Mathematical programming bounds for kissing numbers

Leo Liberti

Abstract We give a short review of existing mathematical programming based bounds for kissing numbers. The *kissing number* in K dimensions is the maximum number of unit balls arranged around a central unit ball in such a way that the intersection of the interiors of any pair of balls in the configuration is empty. It is a cornerstone of the theory of spherical codes, a good way to find n equally spaced points on the surface of a hypersphere, and the object of a diatribe between Isaac Newton and David Gregory.

1 A brit and a scot went down the pub...

"Kissing" is billiard jargon. British players would say two adjacent billiard balls on the table "kiss". The term found its way into mathematics thanks to Isaac Newton (whom everyone knows) and David Gregory (a professor of Mathematics at Edinburgh — without having ever obtained a degree — and then Savilian Professor of Astronomy at Oxford thanks to Newton's influence). In the 1690s, scared of the social unrest in Scotland, Gregory left and visited Newton in Cambridge. According to rumours and well-established British protocol, the brit and the scot went down the pub for a few pints of ale and a game of pool. There, among kissing balls and fumes of alcohol, they got into a brawl about the number of balls that could kiss a central ball on the billiard table. Still sober enough, they counted them, and came to agree on the number six. As the number of pints increased, the two started blabbering about gravity-defying floating balls passionately kissing in three dimensions, and disagreed: Newton, embracing the voice of Kepler, said no more than twelve balls could be arranged around a central one. Gregory, who had to gain his master's approval by attempting to be brilliant and surpris-

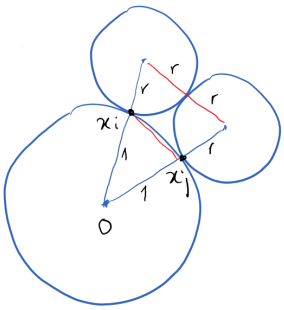
Leo Liberti
CNRS LIX Ecole Polytechnique 91128 Palaiseau, France
e-mail: liberti@lix.polytechnique.fr

ing, said that perhaps thirteen could fit? George Szpiro [20] recounts a different story in `plus.maths.org/content/newton-and-kissing-problem` (some nonsense about astronomy and planets), but since neither he nor I were present at the quabble, his word on the matter is just as good as mine.

1.1 Applications

Quite aside from the satisfaction I get out of spreading academic gossip on Isaac Newton, it turns out that arranging balls in a kissing configuration has applications.

If we only consider the points of contact of n surrounding balls of radius r with the central unit ball, we obtain a set of unit vectors $x_1, \ldots, x_n$ in $\mathbb{R}^K$ that have pairwise distances at least $\frac{2r}{1+r}$. This can be seen in the figure on the left (a two-dimensional section of part of some K-dimensional balls configuration), together with the proportion $2r : (1+r) = d_{ij} : 1$, where $d_{ij} = \|x_i - x_j\|_2$. This is useful if you ever want to send one of the vectors x_i (for $i \leq n$) over a noisy communication channel. You might receive a vector y different from all x_i's, but assuming the channel is not too noisy, you can simply assume that y is a corruption of the closest x_i. This type of error correcting code is called a *spherical code*, and denoted by $A(n,K,r)$. There is interest in maximizing r, since this corresponds to larger balls and consequently more errors being corrected by the code. Kissing number configurations correspond to spherical codes $A(n,K,1)$ where n is maximum for a given K. Finding a spherical code given n,K,r is the SPHERICAL CODE PROBLEM (SCP).



The other (less cited) application is finding n equally spaced points on the K-dimensional sphere S^{K-1}. On websites such as `stackoverflow.com` or `math.stackexchange.com`, it seems people expect this to be an easy problem (possibly because the circle is a simple case: place x_i on the circle at an angle $2i\pi/n$). Some 3D solutions advise scattering equally spaced points on a spiral going from one pole to the opposite, warning readers that it does not guarantee equal spacing. The problem can be formulated for any K by means of spherical codes where the smallest r is maximum. Finding n equally spaced unit vectors on S^{K-1} is the EQUALLY SPACED SPHERICAL POINTS PROBLEM (ESSPP).

