Qutrit witness from the Grothendieck constant of order four

Péter Diviánszky, Erika Bene, and Tamás Vértesi
Institute for Nuclear Research, Hungarian Academy of Sciences, P.O. Box 51, H-4001 Debrecen, Hungary
(Received 24 April 2017; published 13 July 2017)

In this paper, we prove that $K_G(3) < K_G(4)$, where $K_G(d)$ denotes the Grothendieck constant of order d. To this end, we use a branch-and-bound algorithm commonly used in the solution of NP-hard problems. It has recently been proven that $K_G(3) \leqslant 1.4644$. Here we prove that $K_G(4) \geqslant 1.4841$, which has implications for device-independent witnessing dimensions greater than two. Furthermore, the algorithm with some modifications may find applications in various black-box quantum information tasks with large number of inputs and outputs.

DOI: 10.1103/PhysRevA.96.012113

I. INTRODUCTION

The Grothendieck constant K_G [1] is an enigmatic constant arising in Banach space theory [2] with several recent applications in communication complexity [3] and algorithms [4,5]. It is known to be in the range $1.6769 < K_G < 1.7823$, however, its exact value is still unknown. The lower bound above has been given by Davie and Reeds [6], and the upper bound is due to Krivine [7]. There is a refined version of the Grothendieck constant, the Grothendieck constant of order d, denoted by $K_G(d)$. The definition of $K_G(d)$ for any finite $d \geqslant 2$ is given below. Note that the original constant K_G is recovered for $d \to \infty$.

Let us first define $L(M)$ by the optimization problem,

$$L(M) = \max_{a_i=\pm1, b_j=\pm1} \sum_{i=1}^{n}\sum_{j=1}^{n} M_{ij} a_i b_j, \tag{1}$$

over all possible signs of a_i, b_j, $i,j = 1,\ldots,n$, where $M = (M_{ij})$ is an arbitrary $n \times n$ real-valued matrix. The optimization problem (1) is called $K_{n,n}$-quadratic programming in the computer science literature [8]. This is known to be an NP-hard problem in the parameter n [9]. The Grothendieck inequality [1,10] states that

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{n} M_{ij}\vec{a}_i \cdot \vec{b}_j}{L(M)} \leqslant K(d), \tag{2}$$

for all unit vectors $\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n \in \mathbb{R}^d$ and $\vec{b}_1, \vec{b}_2, \ldots, \vec{b}_n \in \mathbb{R}^d$, where $K(d)$ is a universal constant for a fixed d. The smallest value of this constant $K(d)$ such that the inequality still holds is called the Grothendieck constant of order d, which we denote by $K_G(d)$. Recall that $K_G = \lim_{d\to\infty} K_G(d)$.

Despite efforts, the value of the constant $K_G(d)$ is not known in general and its exact value is only known for $d = 2$: $K_G(2) = \sqrt{2}$ [7,11]. For larger d, there appeared better and better lower bounds [6,12–16] and upper bounds [7,17,18] to $K_G(d)$ in the literature.

Our goal is to improve on existing lower bounds for $K_G(d)$ in the case of small dimensions d. Note that according to the definition (2), a lower bound to $K_G(d)$ arises by giving an explicit matrix M and explicit unit vectors $\vec{a}_i$ and $\vec{b}_j$ in dimension d. Denoting by

$$Q(M,d) = \sum_{i=1}^{n}\sum_{j=1}^{n} M_{ij}\vec{a}_i \cdot \vec{b}_j \tag{3}$$

the nominator in the left-hand side of the inequality (2), we get the following lower bound:

$$\frac{Q(M,d)}{L(M)} \leqslant K_G(d). \tag{4}$$

Currently, the best-known lower bound is given by $K_G(4) \geqslant 1.4456$ in Ref. [14]. In this paper, we improve on this bound up to $K_G(4) \geqslant 1.4841$. Since $K_G(3)$ is known to be smaller than 1.4644 [18], the strict relation $K_G(3) < K_G(4)$ follows. As a by-product, we also improve the best lower bound on $K_G(3)$. To this end, we combine the so-called distance algorithm [16,19] with a branch-and-bound algorithm [20]. Let us note that there is a connection between the Grothendieck constant of order d and the nonlocality of XOR games [21]. This link has been established by Tsirelson [22,23] and further expanded in Ref. [24]. Our result $K_G(3) < K_G(4)$ will have implications in this direction as well, entailing a so-called dimension witness for systems beyond qubits [25]. Since both the distance and the branch-and-bound methods have been applied independently in versatile schemes, we believe that together they will find applications in other nonlocality scenarios and large-scale quantum information tasks as well.

The paper is organized as follows. In Sec. II, we introduce the branch-and-bound (BB) algorithm to solve problem (1) and we also present test cases showing its performance for large n matrix dimensions. In Sec. III, the lower bounds are improved both for $K_G(3)$ and $K_G(4)$. In particular, a 92×92 matrix M is constructed in Sec. III A showing that $K_G(4) \geqslant 1.4731$, which is further improved to $K_G(4) \geqslant 1.4841$ in Sec. III B by invoking the distance algorithm. Similarly, it is shown in Sec. III C using a 90×90 matrix that $K_G(3) \geqslant 1.4359$. Note that the best lower bound so far was $K_G(3) \geqslant 1.4261$, presented in Ref. [16]. The connection with nonlocal quantum correlations and the implications for device-independent dimensions witnesses are discussed in Sec. IV. The paper ends with conclusions in Sec. V.

II. THE BRANCH-AND-BOUND (BB) ALGORITHM

A. Description of the algorithm

Let us recall from the introduction that a good lower bound to $K_G(d)$ requires a suitable $n \times n$ matrix M along with a specific arrangement of unit vectors $\vec{a}_i, \vec{b}_j$. Armed with these, we also need a method which is able to efficiently evaluate the

maximum $L(M)$ in formula (1) for large matrix dimensions n. In this section, we propose a solution based on a branch-and-bound technique [20], which is feasible on a standard computer up to $n \sim 90$.

It is known that assuming the Unique Games Conjecture [26], it is NP-hard to approximate the above problem to any factor better than the Grothendieck constant K_G [8]. Actually, if M is the Laplacian matrix of a graph, then the maximum in (1) coincides with the value of the maximum cut of this graph. The maximum cut problem is one of 21 NP-complete problems of Karp [27].

Notice that the optimization problem (1) reduces to the following problem (where the $n \times n$ matrix M is the input to the problem):

$$L(M) = \max_{a_i=\pm 1} \sum_{j=1}^{n} \left| \sum_{i=1}^{n} M_{ij} a_i \right|, \tag{5}$$

where maximization is performed over all possible ± 1 signs of $a_i, i = 1, \ldots, n$. This reformulation of the problem allows us to eliminate variables b_j from the optimization. Note, however, that a brute-force search evaluation of this problem becomes infeasible already for relatively small n, as one has to compute all the 2^{n-1} distinct cases. Such a brute-force technique was used in Refs. [16,28], and the biggest n one could afford (in a reasonable time) on a normal desktop PC was $n = 42$.

In contrast, the BB algorithm is able to cope with generic matrices M with dimensions up to $n \sim 90$ in a reasonable time as it will be discussed next. In our specific problem (5), the BB algorithm performs a systematic enumeration of candidate solutions for $a_i, i = (1, \ldots, n)$ by means of state space search [20]: We can think of our set of candidate solutions as a rooted binary tree with the full set at the root. Let us label a given branch by a particular choice of ± 1 signs of $\{a_1, a_2, \ldots, a_n\}$ variables. Then the algorithm explores branches of this tree representing subsets of the solution set. Before enumerating the candidate solutions of a branch, the branch is checked against estimations of upper bounds on the optimal solution, and the branch is removed if it cannot produce a better solution than the best one found so far by the algorithm.

