes are ordered as they are created by the following depth-first construction of the tree: at the root, for $r = 1, \dots, 6$ and for $s \in \mathcal{S}_4$ in order, create $G[s, r]$ and then its subtree; at a node T with positive shape s and level at least 3, for $r = 1, \dots, 6$ and for $u \in \text{Split}(s)$ in order, create $T[u, r]$ and its subtree unless it exists already, and then $T[s - u, r]$ and its subtree unless it exists already.

The arrays `meta/L`, `meta/q` and `meta/K` contain the level 4, the number $q = 5$ and the number 1, which stands for square matrix multiplication; `meta/Omega` contains the string `592789949057/250000000000`, and `meta/digits` the number 12 of its decimals. The 381 arrays with names beginning with `H/` contain the approximate maximisers and multipliers used by verifier B.

The file was written by the program `verifier_B/certify.py` from the output of the optimiser, which is the file `snap_it2312.npz` of the supplementary archive of §A.3 (SHA-256 `11e1abf717fb49c445ed74f8a6c2c293c7db766af00e5eba46f3bc6b6b55f1f5`). We ran this program a second time on that output, with the same settings, and obtained the same file byte for byte. The optimiser’s output is not needed for the verification.

A.2 Commands

Our runs used Python 3.13.14, NumPy 2.2.2 and python-flint 0.9.0 with FLINT 3.6.0, which contains Arb. Verifier B needs in addition GCC with OpenMP and the libraries MPFR and GMP (we used GCC 15.3.0, MPFR 4.2.1 and GMP 6.3.0): it has a C helper of 120 lines, which is compiled on first use into the directory `verifier_B/advw`. The running times below were measured on a shared machine. The script `verify.sh` runs the steps 1–3.

1. Integrity.

```
sha256sum certificate/CERT_it2312.npz
```

2. Verifier A, with four processes (about 3 minutes).

```
(cd verifier_A && OMP_NUM_THREADS=1 python3 verify.py \
  ../certificate/CERT_it2312.npz --procs 4 --digits 15 --json OUT.json)
```

The last three lines of the output are as follows, and the exit code is 0.

```
ALL 19 MINIMA POSITIVE: YES
CERTIFIED OMEGA (independent verifier): 2.371159796155502 [check PASS]
CLAIMED OMEGA 592789949057/2500000000000 : PASS
```

The file `OUT.json` contains E_{lo}^A , M_{lo}^A and the upper bound for $8 \ln 7$ used by verifier A, as exact rational numbers. Their digits beyond the fiftieth can differ from one run to the next, because the sums of balls are formed in the order in which the processes finish.

3. Verifier B, with two threads (about 4 minutes).

```
(cd verifier_B && OMP_NUM_THREADS=2 python3 verify_certificate.py \
  ../certificate/CERT_it2312.npz)
```

The last line of the output is as follows, up to the running time in its last field, and the exit code is 0.

```
CERTIFIED OMEGA: 592789949057/2500000000000 = 2.371159796228 [PASS] (220.1s)
```

4. Verifier A first checks that the file contains exactly the arrays `P/` listed in §A.1, with these shapes and with double-precision entries. Verifier B checks that these arrays are present with these shapes; it reads their entries as double-precision numbers and ignores other arrays. Both then check the conditions of Step 1 of §4.2. Each verifier exits with code 0 only if the following two conditions hold: its lower bounds for the 19 minima are positive, and the inequality $E + \Omega M \geq R$ holds for the number Ω stored in the file, for its own lower bounds E of $E_{\text{total}}(p)$ and $M > 0$ of $M_{\text{total}}(p)$, and for its own upper bound R of $8 \ln 7$. Verifier B prints its lower bounds for the minima before its last line. As a test we replaced the stored Ω by `2371159796155/1000000000000`, which is smaller than the smallest number that verifier A can certify; both verifiers then reject the file.
5. Unit tests. They need in addition `pytest`, `mpmath`, and `CVXPY` with `Clarabel`.

```
(cd verifier_A && python3 -m pytest tests)
(cd verifier_B && python3 -m pytest tests)
```

Three of the tests use the solution published with [1], which is not part of the archive; they are skipped unless it is supplied as described in `checks/README.md`.