1.2 Contents

Most of the results in this paper are known. I propose a new SDP-based heuristic for finding lower bounds, and I give a practitioner's view of Delsarte's Linear Programming (LP) upper bound [7], which is useful for conducting experiments in view of trying to improve current bounds ([2, 14] also discuss practical issues of computing these bounds). All the experiments reported in this paper are preliminary. Lastly, I enjoyed writing this paper more informally than is usual — I hope readers won't object!

2 The Kissing Number Problem

Finding kissing numbers and kissing configurations is known as the KISSING NUMBER PROBLEM (KNP). Formally, this consists in finding the maximum number n of vectors $x_1, \ldots, x_n \in \mathbb{R}^K$ such that $\|x_i\|_2^2 = 1$ for all $i \leq n$ and $x_i \cdot x_j \leq \frac{1}{2}$ for all $i < j \leq n$. If n is the kissing number in K dimensions, we write $\mathsf{kn}(K) = n$. We know that $\mathsf{kn}(2) = 6$, $\mathsf{kn}(3) = 12$ (so Newton was right), $\mathsf{kn}(4) = 24$, and we do not know $\mathsf{kn}(5)$ but it is at least 40. We also know $\mathsf{kn}(8) = 240$ and $\mathsf{kn}(24) = 196560$ [4, p. 510], and have bounds in many other dimensions (see `https://en.wikipedia.org/wiki/Kissing_number_problem`).

2.1 Computational complexity

The computational complexity of solving KNP, as an optimization or even a decision problem, is a prominent and embarassing question mark. Since the input is a pair of integers n, K, mapping an **NP**-complete problem (e.g. SAT) to a KNP, such that an instance of one problem is yes if and only if the corresponding instance of the other problem is also yes, really seems quite hard.

On the other hand, the KNP is certainly very hard to solve empirically, and no-one working on this problem ever suggested that there might be a polynomial-time algorithm for solving it — even on a real RAM computational model.

Reductions between decision versions of KNP, SCP and ESSPP are as follows: the KNP is included in the SCP by definition (so it trivially reduces to the SCP), and, as pointed out in Sect. 3.2, the KNP can be decided by the decision version of the ESSPP (so, again, it reduces to the ESSPP). The decision versions of the SCP and ESSPP are really the same (though the optimization versions differ). The only reduction I cannot immediately prove is from SCP/ESSCP to the KNP, since only the latter fixes the angular separation at $\pi/3$.

I am not aware of *any method* for proving **NP**-hardness of problems whose input consists of a constant number of integers. For example, the complexity status of the well-known PACKING EQUAL CIRCLES IN A SQUARE (PECS) problem is as yet

undetermined [6]. This is certainly an interesting open problem in computational complexity.

A referee pointed out an interesting link with another problem having undetermined complexity status: the PALLET LOADING PROBLEM (PLP), which asks whether n identical $a \times b$ rectangles can be packed in a given $X \times Y$ rectangle [12, §2.C]. Thus, PLP instances, like KNP ones, are described by a constant number of integers. Again, establishing reductions between KNP, SCP, ESSPP, PECS and PLP is an open question.

3 Lower bounds

Since the KNP is a maximization problem, the cardinality of any kissing configuration of balls yields a lower bound. Since any heuristic method might be able to find a good configuration, lower bounds for the KNP are considered "easy" to obtain.

3.1 The formulation of Maculan, Michelon and Smith

We start with the MMS95 formulation proposed in [11], a Mixed-Integer Nonlinear Program (MINLP) which correctly formulates the KNP:

$$\left.\begin{array}{rl} \max\limits_{x\in[-1,1]^{\tau K},\alpha\in\{0,1\}^\tau} & \sum\limits_{i=1}^{\tau} \alpha_i \\ \forall i \leq \tau & \|x_i\|_2^2 = \alpha_i \\ \forall i < j \leq \tau & \|x_i - x_j\|_2^2 \geq \alpha_i \alpha_j, \end{array}\right\} \tag{1}$$

where τ is some (estimated) upper bound to the kissing number. The α_i (binary) variables choose whether vector x_i is part of the configuration or not. Note that the angular separation constraint $x_i \cdot x_j \leq \frac{1}{2}$ is replaced by a Euclidean distance separation $\|x_i - x_j\|_2^2 \geq 1$, which is equivalent since $1 \leq \|x_i - x_j\|_2^2 = \|x_i\|_2^2 + \|x_j\|_2^2 - 2x_i \cdot x_j = 2 - 2x_i \cdot x_j$ (whence $x_i \cdot x_j \leq 1/2$) as the vectors all have unit norm. Since the problem is formulated exactly, an exact MINLP solver would provide a feasible and optimal solution if it exists, given some guessed upper bound.