The estimation of the upper bound is based on the following inequality. Let us fix values of $a_i = \{+1, -1\}, i = 1, \ldots, k$, corresponding to the level k of the branching tree. Then we have the following upper bound:

$$\max_{a_{k+1},\ldots,a_n} \sum_{j=1}^{n} \left| \sum_{i=1}^{n} M_{ij} a_i \right| \leqslant \sum_{j=1}^{n} \left| \sum_{i=1}^{k} M_{ij} a_i \right| + \max_{a_{k+1},\ldots,a_n} \sum_{j=1}^{n} \left| \sum_{i=k+1}^{n} M_{ij} a_i \right|, \tag{6}$$

where maximization is carried out over all possible ± 1 signs of $a_{k+1}, \ldots, a_n$. It is noted that the first term on the right-hand side of Eq. (6) has some fixed value, which for consecutive k's can be computed at low cost by reusing results from previous computations.

An efficient upperbounding is a crucial part of the algorithm, since without discarding branches, the technique traces back to a brute-force search of all possible solutions, which amounts to evaluating 2^{n-1} solutions growing exponentially with n. According to the upper bound (6), the decision about which branches to remove can be taken quickly. We refer the interested reader to the Appendix for a detailed technical description of the algorithm along with simple illustrative examples.

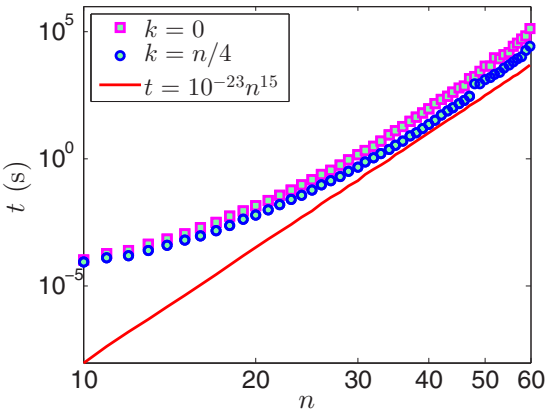

FIG. 1. Time required to compute $L(M)$ as a function of the matrix dimension n of M. The plot has a log-log scale. The purple square and blue circle markers correspond to the curves with parameter $k = 0$ and $k = n/4$, respectively. The red line is for the sake of comparison.

Let us stress that the BB algorithm described above allows us to give an exact value for the problem (1) if all entries of matrix M are integers. This algorithm was implemented in Haskell language; see Ref. [29]. The code performs exact integer arithmetic and includes assembly code in certain crucial parts to boost the computation.

B. Numerical tests

In this section, some benchmark tests are presented. We generated $n \times n$ random M matrices for a given n, where the integer coefficients of the matrix M were chosen within the range $[-100, +100]$ uniformly at random. After averaging over 1000 random matrices for a fixed n, we plot the time required for computing $L(M)$ using the BB algorithm as a function of n in the range $10 \leqslant n \leqslant 60$ (it is noted that for $50 < n \leqslant 60$, the average was taken over only 30 matrices to save computation time). The code was run on a single core of a standard desktop PC. Figure 1 shows the performance of the BB algorithm on a log-log plot. Note that there is a parameter k in the algorithm, which designates the level of the tree above which all nodes are forced to be visited. In this way, we can save computation time since less decisions have to be taken about discarding branches from the tree. To our experience, choosing $k \sim n/4$ gives the best performance. In Fig. 1, we plotted both cases $k = 0$ and $k = n/4$, demonstrating that $k = n/4$ is indeed superior to $k = 0$.

For the sake of comparison, we also plotted the line $t = 10^{-23} n^{15}$ (shown in red). As one can observe, the performance of the BB algorithm can be well approximated with a power-law behavior in the range displayed ($n \leqslant 60$). By extrapolating the curve $t = 10^{-23} n^{15}$ up to $n = 90$, we get a running time in the range of month (carried out on a single core). However, for higher n and generic matrices M, we expect an exponentially

growing behavior due to the NP-complete feature of the problem. Indeed, one can easily construct specific matrices M for which there is no saving in the running time compared to the brute-force technique which has exponential scaling with n. Such a matrix may be built up of $(n/2)$ Clauser-Horne-Shimony-Holt expressions [11] distributed between n settings of Alice and Bob (where n is even). In this case, the number of different $a_i = \pm 1$ $(i = 1,2,\ldots,n)$ strategies attaining $L(M)$ in Eq. (5) grows exponentially with n. In such a case, the number of discarded branches is limited and the performance of the algorithm eventually goes back to that of a brute-force search.

On the other hand, let us also stress that the memory complexity of the algorithm is low. It equals the size of the input matrix M of the problem, which is $O(n^2)$.

III. IMPROVING THE LOWER BOUND ON $K_G(4)$ AND $K_G(3)$

Our task is to come up with a good lower bound to $K_G(d)$, which according to (4) amounts to finding a suitable arrangement of unit vectors $\vec{a}_i, \vec{b}_j$ in $\mathbb{R}^d$ for $i = 1,\ldots,n$ and an $(n \times n)$-dimensional matrix M for which the evaluation of the maximum $L(M)$ in Eq. (1) is feasible on a standard desktop. Due to the numerical tests in Sec. II B, it is expected that $L(M)$ can be computed in a reasonable time up to a matrix dimension $n \approx 90$ by running the BB algorithm.

In order to get $\vec{a}_i$, $i = 1,\ldots,n$, we fix icosahedral symmetry of the set of vectors $\vec{a}_i$ and form a set of $2n$ vectors $\vec{A}_{2i-1} = \vec{a}_i$, $\vec{A}_{2i} = -\vec{a}_i$, for $i = 1,\ldots,n$. Then we optimize the $2n$ unit vectors $\vec{A}_i$ in the d-dimensional Euclidean space assuming icosahedral symmetry such that the optimized configuration corresponds to a (local) minimum of the energy term,

$$E = \sum_{1 \leqslant i < j \leqslant 2n} \frac{1}{\|\vec{A}_i - \vec{A}_j\|}. \tag{7}$$

The goal of this optimization is to find an arrangement of vectors $\vec{A}_i$ on the $(d-1)$ sphere, which distributes the sphere in a relatively even manner. Minimization has been performed using a heuristic search, the so-called Amoeba method [30]. Given the fixed arrangement of vectors $\vec{a}_i$ on Alice's side coming from the above numerical search, due to symmetry reasons we pick the same vectors on Bob's side. That is, we choose $\vec{b}_i = \vec{a}_i$ for all $i = 1,\ldots,n$. This gives an $n \times n$ correlation matrix C defined by the entries $C_{i,j} = \vec{a}_i \cdot \vec{b}_j$. Note that all diagonal entries of this matrix are 1. In the next sections, we present two different methods to obtain the matrix M given the matrix C, considerably improving the lower bound values of $K_G(d)$ for $d = 3$ and $d = 4$.

A. $K_G(4) \geqslant 1.4731$ using a trial-and-error method

We fix $n = 92$ and generate $\vec{a}_i = \vec{b}_i$ in $d = 4$ by optimizing the energy (7), from which we get the matrix C with entries,

$$C_{i,j} = \vec{a}_i \cdot \vec{b}_j. \tag{8}$$

With this in hand, we have to choose the form of the $n \times n$ matrix M. First we would like to demand that the matrix entries $M_{i,j}$ are some function of the entries $C_{i,j}$. Hence we define them as

$$M_{i,j} = [f(C_{i,j})], \tag{9}$$

where we choose the form of the periodic function f as

$$f(q) = 80\sin(\pi q/2) + 100\sin(3\pi q/2), \tag{10}$$

and $[x]$ denotes the nearest integer to x. Rounding has been introduced in order for the entries of M to be integer. On the other hand, the constants appearing in the function (10) are chosen by trial and error such that they would provide good performance, i.e., large lower bound values for $K_G(4)$. The next section (Sec. III B), which uses the distance method to lowerbound $K_G(4)$, will also shed light on the specific choice of the function (10).