A.3 Other files

The directory `expected_output` contains the outputs of our runs of the steps 2 and 3. The directory `checks` contains the material of §4.4: a description of the comparison with the MATLAB program of [1], with its tables and our scripts, the third evaluation, and the scripts and logs of the remaining tests. The MATLAB program and the solution files of [1] are not ours to distribute; they are available at the address given in [1, §7].

A supplementary archive,

omega-2.37116-supplement.tar.gz (83.5 MB)
SHA-256 2578df22c38b8638cda5d83cf19fd3c717646190d0e1e990e1944077e2cc3ba5,

contains what is not needed for the proof of Theorem 1.1: the certificates of the second, third and fourth rows of Table 3, the output of the optimiser from which the file of §4.1 was written, and the programs and logs of the search of §5. The certificate of the first row of Table 3 is obtained from the solution file `W1.00_2.371339.mat` published with [1] by the commands given in the file `checks/README.md` of the main archive.

B From the theorems of [1] to Theorem 3.1

In this appendix, numbered statements of [1] refer to version 3 of the arXiv preprint, logarithms are to base 2 as in [1], and $q \geq 1$, $\ell^* \geq 2$ are fixed. We explain why Theorem 3.1 is a consequence of the statements listed in Remark 3.2. Degenerations of tensors are understood in the sense of [1, §3]. As in [1], numbers of the form cn with a real constant c are used as exponents of tensor powers without mention of rounding.

B.1 Interface tensors

For $\ell \geq 1$ the tensor $\text{CW}_q^{\otimes 2^{\ell-1}}$ is the sum of its *level- ℓ constituent tensors* T_s , $s \in \mathcal{S}_\ell$ [1, §3]. Its X -variables are partitioned into *level-1 blocks* indexed by the sequences in $\{0, 1, 2\}^{2^{\ell-1}}$, and the X -variables of T_s lie in the blocks whose sequences have sum s_X ; the same holds for Y and Z . Consequently the level-1 blocks of a power $T_s^{\otimes n'}$ are indexed, in each dimension, by n' -tuples of such sequences. For complete split distributions $\beta_X, \beta_Y, \beta_Z$ and $\varepsilon \geq 0$, the tensor $T_s^{\otimes n'}[\beta_X, \beta_Y, \beta_Z, \varepsilon]$ is the restriction of $T_s^{\otimes n'}$ to the level-1 blocks in which, for each dimension W , the empirical distribution of the n' sequences is within ε of β_W [1, Definition 3.6]. A *level- ℓ ε -interface tensor* with parameter list $\{(n_t, s_t, \beta_{X,t}, \beta_{Y,t}, \beta_{Z,t})\}_t$ is the Kronecker product of the tensors $T_{s_t}^{\otimes n_t}[\beta_{X,t}, \beta_{Y,t}, \beta_{Z,t}, \varepsilon]$ [1, Definition 4.1].

Let p be a point of the problem of §2. For $2 \leq \ell \leq \ell^*$ and an integer n we consider the parameter list

$$\mathcal{P}_\ell = \{(nm_T, s_T, \beta_{T,X}, \beta_{T,Y}, \beta_{T,Z})\}_{T \text{ of level } \ell}$$

with one term for every node of level ℓ ; terms with $nm_T = 0$ are trivial and are omitted. All complete split distributions occurring in it satisfy the normalisations assumed in [1, Remarks 5.2 and 6.1]: $\beta_{T,W}$ is supported on the sequences with sum $s_{T,W}$, and if exactly one coordinate of s_T vanishes, the distributions of the other two dimensions are exchanged by $L \mapsto \vec{2} - L$. Both hold by the construction (a)–(c) of §2.3.

B.2 The statements used

We use three statements of [1] about degenerations, and the asymptotic sum inequality.

- (S1) *Global stage* [1, Theorem 5.3 and Proposition 5.1]. Let $A \in \Delta(\{1, \dots, 6\})$, let $\alpha^{(r)} \in \Delta(\mathcal{S}_{\ell^*})$, and let $\beta_{W,s}^{(r)}$ be level- ℓ^* complete split distributions. For every $\varepsilon > 0$, a direct sum of $2^{o(n)}$ copies of $(\text{CW}_q^{\otimes 2^{\ell^*-1}})^{\otimes n}$ degenerates into $2^{(\sum_r A_r E_r - o_{1/\varepsilon}(1))n - o(n)}$ copies of the level- ℓ^* ε -interface tensor with parameter list $\{(nA_r \alpha^{(r)}(s), s, \beta_{X,s}^{(r)}, \beta_{Y,s}^{(r)}, \beta_{Z,s}^{(r)})\}_{r,s}$. Here E_r is the minimum of three numbers defined in [1, Proposition 5.1 and Table 2], and $o_{1/\varepsilon}(1)$ denotes a function of ε that tends to 0 as $\varepsilon \rightarrow 0$.