Unfortunately, the state of the art in MINLP solver technology cannot even provide an answer in $K = 2$ if $\tau = 7$ (one more than $\mathsf{kn}(2) = 6$) in "reasonable time" of a "reasonable laptop" using Eq. (1).

3.2 A feasibility formulation

Given K and n, the formulation below finds the configuration of n unit vectors where the minimal separation between closest vectors is maximum [10]:

$$\left.\begin{array}{rl} \max\limits_{x\in[-1,1]^{nK},\alpha\geq 0} & \alpha \\ \forall i \leq n & \|x_i\|_2^2 = 1 \\ \forall i < j \leq n & \|x_i - x_j\|_2^2 \geq \alpha. \end{array}\right\} \tag{2}$$

Eq. (2) has a single scalar α variable, which is continuous and represents the minimum distance between pairs of vectors. Solving this nonconvex Nonlinear Program (NLP) to global optimality and obtaining an optimal $\alpha \geq 1$ yields a proof that $\mathsf{kn}(K) \geq n$; if $\alpha < 1$ then $\mathsf{kn}(K) < n$. The issue with this strategy for proving kissing numbers is the same as for Eq. (1): current solvers just cannot solve these instances to global optimality for interesting values of n, K.

On the other hand, Eq. (2) can be used heuristically to find KNP configurations given n, K. Moreover, these feasible solutions will in general spread points over the K-sphere quite evenly, each point being at roughly the same distance from its closest points. They will therefore provide a practical solution to the ESSPP.

3.3 Semidefinite Programming relaxations

Semidefinite Programming (SDP) relaxations of Eq. (1)-(2) have not been looked at yet, as far as I know. I performed a few preliminary tests on both, and while the SDP from Eq. (1) seems extremely slack (trivially yielding the upper bound τ for whatever given τ, probably due to relaxed integrality), I found the SDP relaxation of Eq. (2) more interesting:

$$\left.\begin{array}{rl} \max\limits_{X\in[-1,1]^{n^2},\alpha\geq 0} & \alpha \\ \forall i \leq n & X_{ii} = 1 \\ \forall i < j \leq n & X_{ii} + X_{jj} - 2X_{ij} \geq \alpha \\ & X \succeq 0. \end{array}\right\} \tag{3}$$

Based on past experience with other problems involving Euclidean distance constraints, I tried the following heuristic strategy:

1. solve Eq. (3) and obtain an optimal solution $(\bar{X}, \bar{\alpha})$;
2. perform Principal Component Analysis (PCA) using the K largest eigenvalues of $\bar{X}$ to obtain an $n \times K$ matrix $\bar{x}$ such that the i-th row $\bar{x}_i$ is a vector in $\mathbb{R}^K$;
3. use $\bar{x}$ as a starting point for a local NLP solver deployed on Eq. (2), obtain a "good" feasible solution (x^*, α^*).

Using Mosek [15], a Python implementation of PCA, and IPOPT [5], I was able to derive the following KNP configurations on my "resonable laptop" based on a 3.1 GHz Intel Core i7 with 16GB RAM.

(n,K)	$(6,2)$	$(12,3)$	$(24,4)$	$(40,5)$	$(72,6)$
ε	0	0	0.04	0.05	0.07
CPU (s)	0.02	0.02	0.32	1.57	12.26

The first row measures the (additive) solution error with respect to a valid KNP configuration: $\alpha^* = 1 - \varepsilon$ for all $\alpha < 1$ (whereas $\alpha^* > 1$ whenever $\varepsilon = 0$). Unfortunately, the next interesting case, $n = 12$ and $K = 7$, made Mosek crash for lack of RAM (the computational bottleneck on solving SDPs is well known).