Using the function in Eq. (10), explicit calculations give $Q(M,4) = \sum_{i,j} M_{i,j}C_{i,j} \simeq 2.6785 \times 10^5$ in Eq. (3). On the other hand, $L(M) = 181\,818$ coming from our BB algorithm, which took roughly three months to evaluate on a desktop computer. Then the lower bound of $K_G(4) \geqslant Q(M,4)/L(M) \simeq 1.4731$ follows from formula (4). A MATHEMATICA file provides all the details of the matrices involved in the computation [29]. It is noted that the algorithm has very low memory requirements. We also recall that the $L(M)$ value does not depend on a specific ordering of the rows and columns of M. In this respect, we found that the running time is quite sensitive to the ordering of the rows, and it is worth trying different orderings to improve time efficiency. We next present an improved lower bound to $K_G(4)$, which uses the distance method [16,19] to generate the function f in Eq. (9).

B. $K_G(4) \geqslant 1.4821$ using the distance method

Here we give a specific M matrix using the Gilbert's distance method [16,19]. In this way, we get further improvement on the lower bound to $K_G(4)$ presented in the previous section.

Let us first briefly describe Gilbert's distance algorithm. It estimates the distance between a point P and an arbitrary convex set S in some finite-dimensional Euclidean space via calls to an oracle which performs linear optimizations over S. In our particular case, the point is given by $P = vC$, that is, the correlation matrix C in Eq. (8) multiplied by a factor $0 < v \leqslant 1$. The convex set in our case is the so-called ± 1 polytope, which is defined by the convex hull of its vertices as follows. For a given n, the dimension of the polytope is $n \times n$, and vertices D_λ are given by matrices with entries $D_\lambda(i,j) = a_i b_j$, where λ corresponds to a specific assignment of $a_i = \pm 1, i = 1,\ldots,n$ and $b_j = \pm 1$, $j = 1,\ldots,n$. This amounts to 2^{2n-1} distinct vertices D_λ. Any point inside the polytope is a convex combination of vertices D_λ with positive weights $p(\lambda)$.

The factor v is chosen in such a way that point $P = vC$ lies (slightly) outside the ± 1 polytope. To this end, let us choose C from Eq. (8) along with $v^* = 1/1.4731 \simeq 0.6788$, where 1.4731 corresponds to the lower bound $K_G(4) \geqslant 1.4731$ obtained in the preceding section. By definition, the point v^*C is outside the ± 1 polytope. Then we call the distance algorithm [16,19] where the inputs to the problem are the point v^*C and the description of the ± 1 polytope. The algorithm outputs (an estimate to) the distance between the point v^*C and the

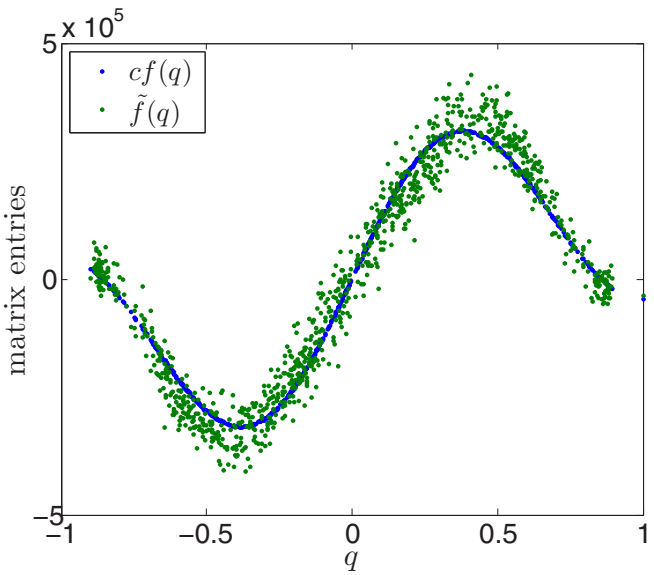

FIG. 2. The functions $f(q)$ and $\tilde{f}(q)$ are displayed in blue and green dots, respectively. An (irrelevant) multiplicative constant $c = 2200$ is introduced for better comparison of the two plots. The feasible q values on the x axis correspond to the relation $q = \vec{a}_i \cdot \vec{b}_j$.

± 1 polytope by providing a separating hyperplane with norm M, which is identified with the $n \times n$ matrix M that we are looking for.

The obtained matrix M (after rounding to integers) is given in a MATHEMATICA file; see Ref. [29]. Explicit calculations show that $Q(M,4) = \sum_{i,j} M_{i,j} C_{i,j} \simeq 6.2223 \times 10^8$ in Eq. (3). On the other hand, the BB algorithm evaluates $L(M) = 419\,810\,256$, which took about three months on our desktop computer. Put together, we get from formula (4) the improved lower bound $K_G(4) \geqslant Q(M,4)/L(M) \simeq 1.4821$.

In the actual implementation of the distance algorithm, we projected the problem from the space of $n \times n$ matrices to a smaller subspace such that the entries $M_{i,j}$ of M are given by $M_{i,j} = \tilde{f}(C_{i,j})$. In this way, one can compare the two functions f and $\tilde{f}$, where f is given by Eq. (10) and $\tilde{f}$ results from the distance algorithm in the present section. The two functions are shown in Fig. 2. According to the figure, the blue dots (representing f) readily well approximate the scattered green dots (representing $\tilde{f}$) within the full range of q and can be considered as a coarse-grained version of it. Using $\tilde{f}$ compared to f in the definition of matrix M gives us the improved lower bound $K_G(4) \geqslant 1.4821$ compared to $K_G(4) \geqslant 1.4731$.

C. $K_G(3) \geqslant 1.4359$ using the distance method

We optimized the energy formula (7) by running the Amoeba method for $d = 3$ and $n = 90$, and fixing icosahedral symmetry. In this way, we obtained the unit vectors $\vec{a}_i$, $i = 1, \ldots, n$, on the sphere. Then, similarly to Sec. III B, the distance algorithm was consulted to compute matrix M. This matrix M, whose entries are rounded to the closest integers, is given in a MATHEMATICA file [29]. With this M and C, we have $Q(M,3) = \sum_{i,j} M_{i,j} C_{i,j} \simeq 4.6560 \times 10^8$. On the other hand, $L(M) = 324\,230\,014$ due to the BB algorithm, where the running time was two weeks on our desktop computer. Then we get from (4) the lower bound $K_G(3) \geqslant Q(M,3)/L(M) \simeq 1.4359$. It is noted that this value gives the best upper bound of $v = 1/1.4359 \simeq 0.6964$ on the critical visibility of the two-qubit Werner states improving on recent upper bounds [31,32].

IV. LINK TO BELL NONLOCALITY

The Grothendieck constant has a direct link to quantum nonlocality problems [33–35], which we discuss briefly below. A detailed survey of this connection can be found in Ref. [36].

In a quantum Bell-like experiment, two parties perform local measurements on a shared entangled state [33]. Let Alice and Bob share a state ρ in $\mathbb{C}^D \otimes \mathbb{C}^D$ and perform two-outcome projective measurements described by observables A_x and B_y, which are D-dimensional Hermitian matrices with eigenvalues $\{\pm 1\}$. Here, $x, y = 1, \ldots, n$ label the measurement settings. Then the correlator, which is the expectation value of the product of Alice and Bob's ± 1 outcomes, is

$$c_{x,y} = \mathrm{tr}(\rho A_x \otimes B_y) \tag{11}$$

for given settings x and y. Such correlations associated with XOR nonlocal games are frequently studied in the computer science literature [21].