(S2) *Constituent stage* [1, Theorem 6.4 and Proposition 6.3]. Let $\ell \geq 2$. Consider a parameter list $\{(n_t, s_t, \beta_{X,t}, \beta_{Y,t}, \beta_{Z,t})\}_t$ of level ℓ in which all shapes s_t are positive. Let $A_t \in \Delta(\{1, \dots, 6\})$, $\alpha_t^{(r)} \in \Delta(\text{Split}(s_t))$, and let $\beta_{W,t,u}^{(r)}$ be level- $(\ell-1)$ complete split distributions for $u \in \text{Split}(s_t)$, such that

$$\beta_{W,t} = \sum_r A_{t,r} \beta_{W,t}^{(r)}, \quad \beta_{W,t}^{(r)} = \sum_{u \in \text{Split}(s_t)} \alpha_t^{(r)}(u) \beta_{W,t,u}^{(r)} \times \beta_{W,t,s_t-u}^{(r)}, \quad (13)$$

and such that the marginal of $\alpha_t^{(r)}$ in each coordinate W is the distribution of the sum of the first half of a sequence with distribution $\beta_{W,t}^{(r)}$. Then, for every $\varepsilon > 0$, a direct sum of $2^{o(n)}$ copies of the level- ℓ 3ε -interface tensor with the given parameter list degenerates into $2^{\sum_r E_r - o_{1/\varepsilon}(1)n - o(n)}$ copies of the level- $(\ell-1)$ ε -interface tensor with parameter list

$$\{(n_t A_{t,r}(\alpha_t^{(r)}(u) + \alpha_t^{(r)}(s_t - u)), u, \beta_{X,t,u}^{(r)}, \beta_{Y,t,u}^{(r)}, \beta_{Z,t,u}^{(r)})\}_{t,r,u}.$$

Here $n = \sum_t n_t$, and E_r is the minimum of three sums over t defined in [1, Proposition 6.3 and Table 3].

(S3) *Terms with a zero coordinate* [1, Theorem 6.2]. Let s be a shape with $s_Z = 0$, and let $\beta_X, \beta_Y, \beta_Z$ satisfy the normalisations of [1, Remark 6.1]. Then $T_s^{\otimes n'}[\beta_X, \beta_Y, \beta_Z, \varepsilon]$ is isomorphic to the matrix multiplication tensor $\langle 1, M, 1 \rangle$ with $\log M = n'(H(\beta_X) + \nu(\beta_X) \log q \pm o_{1/\varepsilon}(1)) \pm o(n)$. The theorem adds that similar results hold when $s_X = 0$ or $s_Y = 0$. The formats are then $\langle 1, 1, M \rangle$ and $\langle M, 1, 1 \rangle$, respectively: the tensor $\langle a, b, c \rangle$ has ab, bc and ca variables in the dimensions X, Y and Z , and a tensor $T_s^{\otimes n'}[\beta_X, \beta_Y, \beta_Z, \varepsilon]$ with $s_W = 0$ has a single variable in the dimension W . In each case $\log M$ is given by the formula above, with the complete split distribution of either of the two other dimensions in place of β_X .

In (S1) and (S2) we call the set of terms with index r the *region* r . Its *weight* is A_r in (S1) and $\sum_t n_t A_{t,r}$ in (S2), and its *exponent* is $n A_r E_r$ in (S1) and E_r in (S2); thus the number of copies produced by a stage is 2 to the sum of the exponents of its regions, up to the error terms. A region of weight 0 contains no tensor, and its exponent is 0. *We use (S1) and (S2) only when every region of positive weight has a positive exponent.*