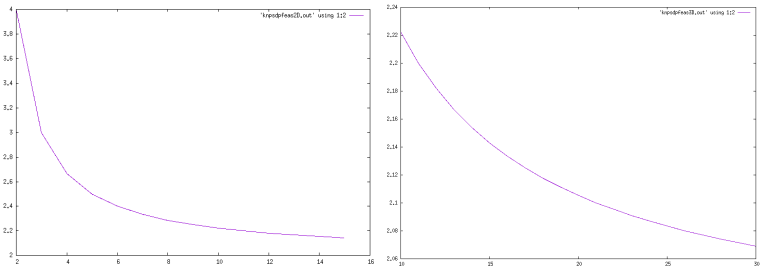
4 Upper bounds

In general, upper bounds to the KNP are considered hard to obtain. Each new upper bound either requires a completely new theoretical point of view, or a substantial body of theoretical work and some computation.

4.1 Direct LP and SDP bounds

A couple of easy upper bounding techniques, however, are readily available in the optimization literature: LP and SDP relaxation. Expanding the Euclidean distances in Eq. (1) and Eq. (2) yields a MINLP and, respectively, a nonconvex NLP involving products of decision variables as the only type of nonlinearity. Using McCormick's [13] for bilinear products and the secant relaxation for square terms one can obtain an LP. For having tested it in the past while I worked on [10], I know that this kind of LP bound is as slack as they come.

Based on some preliminary experiments using the Mosek SDP solver [15] on Eq. (3) for $K \in \{2,3\}$, I observed a regular decrease in the optimal value $\bar{\alpha}$ of the objective function of Eq. (3) as n increases for a given fixed K, as shown in the figures below.



The decreasing sequence of optimal $\bar{\alpha}$ values appears to be converging to 2 for both $K = 2$ and $K = 3$. I have not even started considering how to prove it (or disprove it), nor whether it would have any interesting consequence.

4.2 Delsarte's LP bound

This is an adaptation to the spherical context of Delsarte's upper bound technique for binary codes [8], based on LP.

Broadly speaking, Delsarte's idea is based on deciding the distance distribution of the code so as to maximize its cardinality. Let a_t be the fraction of the vectors in a KNP configuration code $A(n,K,1)$ that have scalar product equal t. Since these codes contain n vectors, there are at most n^2 values of t (there could be fewer if many scalar products have the same value). In any case, summing over the a_t we obtain $n^2/n = n$, which is the cardinality of the code. Also, since these are fractions, $a_t \geq 0$. Moreover, there are exactly n scalar products having value 1, namely $x_i \cdot x_i = \|x\|_2^2 = 1$ for all $i \leq n$, thus $a_1 = n/n = 1$. Hence, the LP

$$\max\{\sum_t a_t \mid a_1 = 1 \wedge a \geq 0\} \qquad (\dagger)$$

is of interest to us, insofar as it maximizes the cardinality of the code it describes. The issue at this point is that this LP is unbounded — there is nothing that links the decision variables a_t to the "code structure". Delsarte's idea consists in finding a family $\mathscr{F} = \{\phi_1, \phi_2, \ldots\}$ of functions $\phi : [-1,1] \to \mathbb{R}$ such that

$$\forall \phi \in \mathscr{F} \sum_t a_t \phi(t) \geq 0 \qquad (\ddagger)$$

is a valid constraint for the LP (†).

Delsarte's "main theorem" has been proved many times, and generalized in various ways [4, §2.2]. Its statement is also valid for the SPC and the ESSPP, as it considers an arbitrary separation angle ζ with $\cos\zeta = z$.

Theorem 1. *Let $c_0 > 0$ and $f : [-1,1] \to \mathbb{R}$ such that:*

(i) $\sum_{i,j \leq n} f(x_i \cdot x_j) \geq 0$
(ii) $\forall t \in [-1,z]\ f(t) + c_0 \leq 0$
(iii) $f(1) + c_0 \leq 1$.

Then $n \leq \frac{1}{c_0}$.

My favorite proof is the one-liner given in [18]. Let $g(t) = f(t) + c_0$, then:

$$n^2 c_0 \leq n^2 c_0 + \sum_{i,j \leq n} f(x_i \cdot x_j) = \sum_{i,j \leq n} g(x_i \cdot x_j) \leq \sum_{i \leq n} g(x_i \cdot x_i) = n g(1) \leq n,$$

whence $n \leq 1/c_0$. This suggests that the problem $\max\{c \mid \text{(i)-(iii)}\}$ $(*)$ is relevant, as higher values for c_0 correspond to tighter bounds. We look for a function f written

as a linear combination of functions $\phi_h \in \mathscr{F}$, and introduce the coefficients c_h so that $f(t) = \sum_h c_h \phi_h(t)$.

