Optimality and uniqueness of the D_4 root system

Nando Leijenhorst
Joint works with Henry Cohn, David de Laat and Willem de Muinck Keizer

Delft University of Technology, The Netherlands

October 2024

Contents

- Kissing number problem

D. de Laat, N. M. Leijenhorst, and W. H. H. de Muinck Keizer. *Optimality and uniqueness of the D_4 root system*. arXiv:2404.18794. Apr. 2024.

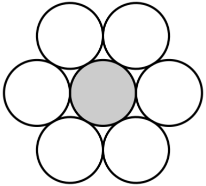
Contents

- Kissing number problem
- Rounding optimal solutions of large semidefinite programs

D. de Laat, N. M. Leijenhorst, and W. H. H. de Muinck Keizer. *Optimality and uniqueness of the D_4 root system*. arXiv:2404.18794. Apr. 2024.

H. Cohn, D. de Laat, and N. Leijenhorst. *Optimality of spherical codes via exact semidefinite programming bounds*. arXiv:2403.16874. Mar. 2024.

The kissing number problem



The kissing number $k(n)$ is the maximum number of unit spheres that simultaneously touch a central unit sphere.

The kissing number problem

- The ‘cannonball packing’ shows that $k(3) \geq 12$, and is rigid.

The kissing number problem

- The 'cannonball packing' shows that $k(3) \geq 12$, and is rigid.
- However, letting spheres touch at the vertices of the regular icosahedron gives a non-rigid configuration of size 12.

The kissing number problem

- The ‘cannonball packing’ shows that $k(3) \geq 12$, and is rigid.
- However, letting spheres touch at the vertices of the regular icosahedron gives a non-rigid configuration of size 12.
- The D_4 root system (vertices of the 24-cell) shows that $k(4) \geq 24$, and is rigid.

The kissing number problem

- The ‘cannonball packing’ shows that $k(3) \geq 12$, and is rigid.
- However, letting spheres touch at the vertices of the regular icosahedron gives a non-rigid configuration of size 12.
- The D_4 root system (vertices of the 24-cell) shows that $k(4) \geq 24$, and is rigid.
- $k(4) = 24$ by O. Musin in 2008

The kissing number problem

- The ‘cannonball packing’ shows that $k(3) \geq 12$, and is rigid.
- However, letting spheres touch at the vertices of the regular icosahedron gives a non-rigid configuration of size 12.
- The D_4 root system (vertices of the 24-cell) shows that $k(4) \geq 24$, and is rigid.
- $k(4) = 24$ by O. Musin in 2008
- Q: can something similar as in dimension 3 happen?

The kissing number problem

- The 'cannonball packing' shows that $k(3) \geq 12$, and is rigid.
- However, letting spheres touch at the vertices of the regular icosahedron gives a non-rigid configuration of size 12.
- The D_4 root system (vertices of the 24-cell) shows that $k(4) \geq 24$, and is rigid.
- $k(4) = 24$ by O. Musin in 2008
- Q: can something similar as in dimension 3 happen?
- A: No, we prove that the D_4 root system is the unique optimal kissing configuration in dimension 4, and is an optimal spherical code.

D. de Laat, N. M. Leijenhorst, and W. H. H. de Muinck Keizer. *Optimality and uniqueness of the D_4 root system*. arXiv:2404.18794. Apr. 2024.

The Lasserre hierarchy for packing problems

The Lasserre hierarchy for the kissing number problem is a sequence of optimization problems

$$\text{las}_1(n) \geq \text{las}_2(n) \geq \ldots \geq \text{las}_{k(n)}(n) = k(n).$$

The Lasserre hierarchy for packing problems

The Lasserre hierarchy for the kissing number problem is a sequence of optimization problems

$$\mathrm{las}_1(n) \geq \mathrm{las}_2(n) \geq \ldots \geq \mathrm{las}_{k(n)}(n) = k(n).$$

Each of these problem can be approximated using sums-of-squares polynomials and semidefinite programming.