Given a dimension D, one wonders if a set of correlators $\{c_{x,y}, x, y = 1, \ldots, n\}$ in Eq. (11) is quantum realizable with a state $\rho \in \mathbb{C}^D \otimes \mathbb{C}^D$ using arbitrary positive operator-valued measure (POVM) measurements, and also allowing Alice and Bob to share an arbitrary large amount of randomness. If it happens not to be the case, we say that the set of correlators $\{c_{x,y}\}$ is not D-dimensional quantum realizable. A convenient tool to address this problem is the use of dimension witnesses [25]. In what follows, we show that our main result $K_G(3) < K_G(4)$ implies a set of correlators $\{c_{x,y}\}$, which are not two-dimensional quantum realizable. This result is based on earlier works [22,24,37], and the argument is as follows.

Let us consider a matrix M' and unit vectors $\vec{a}'_x \in \mathbb{R}^4$ and $\vec{b}'_y \in \mathbb{R}^4$ in formula (2) which give rise to the exact value of $K_G(4)$. It is noted that though the exact value of $K_G(4)$ is unknown, there must exist some matrix M' (of possibly infinite dimension n) and unit vectors $\vec{a}'_x, \vec{b}'_y$ in the four-dimensional Euclidean space which give rise to $K_G(4)$.

Tsirelson [22] has shown that all correlators $c_{x,y}$ equal to dot products $\vec{a}_x \cdot \vec{b}_y$ of the unit vectors $\vec{a}_x, \vec{b}_y \in \mathbb{R}^4$ are realizable as observables,

$$A_x = \sum_{i=1}^{4} a_{x,i} \gamma_i,$$
$$B_y = \sum_{i=1}^{4} b_{y,i} \gamma_i^t, \tag{12}$$

on a pair of maximally entangled four-dimensional quantum systems, $|\psi_4\rangle = (1/2) \sum_{i=1}^{4} |i\rangle|i\rangle$. Here, $a_{x,i}, b_{y,i}$ are entries of the four-dimensional unit vectors $\vec{a}_x, \vec{b}_y \in \mathbb{R}^4$, respectively, t denotes the transposition, and γ_i are chosen as follows:

$$\gamma_1 = \sigma_x \otimes \mathbb{1},$$
$$\gamma_2 = \sigma_y \otimes \mathbb{1},$$

$$\gamma_3 = \sigma_z \otimes \sigma_x,$$
$$\gamma_4 = \sigma_z \otimes \sigma_z. \tag{13}$$

The above γ_i matrices are traceless, anticommuting, and square to the identity. Due to these properties, A_x, B_y are valid traceless observables: $\mathrm{tr}(A_x) = \mathrm{tr}(B_y) = 0$ and $A_x^2 = B_y^2 = \mathbb{1}$. On the other hand, one has

$$c_{x,y} = \mathrm{tr}(|\psi_4\rangle\langle\psi_4| A_x \otimes B_y) = \mathrm{tr}\big(A_x B_y^t\big)/4. \tag{14}$$

Applying Eq. (12) and noting that $\mathrm{tr}(\gamma_i \gamma_j^t) = 4\delta_{i,j}$, where $\delta_{i,j}$ denotes the Kronecker delta, we further have

$$c_{x,y} = \mathrm{tr}\big(A_x B_y^T\big)/4 = \sum_i a_{x,i} b_{y,i} = \vec{a}_x \cdot \vec{b}_y. \tag{15}$$

Replacing $\vec{a}_x$ and $\vec{b}_y$ with the particular vectors $\vec{a}'_x, \vec{b}'_y$, which leads to the exact value of $K_G(4)$, we obtain that the correlators $c'_{x,y} = \vec{a}'_x \cdot \vec{b}'_y$ are realizable as observables A'_x, B'_y on the state $|\psi_4\rangle = (1/2)\sum_{i=1}^4 |i\rangle|i\rangle$. We next show that this set of correlators $c'_{x,y}$ has no two-dimensional quantum representation, i.e., it cannot be realized using qubit systems. The proof exploits the strict relation $K_G(3) < K_G(4)$, which we have proven in the preceding sections. To this end, let us consider the linear function I on the correlators $c_{x,y}$ in Eq. (11) written as

$$I = \sum M'_{x,y} c_{x,y}, \tag{16}$$

where $M'_{x,y}$ is defined by the matrix M' which attains the exact value of $K_G(4)$ in (2). Let us then denote by $I^{(2)}$ the maximum of I it can take if the correlators $c_{x,y}$ come from two-dimensional quantum systems. It appears that $I^{(2)}$ is defined by

$$I^{(2)} = \max \sum_{x,y=1}^{m} M'_{x,y} \vec{a}_x \cdot \vec{b}_y, \tag{17}$$

where maximization is over all three-dimensional unit vectors $\vec{a}_x$ and $\vec{b}_y$ [25,37]. However, by definition (2), $I^{(2)}$ is upper bounded by $K_G(3)L(M')$, where the function L is defined by Eq. (1). Therefore, we have the chain of inequalities

$$\frac{I^{(2)}}{L(M')} \leqslant K_G(3) < K_G(4) = \frac{\sum_{x,y} M'_{x,y} c'_{x,y}}{L(M')}, \tag{18}$$

where comparing the leftmost and the rightmost terms gives us the strict relation

$$I^{(2)} < \sum_{x,y} M'_{x,y} c'_{x,y}. \tag{19}$$

This tells us that the expression (16) cannot be saturated by correlations originating from qubit systems. Hence, the above example shows the existence of a witness-detecting dimension greater than two in the particular case where the witness matrix M' is associated with the $K_G(4)$ value in Eq. (2). A similar argument in Ref. [25] has shown the existence of a qutrit witness from the strict relation $K_G(3) < K_G$. Here we showed that it suffices to consider a pair of four-dimensional quantum systems to certify correlations beyond qubit. Note also that dimension witnesses including any finite-dimension D appeared in the literature based on different methods; see, e.g., Refs. [38–42]. More recent works [43–46] revealed further intriguing properties of the restricted dimensional quantum sets.

V. DISCUSSION

We proved that $K_G(4)$ is strictly larger than $K_G(3)$. To this end, we used the so-called branch-and-bound algorithm commonly used in the solution of NP-hard problems. This allowed us to solve the problem (1) up to matrix sizes 92 on a standard desktop PC. Further, due to the principles of the branch-and-bound algorithm (i.e., the calculation of the bounds and the branching in each node is independent), it is a natural idea to adapt the algorithm to a graphics processing unit (GPU), grid computing, or field-programmable gate array (FPGA).

As we have shown, our result is relevant in quantum nonlocality, as one can construct a dimension witness for detecting dimension greater than two based on the relation $K_G(3) < K_G(4)$. Hence, we provided an application of the branch-and-bound technique in the context of quantum nonlocality, particularly in XOR nonlocal games. However, we expect that the presented algorithm in combination with other powerful methods (such as Gilbert's distance algorithm) may find applications beyond XOR nonlocal games as well. Such possible tasks concern Bell nonlocality with more inputs [47,48], more outcomes [49], or genuine nonlocality in the case of multipartite settings [50,51]. Note a recent method [52] based on the Navascues-Pironio-Acin hierarchy [53], which tackles these problems in a different way. Combining the two approaches may also lead to improvement in our bipartite setting.

It would also be interesting to adapt the branch-and-bound technique to bound the so-called unsteerability limit in Einstein-Podolsky-Rosen (EPR) steering inequalities [54,55] with large number of inputs. Finally, the algorithm is likely applicable in random access codes [56] or noncontextuality inequalities as well [57,58] with large number of input-output alphabets.

ACKNOWLEDGMENTS

We thank N. Brunner, N. Gisin, J. Kaniewski, Y.C. Liang, and M. Navascués for useful discussions. Technical assistance from B. Kőműves is gratefully acknowledged. We acknowledge financial support from the Hungarian National Research Fund OTKA (Grant No. K111734).