The statements are combined as in [1, Algorithm 1]. Fix $\varepsilon > 0$. Apply (S1) with $3^{\ell^*} \varepsilon$ in place of ε . Then, for $\ell = \ell^*, \dots, 2$, the level- ℓ interface tensor at hand is a $3^\ell \varepsilon$ -interface tensor; it is the Kronecker product of its terms with a zero shape, which are matrix multiplication tensors by (S3), and of the level- ℓ interface tensor formed by its other terms, to which (S2) is applied with $3^{\ell-1} \varepsilon$ in place of ε . Finally (S3) is applied to all terms of the level-1 interface tensor. Direct sums and Kronecker products of degenerations are degenerations, and so are compositions. When the number of copies at hand grows exponentially with n , grouping the copies into batches of the size $2^{o(n)}$ required by (S2) costs a factor $2^{o(n)}$. The result is a degeneration of a direct sum of $2^{o(n)}$ copies of $\text{CW}_q^{\otimes 2^{\ell^*-1}n}$ into a direct sum of V copies of a matrix multiplication tensor $\langle A, B, C \rangle$, where V is, up to a factor $2^{o(n)}$, the product of the numbers of copies of all stages, and $\langle A, B, C \rangle$ is the Kronecker product of all the matrix multiplication tensors obtained from (S3).

The last step of [1, Algorithm 1] is an application of the asymptotic sum inequality. We state it in the form in which we use it.

Lemma B.1. *Let $N \geq 1$ be an integer, and let E and M be real numbers with $M > 0$. Suppose that there is a function c with $c(\varepsilon) \rightarrow 0$ as $\varepsilon \rightarrow 0$ such that, for every sufficiently small $\varepsilon > 0$ and every sufficiently large n , a direct sum of $2^{o(n)}$ copies of $\text{CW}_q^{\otimes Nn}$ degenerates into a direct sum of $V \geq 1$ copies of a matrix multiplication tensor $\langle A, B, C \rangle$ with*

$$\log V \geq (E - c(\varepsilon))n - o(n), \quad \log \min\{A, B, C\} \geq (M - c(\varepsilon))n - o(n).$$

Here V , A , B , C and the terms $o(n)$ may depend on ε , and logarithms are to base 2. If ω' is a real number with $E + \omega'M \geq N \log(q + 2)$, then $\omega \leq \omega'$.

Proof. Fix ε and n , and put $m = \min\{A, B, C\}$. The tensor $\langle m, m, m \rangle$ is a restriction of $\langle A, B, C \rangle$. The asymptotic rank $\tilde{R}$ does not increase under degeneration. It satisfies $\tilde{R}(S^{\otimes k}) = \tilde{R}(S)^k$, and the asymptotic rank of a direct sum of K copies of a tensor S is at most $K \tilde{R}(S)$, because the k -th Kronecker power of this direct sum is a direct sum of K^k copies of $S^{\otimes k}$. Hence the asymptotic sum inequality [1, Theorem 3.2], applied to V copies of $\langle m, m, m \rangle$, and the bound $\tilde{R}(\text{CW}_q) \leq q + 2$ give

$$V m^\omega \leq \tilde{R}\left(\bigoplus_{i=1}^V \langle m, m, m \rangle\right) \leq 2^{o(n)} (q + 2)^{Nn}.$$

We take logarithms, use $\omega \geq 0$ and the two lower bounds, and divide by n :

$$E - c(\varepsilon) + \omega(M - c(\varepsilon)) - o(1) \leq N \log(q + 2) + o(1) \leq E + \omega'M + o(1).$$

Letting $n \rightarrow \infty$ and then $\varepsilon \rightarrow 0$ we obtain $(\omega - \omega')M \leq 0$, and $M > 0$ gives $\omega \leq \omega'$. $\square$

B.3 The stages for a point of the problem

Let p be a point. We check that p provides parameters for every stage, that the interface tensor passed from each stage to the next has parameter list $\mathcal{P}_\ell$, and we compute the numbers of copies and the sizes.

Global stage. In (S1) take $A = A_G$, $\alpha^{(r)} = \alpha_G^{(r)}$ and $\beta_{W,s}^{(r)} = \beta_{G[s,r],W}$. By (1) the resulting parameter list is $\mathcal{P}_{\ell^*}$. The three numbers whose minimum is E_r are, in the notation of [1, Table 2] and for $r = 1$,

$$H(\alpha_X) - P_\alpha, \quad H(\bar{\beta}_{Y,*,*,*}) - \eta_Y, \quad H(\bar{\beta}_{Z,*,*,*}) - \lambda_Z.$$