We now come to the family $\mathscr{F}$: Delsarte's LP bounding techniques require an orthogonal family of polynomials. In the KNP case, this family consist of *Gegenbauer polynomials* $C_h^{(\lambda)}(t)$ [1, p. 776], that encode certain properties of S^{K-1}. For example, so far our description of Delsarte's LP has failed to include any information about K, which is provided by this choice of $\mathscr{F}$, notably by setting $\lambda = (K-2)/2$ [19, §3]. Another crucial property is that if a function f is a conic combination of Gegenbauer polynomials, then $(f(x_i \cdot x_j)_{ij}) \succeq 0$, whence condition (i) of Thm. 1 holds; moreover, conversely, *any* function satisfying $(f(x_i \cdot x_j)_{ij}) \succeq 0$ can be written as a conic combination of Gegenbauer polynomials [19]. We therefore adjoin the constraints (‡) quantified over $\mathscr{G}^K = \{C_h^{(K-2)/2}(t) \mid h \in H\}$ (for some set H) to the LP (†).

We explicitly write the LP (∗) by requiring that the f appearing in Thm. 1 should be a conic combination of elements of $\mathscr{G}^K$:

$$\max_{c \geq 0} \{c_0 \mid \forall t \; (f(t) = \sum_{\phi_h \in \mathscr{G}^K} c_h \phi_h(t) \; \wedge \; c_0 + f(t) \leq 0) \quad \wedge \quad c_0 + f(1) \leq 1\}. \quad (4)$$

As observed in [17], (†) and (∗) are a pair of dual LPs. You have to "massage" (∗) somewhat before the duality relation with (4) becomes apparent, though (eliminate c_0 and minimize $\sum_h c_h$, see [2, Eq. (5), p. 615]). By duality, we only focus on one LP, namely Eq. (4).

4.2.1 Getting your hands dirty

As stated, Eq. (4) is infinite in both dimensions: $\mathscr{F}$ is countably infinite (if we restrict λ to be integer) and t varies in the uncountably infinite set $\bar{T} = [-1, 1/2] \cup \{1\}$. Only a couple among the many works in the literature mention this as a difficulty (without providing a discussion, however). The only paper I found that provides a satisfactory discussion of this issue is [2].

1. For an upper bound to $\mathsf{kn}(K)$, start with polynomials in $\mathscr{G}^K$ up to degree, say, 15 [17], and gradually work your way up the degrees to see if the bounds improve (be wary of floating point errors arising from evaluating polynomials large degrees — they may invalidate the bound, see [2, Lemma 1]). You can also use any polynomial from $\mathscr{G}^\ell$ for $\ell \geq K$ (though not for $\ell < K$), since Gegenbauer polynomials having lower λ "dominate" — in the sense that they yield larger values of c_0 — those for higher λ. Essentially, the minimum ℓ for which you choose elements in $\mathscr{G}^\ell$ determines the dimensionality of the KNP instance you are going to compute a bound for.
2. A safe way to deal with the fact that $\bar{T}$ is infinite is to choose any finite $T \subseteq \bar{T}$, as removing constraints yields a relaxation (and hence a valid bound). A few experiments will convince you that this discretization has a very small impact on

the value of the bound — moreover, since these are "small LPs", you can still instantly obtain answers even when $|T|$ is pretty large (e.g. $O(10^5)$).
3. In order to obtain closed form expressions for Gegenbauer polynomials, I used Mathematica's `GegenbauerC[h,λ,t]`. However, Eq. (4) expects Gegenbauer polynomials to be normalized so that $C_h^\lambda(1) = 1$, whereas Mathematica's implementation normalizes them differently — you have to remember to evaluate `GegenbauerC[h,λ,t] / GegenbauerC[h,λ,1]`.

I obtained the following (standard) LP upper bounds using Gegenbauer families $C_h^{0.5}$ and C_h^1 respectively, for $h \leq 10$: $\mathsf{kn}(3) \leq \lfloor 13.1583 \rfloor$, $\mathsf{kn}(4) \leq \lfloor 25.5581 \rfloor$ and $\mathsf{kn}(5) \leq \lfloor 46.3365 \rfloor$.