The Lasserre hierarchy for packing problems

The Lasserre hierarchy for the kissing number problem is a sequence of optimization problems

$$\mathrm{las}_1(n) \geq \mathrm{las}_2(n) \geq \ldots \geq \mathrm{las}_{k(n)}(n) = k(n).$$

Each of these problem can be approximated using sums-of-squares polynomials and semidefinite programming.

- In dimension $n = 8$ and $n = 24$, $\mathrm{las}_1(n) = k(n)$ (Delsarte LP bound).

The Lasserre hierarchy for packing problems

The Lasserre hierarchy for the kissing number problem is a sequence of optimization problems

$$\mathrm{las}_1(n) \geq \mathrm{las}_2(n) \geq \ldots \geq \mathrm{las}_{k(n)}(n) = k(n).$$

Each of these problem can be approximated using sums-of-squares polynomials and semidefinite programming.

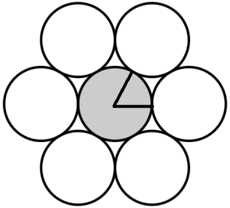
- In dimension $n = 8$ and $n = 24$, $\mathrm{las}_1(n) = k(n)$ (Delsarte LP bound).
- We prove that $\mathrm{las}_2(4) = k(4) = 24$ by giving an exact solution.

The independent set problem

Let $G = (V, E)$ be a graph. A set $I \subseteq V$ is independent if $\{x, y\} \notin E$ for all $x, y \in I$.

The independent set problem

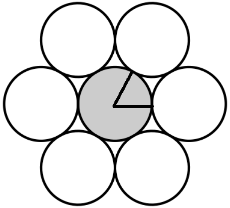
Let $G = (V, E)$ be a graph. A set $I \subseteq V$ is independent if $\{x, y\} \notin E$ for all $x, y \in I$.



Take $V = S^{n-1}$ and distinct $x, y \in V$ adjacent if $x \cdot y > 1/2$.

The independent set problem

Let $G = (V, E)$ be a graph. A set $I \subseteq V$ is independent if $\{x, y\} \notin E$ for all $x, y \in I$.



Take $V = S^{n-1}$ and distinct $x, y \in V$ adjacent if $x \cdot y > 1/2$. Then the kissing number is the maximum size of an independent set in this graph.

The Lasserre hierarchy for packing problems

- $\mathcal{I}_t$: independent sets of size at most t

The Lasserre hierarchy for packing problems

- $\mathcal{I}_t$: independent sets of size at most t
- $C(\mathcal{I}_t \times \mathcal{I}_t)_{\succeq 0}$: the positive definite kernels on $\mathcal{I}_t$.

The Lasserre hierarchy for packing problems

- $\mathcal{I}_t$: independent sets of size at most t
- $C(\mathcal{I}_t \times \mathcal{I}_t)_{\succeq 0}$: the positive definite kernels on $\mathcal{I}_t$.
- $A_t : C(\mathcal{I}_t \times \mathcal{I}_t) \to C(\mathcal{I}_{2t})$ is the operator

$$A_t(K)(S) = \sum_{\substack{J, J' \in \mathcal{I}_t \\ J \cup J' = S}} K(J, J')$$

The Lasserre hierarchy for packing problems

- $\mathcal{I}_t$: independent sets of size at most t
- $C(\mathcal{I}_t \times \mathcal{I}_t)_{\succeq 0}$: the positive definite kernels on $\mathcal{I}_t$.
- $A_t : C(\mathcal{I}_t \times \mathcal{I}_t) \to C(\mathcal{I}_{2t})$ is the operator

$$A_t(K)(S) = \sum_{\substack{J, J' \in \mathcal{I}_t \\ J \cup J' = S}} K(J, J')$$