Here $P_\alpha = H_{\mathcal{S}_{\ell^*}}^{\max}(\alpha) - H(\alpha)$, the distribution $\bar{\beta}_{W,*,*,*} = \sum_s \alpha(s) \beta_{W,s}$ is our b_W , and

$$\eta_Y = \sum_{s: s_Z=0} \alpha(s) H(\beta_{Y,s}) + \sum_j \alpha(*, j, +) H(\bar{\beta}_{Y,*,j,+}),$$

where $\alpha(*, j, +) = w(U_j)$ and $\bar{\beta}_{Y,*,j,+} = \bar{\beta}_{U_j,Y}$ in the notation of §2.4; similarly for λ_Z and the sets U'_k . Hence the three numbers are $x_G^{(1)}, y_G^{(1)}, z_G^{(1)}$. For the other regions the dimensions are permuted by π_r in [1] exactly as in §2.4. So the exponent of the region r is $n A_G^{(r)} \min\{x_G^{(r)}, y_G^{(r)}, z_G^{(r)}\}$, we have $\sum_r A_r E_r = E_G$, and the number of copies produced by the global stage is $2^{(E_G - o_{1/\varepsilon}(1))n - o(n)}$.

Constituent stage at a level $\ell \geq 3$. The terms of $\mathcal{P}_\ell$ with positive shape are indexed by the nodes T of level ℓ with positive shape. In (S2) take $A_t = A_T$, $\alpha_t^{(r)} = \alpha_T^{(r)}$ and $\beta_{W,t,u}^{(r)} = \beta_{T[u,r],W}$. The two equations (13) are the definition (2). The condition on the marginals holds because $\beta_{T[u,r],W}$ is supported on sequences with sum u_W : the sum of the first half of a sequence with distribution $\beta_{T,W}^{(r)}$ equals l with probability $\sum_{u: u_W=l} \alpha_T^{(r)}(u)$. By (1) the resulting parameter list is $\mathcal{P}_{\ell-1}$. The three sums whose minimum is E_r have, for $r = 1$, the summands

$$n_t A_{t,1} (H(\gamma_{X,t}) - P_{\alpha,t}), \quad n_t A_{t,1} (H(\beta_{Y,t}^{(1)}) - \eta_{Y,t}), \quad n_t A_{t,1} (H(\beta_{Z,t}^{(1)}) - \lambda_{Z,t}).$$

Here $\gamma_{X,t}$ is the distribution of the pair formed by the sums of the two halves of a sequence with distribution $\beta_{X,t}^{(1)}$; the two sums add up to $s_{t,X}$, so by the condition on the marginals $H(\gamma_{X,t})$ is the entropy of the marginal of α_t in the coordinate X . The penalty is $P_{\alpha,t} = \max_{\alpha'} H(\alpha') - H(\alpha_t)$, where α' runs over the distributions on the level- $(\ell - 1)$ shapes that have the marginals of α_t , as in the proof of [1, Proposition 6.3]; such distributions are supported on $\text{Split}(s_t)$, so $P_{\alpha,t} = H_{\text{Split}(s_t)}^{\max}(\alpha_t) - H(\alpha_t)$. Finally $\eta_{Y,t}$ and $\lambda_{Z,t}$ are defined in [1, Table 3] by the formulas for η_Y , λ_Z above with the weights $\alpha_t(u) + \alpha_t(s_t - u)$ in place of $\alpha(s)$. With $n_t = n m_T$ these summands are $n m_T A_T^{(1)}$ times $x_T^{(1)}$, $y_T^{(1)}$, $z_T^{(1)}$; see Remark 2.2 for the normalising weights. Hence the exponent of the region r is n times the r -th minimum in (8), we have $\sum_r E_r = n E_\ell$, and the number of copies produced is $2^{(E_\ell - o_{1/\varepsilon}(1))n - o(n)}$. The error terms of (S2) refer to $\sum_t n_t$, which is at most $2^{\ell^* - \ell} n$: by (1) the masses of the nodes of level ℓ sum to at most $2^{\ell^* - \ell}$.