4.3 Extensions

I would like to emphasize three among the extensions to Delsarte's LP bound:

(a) Oleg Musin's extension [16], which brought us the proof that $\mathsf{kn}(4) = 24$;
(b) Pfender's extension [18], yielding new upper bounds for $K \in \{10, 16, 17, 25, 26\}$;
(c) Bachoc and Vallentin's extension [3], which brought us $\mathsf{kn}(5) \leq 45$ (further improved to $\mathsf{kn}(5) \leq \lfloor 44.99899685 \rfloor = 44$ in [14], obtained using the GNU Multiple Precision (GMP) arithmetic library: if you trust the GMP library, you should also trust this bound.

The first two extensions are based on enriching the function family $\mathscr{F}$: in case (a) by including some polynomials f such that condition (ii) of Thm. 1 is violated for a fixed x_i and finitely many x_j (the exceptions that ensue are dealt with combinatorially); and in case (b) by adding a new family of functions to $\mathscr{F}$ that are not convex combinations of Gegenbauer polynomials but satisfy the conditions of Thm. 1. The case (c) is even more interesting as it considers distance distributions on *triplets* of vectors (rather than pairs) and derives a semidefinite programming (SDP) formulation instead of an LP.

See [4] for further extensions.

References

1. M. Abramowitz and I. Stegun, editors. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Number 55 in National Bureau of Standards, Applied Mathematics Series. US Government Printing Office, Washington, 10^t*h* edition, 1972.
2. K. Antreicher. The thirteen spheres: a new proof. *Discrete and Computational Geometry*, 31:613–625, 2004.
3. C. Bachoc and F. Vallentin. New upper bounds for kissing numbers from semidefinite programming. *Journal of the American Mathematical Society*, 21:909–924, 2008.
4. P. Boyvalenkov, S. Dodunekov, and O. Musin. A survey on the kissing numbers. *Serdica Mathematical Journal*, 38:507–522, 2012.

5. COIN-OR. *Introduction to IPOPT: A tutorial for downloading, installing, and using IPOPT*, 2006.
6. A. Costa, P. Hansen, and L. Liberti. On the impact of symmetry-breaking constraints on spatial branch-and-bound for circle packing in a square. *Discrete Applied Mathematics*, 161:96–106, 2013.
7. P. Delsarte, J.M. Goethals, and J.J. Seidel. Spherical codes and designs. *Geometriæ Dedicata*, 6:363–388, 1977.
8. Ph. Delsarte. Bounds for unrestricted codes by linear programming. *Philips Research Reports*, 27:272–289, 1972.
9. G. Dias and L. Liberti. Diagonally dominant programming in distance geometry. In R. Cerulli, S. Fujishige, and R. Mahjoub, editors, *International Symposium in Combinatorial Optimization*, volume 9849 of *LNCS*, pages 225–236, New York, 2016. Springer.
10. S. Kucherenko, P. Belotti, L. Liberti, and N. Maculan. New formulations for the kissing number problem. *Discrete Applied Mathematics*, 155(14):1837–1841, 2007.
11. N. Maculan, P. Michelon, and J. MacGregor Smith. Bounds on the kissing numbers in $\mathbb{R}^n$: Mathematical programming formulations. Technical report, University of Massachusetts, Amherst, USA, 1996.
12. G. Martins. *Packing in two and three dimensions*. PhD thesis, Naval Postgraduate School, 2003.
13. G.P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I — Convex underestimating problems. *Mathematical Programming*, 10:146–175, 1976.
14. H. Mittelmann and F. Vallentin. High-accuracy semidefinite programming bounds for kissing numbers. *Experimental Mathematics*, 19(2):175–179, 2010.
15. Mosek ApS. *The `mosek` manual, Version 8*, 2016. (`www.mosek.com`).
16. O. Musin. The kissing number in four dimensions. *Annals of Mathematics*, 168:1–32, 2008.
17. A. Odlyzko and N. Sloane. New bounds on the number of unit spheres that can touch a unit sphere in *n* dimensions. *Journal of Combinatorial Theory A*, 26:210–214, 1979.
18. F. Pfender. Improved delsarte bounds for spherical codes in small dimensions. *Journal of Combinatorial Theory A*, 114(6):1133–1147, 2007.
19. I. Schoenberg. Positive definite functions on spheres. *Duke Mathematical Journal*, 9(1):96–108, 1942.
20. G. Szpiro. Newton and the kissing problem. *Plus magazine (online)*, 23, January 2003.