APPENDIX: DESCRIPTION AND IMPLEMENTATION DETAILS OF THE $K_{m,n}$ PROGRAMMING ALGORITHM

1. Introduction

$K_{m,n}$-quadratic programming is a quadratic optimization problem with binary variables. In the main text, the algorithm to solve $K_{m,n}$-quadratic programming is defined (where we have set $m = n$). In this appendix, we prove the correctness of the algorithm. We also provide tips about the efficient implementation of the algorithm on a desktop computer. An implementation of this algorithm with application in XOR nonlocal games is publicly available at [29].

2. Notation

$\mathbb{N}$	set of natural numbers
$\mathbb{Z}$	set of integers
A^n	n-ary Cartesian product $A \times \cdots \times A$
v_i	ith coordinate of $v \in A^n$, $i = 1,2,\ldots,n$
$(v_1, v_2, \ldots, v_n)$	construction of $v \in A^n$
$\|v\|_1$	Manhattan norm, i.e., $\sum_i \|v_i\|$
$\mathcal{M}_{n\times m}(A)$	matrices with n rows and m columns over A
M_i	ith row of $M \in \mathcal{M}_{n\times m}$, $i = 1,2,\ldots,n$

3. $K_{m,n}$-Quadratic Programming

Let M be an $n \times m$ matrix of integers. The goal of $K_{m,n}$-quadratic programming is to efficiently compute the L function, which is defined as follows:

$$L : \mathcal{M}_{n\times m}(\mathbb{Z}) \to \mathbb{Z}$$
$$L(M) := \max_{a_i=\pm1, b_j=\pm1} \sum_{i=1}^{n}\sum_{j=1}^{m} M_{ij} a_i b_j$$

Basic properties

Theorem.

$$L(M) = \max_{a_i=\pm1} \left\| \sum_{i=1}^{n} a_i M_i \right\|_1 .$$

Proof.

$$\begin{aligned}\max_{a_i=\pm1} \left\| \sum_{i=1}^{n} a_i M_i \right\|_1 &= \max_{a_i=\pm1} \sum_{j=1}^{m} \left| \sum_{i=1}^{n} a_i M_{ij} \right| \\ &= \max_{a_i=\pm1} \max_{b_j=\pm1} \sum_{j=1}^{m} b_j \left(\sum_{i=1}^{n} a_i M_{ij} \right) \\ &= \max_{a_i=\pm1, b_j=\pm1} \sum_{i=1}^{n}\sum_{j=1}^{m} M_{ij} a_i b_j = L(M).\end{aligned}$$

Theorem.

Let $M \in \mathcal{M}_{n\times m}$, $M = (M_1, M_2, \ldots, M_n)$, where M_i is the ith row of M:

$$L(M) = \max[L(M^+), L(M^-)],$$

where $M^+ = (M_1 + M_2, M_3, M_4, \ldots, M_n)$ and $M^- = (M_1 - M_2, M_3, M_4, \ldots, M_n)$.

Proof.

$$\begin{aligned}L(M) &= \max_{a_i=\pm1} \left\| \sum_{i=1}^{n} a_i M_i \right\|_1 \\ &= \max \left(\max_{a_i=\pm1, a_1=a_2} \left\| \sum_{i=1}^{n} a_i M_i \right\|_1 , \right. \\ &\qquad \left. \max_{a_i=\pm1, a_1\neq a_2} \left\| \sum_{i=1}^{n} a_i M_i \right\|_1 \right) \\ &= \max[L(M^+), L(M^-)].\end{aligned}$$

Theorem.

Let $M \in \mathcal{M}_{n\times m}$, $M = (M_1, M_2, \ldots, M_n)$, where M_i is the ith row of M:

$$L(M) \leqslant L(M^U) + L(M^L),$$

where $M^U = (M_1, M_2, \ldots, M_k)$ and $M^L = (M_{k+1}, M_{k+2}, \ldots, M_n)$.

Proof.

$$\begin{aligned}L(M) &= \max_{a_i=\pm1} \left\| \sum_{i=1}^{n} a_i M_i \right\|_1 \\ &\leqslant \max_{a_i=\pm1} \left(\left\| \sum_{i=1}^{k} a_i M_i \right\|_1 + \left\| \sum_{i=k+1}^{n} a_i M_i \right\|_1 \right) \\ &= \max_{a_i=\pm1} \left\| \sum_{i=1}^{k} a_i M_i \right\|_1 + \max_{a_i=\pm1} \left\| \sum_{i=k+1}^{n} a_i M_i \right\|_1 \\ &= L(M^U) + L(M^L).\end{aligned}$$

4. Recursive Calculation of L

We define a recursive function f to calculate L. The function f is not efficient, but it helps to understand the efficient functions defined later and it is also used in their correctness proofs.

Let $n,m \in \mathbb{N}$.

Let $M = (M_1, M_2, \ldots, M_n) \in \mathcal{M}_{n\times m}$.

M is a fixed parameter of the function f so it is placed in the subscript as f_M.

The recursive function f_M is defined as

$$f_M : \mathbb{N} \times \mathbb{Z}^m \to \mathbb{Z}$$
$$f_M(k,v) := \begin{cases} \|v\|_1 & \text{if } k = n, \\ \max[f_M(k+1, v + M_{k+1}), f_M(k+1, v - M_{k+1})] & \text{otherwise.} \end{cases}$$

Theorem.

$$f_M(k,v) = L((v, M_{k+1}, M_{k+2}, \ldots, M_n)).$$

Proof.

By induction on $k = n, n-1, n-2, \ldots$:

(i) Base case: $k = n$.

$$f_M(k,v) = \|v\|_1 = L((v)) = L((v, M_{k+1}, \ldots, M_n)).$$

(ii) Inductive step: $k < n$.

$$\begin{aligned}f_M(k,v) &= \max[f_M(k+1, v + M_{k+1}), f_M(k+1, v - M_{k+1})] \\ &= \max[L((v + M_{k+1}, M_{k+2}, \ldots, M_n)), \\ &\quad L((v - M_{k+1}, M_{k+2}, \ldots, M_n))] \\ &= L((v, M_{k+1}, M_{k+2}, \ldots, M_n)).\end{aligned}$$

Corollary.

$$L(M) = f_M(1, M_1).$$

Example.

Let $M = \begin{pmatrix} 2 & 3 & 3 & 0 \\ 3 & 2 & -3 & -3 \\ 3 & -3 & 2 & 3 \\ 0 & -3 & 3 & 2 \end{pmatrix} \in \mathcal{M}_{4\times 4}$.

$$\begin{aligned}
L(M) &= f_M(1,(2,3,3,0)) \\
&= \max[f_M(2,(5,5,0,-3)), f_M(2,(-1,1,6,3))] \\
&= \max\{\max[f_M(3,(8,2,2,0)), f_M(3,(2,8,-2,-6))], \\
&\quad \max[f_M(3,(2,-2,8,6)), f_M(3,(-4,4,4,0))]\} \\
&= \max(\max\{\max[f_M(4,(8,-1,5,2)), \\
&\quad f_M(4,(8,5,-1,-2))], \\
&\quad \max[f_M(4,(2,5,1,-4)), f_M(4,(2,11,-5,-8))]\}, \\
&\quad \max\{\max[f_M(4,(2,-5,11,8)), f_M(4,(2,1,5,4))], \\
&\quad \max[f_M(4,(-4,1,7,2)), f_M(4,(-4,7,1,-2))]\}) \\
&= \max(\max\{\max[\|(8,-1,5,2)\|_1, \|(8,5,-1,-2)\|_1], \\
&\quad \max[\|(2,5,1,-4)\|_1, \|(2,11,-5,-8)\|_1]\}, \\
&\quad \max\{\max[\|(2,-5,11,8)\|_1, \|(2,1,5,4)\|_1], \\
&\quad \max[\|(-4,1,7,2)\|_1, \|(-4,7,1,-2)\|_1)]\}) \\
&= \max\{\max[\max(16,16), \max(12,26)], \\
&\quad \max[\max(26,12), \max(14,14)]\} \\
&= 26.
\end{aligned}$$

Note that the total number of $f_M(k,v)$ calls is $1+2+4+\cdots+2^{n-1} = 2^n - 1 = 15$.