Constituent stage at level 2. Let T be a node of level 2 with positive shape s , let W be the dimension with $s_W = 2$, and let W', W'' be the other two. Write e_V for the triple with a 1 in the coordinate V and zeros elsewhere. Then $\text{Split}(s) = \{2e_W, e_{W'} + e_{W''}, e_W + e_{W'}, e_W + e_{W''}\}$, and the map $u \mapsto s - u$ exchanges the first two and the last two elements. In (S2) with $\ell = 2$ take $A_t = (1, 0, 0, 0, 0, 0)$ and

$$\alpha_t^{(1)}(2e_W) = \alpha_t^{(1)}(e_{W'} + e_{W''}) = \mu_T, \quad \alpha_t^{(1)}(e_W + e_{W'}) = \alpha_t^{(1)}(e_W + e_{W''}) = \frac{1}{2} - \mu_T;$$

the regions $r \geq 2$ have weight 0, and for them we take the same distributions as for $r = 1$. Level-1 complete split distributions are point masses, since the sets $\mathcal{C}_{1,a}$ have one element. Therefore (13) gives $\beta_{W,t}^{(1)} = \mu_T \delta_{(2,0)} + \mu_T \delta_{(0,2)} + (1 - 2\mu_T) \delta_{(1,1)}$ and $\beta_{V,t}^{(1)} = \frac{1}{2} \delta_{(0,1)} + \frac{1}{2} \delta_{(1,0)}$ for $V \neq W$, in agreement with (b) of §2.3; the complete split distributions of the children are point masses, and so are their averages over the sets U_j and U'_k , whose members share the coordinate in question, so $\eta_{Y,t} = \lambda_{Z,t} = 0$; and $\alpha_t^{(1)}$ is the only distribution on $\text{Split}(s)$ with its marginals, so $P_{\alpha,t} = 0$. The three summands above are therefore $n m_T e_{T,X}$, $n m_T e_{T,Y}$, $n m_T e_{T,Z}$ with $e_{T,V}$ as in (9): the marginal of $\alpha_t^{(1)}$ in the coordinate X and the distributions $\beta_{Y,t}^{(1)}$ and $\beta_{Z,t}^{(1)}$ have entropy $H(\mu_T, \mu_T, 1 - 2\mu_T)$ if the dimension in question is W , and entropy $\log 2$ otherwise. Hence the exponent of the region 1 is $n E_2$, and $\sum_r E_r = n E_2$. The level-1 interface tensor produced has, for each T , four terms: the shapes $2e_W$ and $e_{W'} + e_{W''}$ with exponent $2\mu_T m_T n$, and the shapes $e_W + e_{W'}$ and $e_W + e_{W''}$ with exponent $(1 - 2\mu_T) m_T n$.

Matrix sizes. By (S3), a term of $\mathcal{P}_\ell$ whose node T has a zero shape is a matrix multiplication tensor with two of its three sizes equal to 1; the normalisations required in (S3) hold by §B.1. If exactly one coordinate W_0 of s_T vanishes, the third size M satisfies $\log M = n m_T (H(\beta_T) + \nu(\beta_T) \log q)$ up to the error terms of (S3), because H and ν take the same values on β_T and on its image under $L \mapsto \vec{2} - L$; this is $n M_{T,W_0}$. If two coordinates vanish, the tensor has one variable in each dimension, and $M_{T,W_0} = 0$. The position of M in $\langle \cdot, \cdot, \cdot \rangle$ is the first if $W_0 = Y$, the second if $W_0 = Z$ and the third if $W_0 = X$. For the level-1 terms coming from a node T of level 2 with positive shape, (S3) gives: the term of shape $2e_W$ has size 1; the term of shape $e_{W'} + e_{W''}$, whose zero coordinate is W , has $\log M = 2\mu_T m_T n \log q$; and the terms of shapes $e_W + e_{W'}$ and $e_W + e_{W''}$, whose zero coordinates are W'' and W' , have $\log M = (1 - 2\mu_T) m_T n \log q$. These are n times the local matrix sizes $M_{T,W}$, $M_{T,W''}$, $M_{T,W'}$ of §2.5. Taking the Kronecker product over all leaves, we get

$$\log A = n M_Y, \quad \log B = n M_Z, \quad \log C = n M_X$$

up to terms $o_{1/\varepsilon}(1) n + o(n)$.