5. Speeding up The Recursion

We define another recursive function, g, to calculate L. g is more efficient than f because it tries to skip whole branches of recursive calls by comparing the best found maximum so far and the estimated result of the branch.

Let $c_k \geqslant L((M_{k+1},M_{k+2},\ldots,M_n))$ arbitrary constants, where $k = 1,2,\ldots,n-1$. We discuss later how to choose c_k. Let $c_n = 0$.

Let $c = (c_1,c_2,\ldots,c_n) \in \mathbb{Z}^n$.

M and c are fixed parameters of the function g so they are placed in the subscript as $g_{M,c}$.

$g_{M,c}$ is defined as

$$g_{M,c} : \mathbb{N} \times \mathbb{Z}^m \times \mathbb{Z} \to \mathbb{Z}$$

$$g_{M,c}(k,v,m) := \begin{cases} m & \text{if } m \geqslant \|v\|_1 + c_k, \\ \|v\|_1 & \text{otherwise if } k = n, \\ g_{M,c}(k+1, v - M_{k+1}, g_{M,c}(k+1, v + M_{k+1}, m)) & \text{otherwise.} \end{cases}$$

Theorem.

$$g_{M,c}(k,v,m) = \max[f_M(k,v),m].$$

Proof.

By induction on $k = n,n-1,n-2,\ldots$:

(i) Base case: $k = n$.

(a) Case $m \geqslant \|v\|_1$.

$$g_{M,c}(k,v,m) = m = \max(\|v\|_1,m) = \max[f_M(k,v),m].$$

(b) Case $m < \|v\|_1$.

$$g_{M,c}(k,v,m) = \|v\|_1 = \max(\|v\|_1,m) = \max[f_M(k,v),m].$$

(ii) Inductive step: $k < n$.

(a) Case $m \geqslant \|v\|_1 + c_k$.

$$\begin{aligned}
g_{M,c}(k,v,m) &= m \\
&= \max(\|v\|_1 + c_k, m) \\
&= \max[L((v,M_{k+1},\ldots,M_n)),m] \\
&= \max[f_M(k,v),m].
\end{aligned}$$

(b) Case $m < \|v\|_1 + c_k$.

$$\begin{aligned}
g_{M,c}(k,v,m) &= g_{M,c}(k+1, v - M_{k+1}, g_{M,c}(k+1, v + M_{k+1}, m)) \\
&= \max\{f_M(k+1, v - M_{k+1}), \max[f_M(k+1, v + M_{k+1}), m]\} \\
&= \max\{\max[f_M(k+1, v - M_{k+1}), f_M(k+1, v + M_{k+1})], m\} \\
&= \max[f_M(k,v),m].
\end{aligned}$$

Corollary.

$$L(M) = g_{M,c}(1, M_1, 0).$$

Choosing c_k

The c_k constants can be chosen arbitrarily unless they are greater than or equal to $L((M_{k+1}, M_{k+2}, \ldots, M_n))$. Lower c_k constants prevent more $g_{M,c}(k,v,m)$ computations. There is a trade-off between computing the lower bound of c_k to speed up later computations, and using less resources on c_k and doing more computation later.

The lower bound of c_k can also be computed with $g_{M,c}$:

$$L((M_{k+1}, M_{k+2}, \ldots, M_n)) = g_{M,c}(k+1, M_{k+1}, 0).$$

We have found by experience that it is worthwhile to compute the lower bound of $c_n, c_{n-1}, \ldots, c_i$, and set the remaining $c_{i-1}, c_{i-2}, \ldots, c_2$ constants to ∞, where i is around $\lceil n/4 \rceil$. Note that during the computation of the lower bound of c_k, the $c_j, j > k$ constants are also needed, so one should compute the lower bounds of $c_n, c_{n-1}, \ldots, c_i$ one after another in this order.

Example.

Let $M = \begin{pmatrix} 2 & 3 & 3 & 0 \\ 3 & 2 & -3 & -3 \\ 3 & -3 & 2 & 3 \\ 0 & -3 & 3 & 2 \end{pmatrix} \in \mathcal{M}_{4\times 4}$.

$$\begin{aligned}
c_4 &= 0 \quad \text{by definition.} \\
c_3 &:= L((M_4)) \quad \text{choose the lower bound} \\
&= g_{M,c}(4,(0,-3,3,2),0) \\
&= \|(0,-3,3,2)\|_1 \quad \text{because } 0 \not\geq \|(0,-3,3,2)\|_1 + c_4 \\
&= 8. \\
c_2 &:= \infty \quad \text{avoid computation of } L((M_3, M_4)). \\
c_1 &:= \infty \quad \text{avoid computation of } L((M_2, M_3, M_4)). \\
L(a) &= g_{M,c}(1,(2,3,3,0),0) \\
&= g_{M,c}[2,(-1,1,6,3), g_{M,c}(2,(5,5,0,-3),0)] \\
&= g_{M,c}\{2,(-1,1,6,3), g_{M,c}[3,(2,8,-2,-6), g_{M,c}(3,(8,2,2,0),0)]\} \\
&= g_{M,c}(2,(-1,1,6,3), g_{M,c}\{3,(2,8,-2,-6), g_{M,c}[4,(8,5,-1,-2), g_{M,c}(4,(8,-1,5,2),0)]\}) \\
&= g_{M,c}\{2,(-1,1,6,3), g_{M,c}[3,(2,8,-2,-6), g_{M,c}(4,(8,5,-1,-2), \|(8,-1,5,2)\|_1)]\} \\
&= g_{M,c}\{2,(-1,1,6,3), g_{M,c}[3,(2,8,-2,-6), g_{M,c}(4,(8,5,-1,-2),16)]\} \\
&= g_{M,c}[2,(-1,1,6,3), g_{M,c}(3,(2,8,-2,-6),16)] \\
&= g_{M,c}\{2,(-1,1,6,3), g_{M,c}[4,(2,11,-5,-8), g_{M,c}(4,(2,5,1,-4),16)]\} \\
&= g_{M,c}[2,(-1,1,6,3), g_{M,c}(4,(2,11,-5,-8),16)] \\
&= g_{M,c}(2,(-1,1,6,3), \|(2,11,-5,-8)\|_1) \\
&= g_{M,c}(2,(-1,1,6,3),26) \\
&= g_{M,c}[3,(-4,4,4,0), g_{M,c}(3,(2,-2,8,6),26)] \\
&= g_{M,c}(3,(-4,4,4,0),26) \quad \text{optimization kicks in} \\
&= 26. \quad \text{optimization kicks in.}
\end{aligned}$$

Note that the total number of $g_{M,c}(k,v,m)$ calls is 12.

6. Tail-Recursive Form

It is possible to refactor $g_{M,c}$ into two mutually tail-recursive functions $d_{M,c}$ and $u_{M,c}$ such that each recursive call is a tail call, i.e., there are no further operations involved after the call is completed [59]. Tail calls can be implemented by goto statements so they do not need stack operations, which is a requirement on a GPU and also speeds up computation on a CPU.