Conclusion. Let p and Ω be feasible in the sense of Definition 2.1. By (F2), every region of positive weight has a positive exponent at every stage: at the global stage because the six minima

in (7) are positive, at the levels $\ell \geq 3$ because the minima in (8) are positive, and at level 2, where only the region 1 has positive weight, because $E_2 > 0$. So (S1) and (S2) are used as announced. Moreover $E_G > 0$, so that for ε small enough the numbers of copies grow exponentially with n from the first stage on. Multiplying the numbers of copies of all stages, and taking the minimum of the three sizes, we obtain a function c with $c(\varepsilon) \rightarrow 0$ as $\varepsilon \rightarrow 0$ such that

$$\log V \geq (E_{\text{total}} - c(\varepsilon))n - o(n), \quad \log \min\{A, B, C\} \geq (M_{\text{total}} - c(\varepsilon))n - o(n).$$

By (F1) we have $M_{\text{total}} > 0$, and by (F3) we have $E_{\text{total}} + \Omega M_{\text{total}} \geq 2^{\ell^* - 1} \log(q + 2)$. Lemma B.1 with $N = 2^{\ell^* - 1}$, $E = E_{\text{total}}$, $M = M_{\text{total}}$ and $\omega' = \Omega$ gives $\omega \leq \Omega$. This proves Theorem 3.1, given (S1)–(S3), the asymptotic sum inequality and the bound $\tilde{R}(\text{CW}_q) \leq q + 2$.

References

- [1] J. Alman, R. Duan, V. Vassilevska Williams, Y. Xu, Z. Xu, R. Zhou. More asymmetry yields faster matrix multiplication. In *Proceedings of the 2025 Annual ACM–SIAM Symposium on Discrete Algorithms (SODA)*, pp. 2005–2039, 2025. arXiv:2404.16349. Statements are cited by their numbers in version 3 of the arXiv preprint (19 August 2026).
- [2] J. Alman, V. Vassilevska Williams. A refined laser method and faster matrix multiplication. *TheoretCS* 3, Article 21, 2024. Conference version in *Proceedings of the 2021 ACM–SIAM Symposium on Discrete Algorithms (SODA)*, pp. 522–539, 2021. arXiv:2010.05846.
- [3] A. Ambainis, Y. Filmus, F. Le Gall. Fast matrix multiplication: limitations of the Coppersmith–Winograd method. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 585–593, 2015. arXiv:1411.5414.
- [4] D. Bini, M. Capovani, F. Romani, G. Lotti. $O(n^{2.7799})$ complexity for $n \times n$ approximate matrix multiplication. *Inform. Process. Lett.* 8(5):234–235, 1979.
- [5] M. Bläser. Fast matrix multiplication. *Theory of Computing Library, Graduate Surveys* 5:1–60, 2013.
- [6] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, Y. Katariya, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, Q. Zhang. JAX: composable transformations of Python+NumPy programs. Software, 2018, <https://github.com/google/jax>.
- [7] BSD. Improving a matrix multiplication bound with a MacBook and a proof assistant. Manuscript dated 25 August 2026, Zenodo, doi:10.5281/zenodo.22101463.
- [8] BSD. Improving the matrix multiplication exponent with a MacBook and a proof assistant: a CN-certified level-four bound, conditional on the combination-loss theorem. Manuscript dated 4 October 2026, in the repository <https://github.com/bsd-developer/matrix-math-publish> (commit 6752e13 of 4 October 2026; consulted on 5 October 2026).
- [9] P. Bürgisser, M. Clausen, M. A. Shokrollahi. *Algebraic Complexity Theory*. Grundlehren der mathematischen Wissenschaften 315, Springer, Berlin, 1997.
- [10] M. Christandl, P. Vrana, J. Zuiddam. Barriers for fast matrix multiplication from irreversibility. *Theory Comput.* 17, Article 2, 2021.