The definitions of $d_{M,c}$ and $u_{M,c}$ are

$$
\begin{array}{l}
d_{M,c},u_{M,c} : \mathbb{N}\times\mathbb{N}\times\mathbb{Z}^m\times\mathbb{N}\to\mathbb{N} \\
d_{M,c}(k,b,v,m) := \begin{cases} u_{M,c}(k,b,v,m) & \text{if } m \geqslant \|v\|_1 + c_k, \\ u_{M,c}(k,b,v,\|v\|_1) & \text{otherwise if } k = n, \\ d_{M,c}(k+1,2b,v+M_{k+1},m) & \text{otherwise.} \end{cases} \\
u_{M,c}(k,b,v,m) := \begin{cases} m & \text{if } k = 1, \\ d_{M,c}(k,b+1,v-2M_k,m) & \text{otherwise if } b = 2b', \\ u_{M,c}(k-1,b',v+M_k,m) & \text{otherwise if } b = 2b'+1. \end{cases}
\end{array}
$$

Theorem.
For all $k \in \mathbb{N}, 1 \leqslant k \leqslant n$ and $a_2,a_3,\ldots,a_k = \pm 1$,

$$
d_{M,c}\left[k,\overline{a_2a_3\cdots a_k},M_1+\sum_{i=2}^{k}a_iM_i,\ \max_{\frac{b_2,b_3,\ldots,b_n=\pm1}{\overline{b_2b_3\cdots b_k}<\overline{a_2a_3\cdots a_k}}}\left(M_1+\sum_{i=2}^{n}b_iM_i\right)\right] = L(M),
$$

$$
u_{M,c}\left[k,\overline{a_2a_3\cdots a_k},M_1+\sum_{i=2}^{k}a_iM_i,\ \max_{\frac{b_2,b_3,\ldots,b_n=\pm1}{\overline{b_2b_3\cdots b_k}\leqslant\overline{a_2a_3\cdots a_k}}}\left(M_1+\sum_{i=2}^{n}b_iM_i\right)\right] = L(M),
$$

where

$$
\overline{x_1x_2\cdots x_i} := \sum_{j=1}^{i}\frac{1-x_j}{2}2^{i-j},
$$

and

$$
\max_{x\in\emptyset} x = 0.
$$

Sketch of the proof.
One has to show that if the parameters of $d_{M,c}$ and $u_{M,c}$ are in the form given in the theorem, then in each possible case, the next call of $d_{M,c}$ or $u_{M,c}$ has parameters in the form given in the theorem too.

Corollary.

$$
L(M) = d_{M,c}(1,0,M_1,0).
$$

Example.
Let $M = \begin{pmatrix} 2 & 3 & 3 & 0 \\ 3 & 2 & -3 & -3 \\ 3 & -3 & 2 & 3 \\ 0 & -3 & 3 & 2 \end{pmatrix} \in \mathcal{M}_{4\times4}$.
Let $c_4 = 0, c_3 = 8, c_2 = \infty, c_1 = \infty$ as in the previous example.

$$
\begin{aligned}
L(M) &= d_{M,c}(1,0,(2,3,3,0),0) \\
&= d_{M,c}(2,0,(5,5,0,-3),0) \\
&= d_{M,c}(3,0,(8,2,2,0),0) \\
&= d_{M,c}(4,0,(8,-1,5,2),0) && \text{where } \|(8,-1,5,2)\|_1 = 16 \\
&= u_{M,c}(4,0,(8,-1,5,2),16) && \text{where } 0 = 2\cdot0 \\
&= d_{M,c}(4,1,(8,5,-1,-2),16) \\
&= u_{M,c}(4,1,(8,5,-1,-2),16) && \text{where } 1 = 2\cdot0+1 \\
&= u_{M,c}(3,0,(8,2,2,0),16) && \text{where } 0 = 2\cdot0 \\
&= d_{M,c}(3,1,(2,8,-2,-6),16) \\
&= d_{M,c}(4,2,(2,5,1,-4),16) \\
&= u_{M,c}(4,2,(2,5,1,-4),16) && \text{where } 2 = 2\cdot1
\end{aligned}
$$

$$
\begin{aligned}
&= d_{M,c}(4,3,(2,11,-5,-8),16) && \text{where } \|(2,11,-5,-8)\|_1 = 26\\
&= u_{M,c}(4,3,(2,11,-5,-8),26) && \text{where } 3 = 2\cdot 1+1\\
&= u_{M,c}(3,1,(2,8,-2,-6),26) && \text{where } 1 = 2\cdot 0+1\\
&= u_{M,c}(2,0,(5,5,0,-3),26) && \text{where } 0 = 2\cdot 0\\
&= d_{M,c}(2,1,(-1,1,6,3),26)\\
&= d_{M,c}(3,2,(2,-2,8,6),26) && \text{where } 26 \geqslant \|(2,-2,8,6)\|_1 + 8\\
&= u_{M,c}(3,2,(2,-2,8,6),26) && \text{where } 2 = 2\cdot 1\\
&= d_{M,c}(3,3,(-4,4,4,0),26) && \text{where } 26 \geqslant \|(-4,4,4,0)\|_1 + 8\\
&= u_{M,c}(3,3,(-4,4,4,0),26) && \text{where } 3 = 2\cdot 1+1\\
&= u_{M,c}(2,1,(-1,1,6,3),26) && \text{where } 1 = 2\cdot 0+1\\
&= u_{M,c}(1,0,(2,3,3,0),26) && \text{where } k = 1\\
&= 26.
\end{aligned}
$$

[1] A. Grothendieck, Résumé of the metric theory of tensor products of topological vector spaces, Bol. Soc. Mat. Sao Paulo **8**, 1 (1953).

[2] G. Pisier, *Factorization of Linear Operators and Geometry of Banach Spaces* (American Mathematical Society, Providence, RI, 1986); Grothendieck's theorem, past and present, Bull. Amer. Math. Soc. **49**, 237 (2012).

[3] N. Linial, S. Mendelson, G. Schechtman, and A. Shraibman, Complexity measures of sign matrices, Combinatorica **27**, 439 (2008).

[4] N. Alon and A. Naor, Approximating the cut-norm via Grothendieck's inequality, SIAM J. Comput. **35**, 787 (2006).

[5] A. Frieze and R. Kannan, Quick approximation to matrices and applications, Combinatorica **19**, 175 (1999).

[6] A. M. Davie (unpublished) and J. A. Reeds (unpublished).

[7] J. L. Krivine, Grothendieck's constants and positive functions on spheres, Adv. Math. **31**, 16 (1979).

[8] P. Raghavendra and D. Steurer, Towards computing the Grothendieck constant, in *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms* (Society for Industrial and Applied Mathematics, 2009), pp. 525–534.

[9] I. Pitowsky, New Bell inequalities for the singlet state: Going beyond the Grothendieck bound, J. Math. Phys. **49**, 012101 (2008).

[10] S. R. Finch, *Mathematical Constants, ser. Encyclopedia of Mathematics and its Applications* (Cambridge University Press, Cambridge, 2003).

[11] J. F. Clauser, M. A. Horne, A. Shimony, and R. A. Holt, Proposed Experiment to Test Local Hidden-Variable Theories, Phys. Rev. Lett. **23**, 880 (1969).

[12] P. C. Fishburn and J. A. Reeds, Bell Inequalities, Grothendieck's Constant, and Root Two, SIAM J. Discrete Math. **7**, 48 (1994).

[13] T. Vértesi, More efficient Bell inequalities for Werner states, Phys. Rev. A **78**, 032112 (2008).

[14] B. Hua, M. Li, T. Zhang, C. Zhou, X. Li-Jost, S.-M. Fei, Towards Grothendieck Constants and LHV Models in Quantum Mechanics, J. Phys. A **48**, 065302 (2015).

[15] M. Li, T. Zhang, B. Hua, S.-M. Fei, and X. Li-Jost, Quantum Nonlocality of Arbitrary Dimensional Bipartite States, Sci. Rep. **5**, 13358 (2015).

[16] S. Brierley, M. Navascués, and T. Vértesi, Convex separation from convex optimization for large-scale problems, arXiv:1609.05011.

[17] M. Braverman, K. Makarychev, Y. Makarychev, and A. Naor, The Grothendieck constant is strictly smaller than Krivine's bound, arXiv:1103.6161.

[18] F. Hirsch, M. T. Quintino, T. Vértesi, M. Navascués, and N. Brunner, Better local hidden variable models for two-qubit Werner states and an upper bound on the Grothendieck constant $K_G(3)$, Quantum **1**, 3 (2017).

[19] E. G. Gilbert, An iterative procedure for computing the minimum of a quadratic form on a convex set, SIAM J. Control Optim. **4**, 61 (1966).

[20] A. H. Land and A. G. Doig, An automatic method of solving discrete programming problems, Econometrica **28**, 497 (1960).

[21] R. Cleve, P. Hoyer, B. Toner, and J. Watrous, Consequences and limits of nonlocal strategies, in *Proceedings of the 19th IEEE Annual Conference on Computational Complexity* (IEEE, 2004), pp. 236–249.

[22] B. S. Tsirelson, Quantum analogues of the Bell inequalities. The case of two spatially separated domains, J. Sov. Math. **36**, 557 (1987).

[23] B. Tsirelson, Some results and problems on quantum Bell-type inequalities, Hadronic J. Suppl. **8**, 329 (1993).

[24] A. Acín, N. Gisin, and B. Toner, Grothendieck's constant and local models for noisy entangled quantum states, Phys. Rev. A **73**, 062105 (2006).

[25] N. Brunner, S. Pironio, A. Acin, N. Gisin, A. A. Méthot, and V. Scarani, Testing the Dimension of Hilbert Spaces, Phys. Rev. Lett. **100**, 210503 (2008).

[26] S. Khot, On the power of unique 2-prover 1-round games, in *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing* (ACM, Melville, NY, 2002), pp. 767–775.

[27] R. M. Karp, Reducibility among combinatorial problems, in *Complexity of Computer Computations*, edited by R. E. Miller and J. W. Thatcher (Plenum, New York, 1972), pp. 85.

[28] A. Montina and S. Wolf, Can nonlocal correlations be discriminated in polynomial time?, arXiv:1609.06269.

[29] https://github.com/divipp/kmn-programming (unpublished).

[30] J. A. Nelder and R. Mead, A simplex method for function minimization, Comput. J. **7**, 308 (1965).
[31] R. F. Werner, Quantum states with Einstein-Podolsky-Rosen correlations admitting a hidden-variable model, Phys. Rev. A **40**, 4277 (1989).
[32] N. Gisin, problem 19 (2003), http://qig.itp.uni-hannover.de/qiproblems/19 (unpublished).
[33] J. S. Bell, On the Einstein-Poldolsky-Rosen paradox, Physics **1**, 195 (1964).
[34] N. Brunner, D. Cavalcanti, S. Pironio, V. Scarani, and S. Wehner, Bell nonlocality, Rev. Mod. Phys. **86**, 419 (2014).
[35] R. Augusiak, M. Demianowicz, and A. Acín, Local hidden–variable models for entangled quantum states, J. Phys. A **47**, 424002 (2014).
[36] C. Palazuelos and T. Vidick, Survey on Nonlocal Games and Operator Space Theory, J. Math. Phys. **57**, 015220 (2016).
[37] T. Vértesi and K. F. Pál, Generalized Clauser-Horne-Shimony-Holt inequalities maximally violated by higher dimensional systems, Phys. Rev. A **77**, 042106 (2008).
[38] D. Pérez-Garcia, M. Wolf, C. Palazuelos, I. Villanueva, and M. Junge, Unbounded violation of tripartite bell inequalities, Commun. Math. Phys. **279**, 455 (2008).
[39] T. Vértesi and K. F. Pál, Bounding the dimension of bipartite quantum systems, Phys. Rev. A **79**, 042106 (2009).
[40] J. Briët, H. Buhrman, and B. Toner, A Generalized Grothendieck Inequality and Nonlocal Correlations that Require High Entanglement, Commun. Math. Phys. **305**, 827 (2011).
[41] M. Junge, and C. Palazuelos, Large Violation of Bell Inequalities with Low Entanglement, Commun. Math. Phys. **306**, 695 (2011).
[42] O. Regev, Bell violations through independent bases games, Quantum Inf. Comput. **12**, 9 (2012).
[43] M. Navascués and T. Vértesi, Bounding the Set of Finite Dimensional Quantum Correlations, Phys. Rev. Lett. **115**, 020501 (2015).
[44] J. Sikora, A. Varvitsiotis, and Z. Wei, Minimum Dimension of a Hilbert Space Needed to Generate a Quantum Correlation, Phys. Rev. Lett. **117**, 060401 (2016).
[45] J. I. de Vicente, Shared randomness and device-independent dimension witnessing, Phys. Rev. A **95**, 012340 (2017).
[46] W. Cong, Y. Cai, J.-D. Bancal, and V. Scarani, Irreducible dimension witness, arXiv:1611.01258.
[47] J. Gondzio, J. Gruca, J. Hall, W. Laskowski, and M. Żukowski, Solving Large-Scale Optimization Problems Related to Bell's Theorem, J. Comput. Appl. Math. **263**, 392 (2014).
[48] S. Schwarz, A. Stefanov, S. Wolf, and A. Montina, Optimal measurements for nonlocal correlations, Phys. Rev. A **94**, 022322 (2016).
[49] Y.-C. Liang, C.-W. Lim, and D.-L. Deng, Reexamination of a multisetting Bell inequality for qudits, Phys. Rev. A **80**, 052116 (2009).
[50] G. Svetlichny, Distinguishing three-body from two-body nonseparability by a Bell-type inequality, Phys. Rev. D **35**, 3066 (1987).
[51] J.-D. Bancal, N. Brunner, N. Gisin, and Y.-C. Liang, Detecting Genuine Multipartite Quantum Nonlocality: A Simple Approach and Generalization to Arbitrary Dimensions, Phys. Rev. Lett. **106**, 020405 (2011).
[52] F. Baccari, D. Cavalcanti, P. Wittek, and A. Acín, Efficient device-independent entanglement detection for multipartite systems, Phys. Rev. X **7**, 021042 (2017).
[53] M. Navascués, S. Pironio, and A. Acín, Bounding the Set of Quantum Correlations, Phys. Rev. Lett. **98**, 010401 (2007).
[54] H. M. Wiseman, S. J. Jones, and A. C. Doherty, Steering, Entanglement, Nonlocality, and the Einstein-Podolsky-Rosen Paradox, Phys. Rev. Lett. **98**, 140402 (2007).
[55] D. Cavalcanti and P. Skrzypczyk, Quantum steering: a short review with focus on semidefinite programming, Rep. Prog. Phys. **80**, 024001 (2017).
[56] A. Tavakoli, A. Hameedi, B. Marques, and M. Bourennane, Quantum Random Access Codes Using Single d-Level Systems, Phys. Rev. Lett. **114**, 170502 (2015).
[57] R. W. Spekkens, D. H. Buzacott, A. J. Keehn, Ben Toner, and G. J. Pryde, Preparation Contextuality Powers Parity-Oblivious Multiplexing, Phys. Rev. Lett. **102**, 010401 (2009).
[58] Y.-C. Liang, R. W. Spekkens, and H. M. Wiseman, Specker's parable of the overprotective seer: A road to contextuality, nonlocality and complementarity, Phys. Rep. **506**, 1 (2011).
[59] S. G. Lewis, Jr., Debunking the "expensive procedure call" myth or, procedure call implementations considered harmful or, LAMBDA: The Ultimate GOTO, in *Proceedings of the 1977 Annual Conference* (ACM, Melville, NY, 1977), pp. 153–162.