- [11] D. Coppersmith, S. Winograd. On the asymptotic complexity of matrix multiplication. *SIAM J. Comput.* 11(3):472–492, 1982.
- [12] D. Coppersmith, S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symbolic Comput.* 9(3):251–280, 1990. Conference version in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 1–6, 1987.
- [13] A. M. Davie, A. J. Stothers. Improved bound for complexity of matrix multiplication. *Proc. Roy. Soc. Edinburgh Sect. A* 143(2):351–369, 2013.
- [14] W. Dinkelbach. On nonlinear fractional programming. *Management Sci.* 13(7):492–498, 1967.
- [15] R. Duan, H. Wu, R. Zhou. Faster matrix multiplication via asymmetric hashing. In *Proceedings of the 64th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 2129–2138, 2023. arXiv:2210.10173.
- [16] E. Dupont, M. Eisenberger, B. Kozlovskii, A. Mehrabian, F. J. R. Ruiz, A. See, R. Zhou, J. Alman, V. Vassilevska Williams, M. Balog. Improving the matrix multiplication exponent with modern optimization and AlphaEvolve. arXiv:2608.16884, version 1, 17 August 2026.
- [17] The FLINT team. FLINT: Fast Library for Number Theory, version 3.6.0, 2026. <https://flintlib.org>
- [18] L. Fousse, G. Hanrot, V. Lefèvre, P. Pélicier, P. Zimmermann. MPFR: a multiple-precision binary floating-point library with correct rounding. *ACM Trans. Math. Software* 33(2), Article 13, 2007.
- [19] P. J. Goulart, Y. Chen. Clarabel: an interior-point solver for conic programs with quadratic objectives. *Math. Program. Comput.*, published online 2026. arXiv:2405.12762.
- [20] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*, second edition. SIAM, Philadelphia, 2002.
- [21] F. Johansson. Arb: efficient arbitrary-precision midpoint-radius interval arithmetic. *IEEE Trans. Comput.* 66(8):1281–1292, 2017.
- [22] D. P. Kingma, J. Ba. Adam: a method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*, San Diego, 2015. arXiv:1412.6980.
- [23] F. Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pp. 296–303, 2014. arXiv:1401.7714.
- [24] D. C. Liu, J. Nocedal. On the limited memory BFGS method for large scale optimization. *Math. Programming* 45:503–528, 1989.
- [25] J. Nocedal. Updating quasi-Newton matrices with limited storage. *Math. Comp.* 35(151):773–782, 1980.
- [26] A. Novikov, N. Vű, M. Eisenberger, E. Dupont, P.-S. Huang, A. Z. Wagner, S. Shirobokov, B. Kozlovskii, F. J. R. Ruiz, A. Mehrabian, M. P. Kumar, A. See, S. Chaudhuri, G. Holland, A. Davies, S. Nowozin, P. Kohli, M. Balog. AlphaEvolve: a coding agent for scientific and algorithmic discovery. arXiv:2506.13131, 2025.

- [27] V. Ya. Pan. Strassen’s algorithm is not optimal: trilinear technique of aggregating, uniting and canceling for constructing fast algorithms for matrix operations. In *19th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 166–176, 1978.
- [28] F. Romani. Some properties of disjoint sums of tensors related to matrix multiplication. *SIAM J. Comput.* 11(2):263–267, 1982.
- [29] A. Schönhage. Partial and total matrix multiplication. *SIAM J. Comput.* 10(3):434–455, 1981.
- [30] A. J. Stothers. *On the complexity of matrix multiplication*. PhD thesis, University of Edinburgh, 2010. <http://hdl.handle.net/1842/4734>
- [31] V. Strassen. Gaussian elimination is not optimal. *Numer. Math.* 13(4):354–356, 1969.
- [32] V. Strassen. The asymptotic spectrum of tensors and the exponent of matrix multiplication. In *27th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 49–54, 1986.
- [33] V. Strassen. Relative bilinear complexity and matrix multiplication. *J. Reine Angew. Math.* 375/376:406–443, 1987.
- [34] A. Townsend. A sharper numerical certificate for the matrix multiplication exponent. Manuscript dated 13 August 2026 (consulted on 5 October 2026). <https://github.com/ajt60gaibb/MatrixMultiplicationBetterExponent>
- [35] V. Vassilevska Williams. Multiplying matrices faster than Coppersmith–Winograd. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 887–898, 2012.
- [36] V. Vassilevska Williams. Multiplying matrices in $O(n^{2.373})$ time. Full version of [35], dated 1 July 2014, <https://people.csail.mit.edu/virgi/matrixmult-f.pdf>
- [37] V. Vassilevska Williams, Y. Xu, Z. Xu, R. Zhou. New bounds for matrix multiplication: from alpha to omega. In *Proceedings of the 2024 Annual ACM–SIAM Symposium on Discrete Algorithms (SODA)*, pp. 3792–3835, 2024. arXiv:2307.07970. Statements are cited by their numbers in the arXiv preprint.
- [38] J. Zuiddam. A Lean 4 proof that $\omega < 2.371339$. Software, 2026, <https://github.com/jzuiddam/matrix-mult-exponent> (consulted on 5 October 2026).