Then the t-th step of the Lasserre hierarchy is given by:

$$\begin{array}{lll} \text{minimize} & K(\emptyset, \emptyset) & \\ \text{subject to} & A_t K(S) \leq -\chi_{\mathcal{I}_{=1}}(S) & \forall\, S \in \mathcal{I}_{2t} \setminus \{\emptyset\} \\ & K \in C(\mathcal{I}_t \times \mathcal{I}_t)_{\succeq 0} & \end{array}$$

D. de Laat and F. Vallentin. "A semidefinite programming hierarchy for packing problems in discrete geometry". en. In: *Mathematical Programming* 151.2 (July 2015). arXiv: 1311.3789, pp. 529–553. ISSN: 0025-5610, 1436-4646. DOI: 10.1007/s10107-014-0843-4. URL: http://arxiv.org/abs/1311.3789 (visited on 10/26/2021).

Proof that las_t gives an upper bound

$$\begin{array}{lll} \text{minimize} & K(\emptyset,\emptyset) & \\ \text{subject to} & A_t K(S) \leq -\chi_{\mathcal{I}_{=1}}(S) & \forall\, S \in \mathcal{I}_{2t} \setminus \{\emptyset\} \\ & K \in C(\mathcal{I}_t \times \mathcal{I}_t)_{\geq 0} & \end{array}$$

Let $C \subset S^{n-1}$ be a independent set and K a feasible kernel. Then

$$\sum_{\substack{J,J' \subseteq C \\ |J|,|J'| \leq t}} K(J, J')$$

Proof that las_t gives an upper bound

$$\begin{array}{lll} \text{minimize} & K(\emptyset, \emptyset) & \\ \text{subject to} & A_t K(S) \leq -\chi_{\mathcal{I}_{=1}}(S) & \forall\, S \in \mathcal{I}_{2t} \setminus \{\emptyset\} \\ & K \in C(\mathcal{I}_t \times \mathcal{I}_t)_{\succeq 0} & \end{array}$$

Let $C \subset S^{n-1}$ be a independent set and K a feasible kernel. Then

$$0 \leq \sum_{\substack{J, J' \subseteq C \\ |J|, |J'| \leq t}} K(J, J')$$

Proof that las_t gives an upper bound

$$\begin{array}{lll} \text{minimize} & K(\emptyset,\emptyset) & \\ \text{subject to} & A_t K(S) \leq -\chi_{\mathcal{I}_{=1}}(S) & \forall\, S \in \mathcal{I}_{2t} \setminus \{\emptyset\} \\ & K \in C(\mathcal{I}_t \times \mathcal{I}_t)_{\geq 0} & \end{array}$$

Let $C \subset S^{n-1}$ be a independent set and K a feasible kernel. Then

$$0 \leq \sum_{\substack{J,J' \subseteq C \\ |J|,|J'| \leq t}} K(J,J') = \sum_{\substack{S \subseteq C \\ |S| \leq 2t}} A_t K(S)$$

Proof that las_t gives an upper bound

$$\begin{array}{lll} \text{minimize} & K(\emptyset,\emptyset) & \\ \text{subject to} & A_t K(S) \leq -\chi_{\mathcal{I}_{=1}}(S) & \forall\, S \in \mathcal{I}_{2t} \setminus \{\emptyset\} \\ & K \in C(\mathcal{I}_t \times \mathcal{I}_t)_{\succeq 0} & \end{array}$$

Let $C \subset S^{n-1}$ be a independent set and K a feasible kernel. Then

$$0 \leq \sum_{\substack{J,J' \subseteq C \\ |J|,|J'| \leq t}} K(J,J') = \sum_{\substack{S \subseteq C \\ |S| \leq 2t}} A_t K(S) \leq K(\emptyset,\emptyset) - |C|$$

Proof that las_t gives an upper bound

$$\begin{array}{lll} \text{minimize} & K(\emptyset,\emptyset) & \\ \text{subject to} & A_tK(S) \leq -\chi_{\mathcal{I}_{=1}}(S) & \forall\, S \in \mathcal{I}_{2t} \setminus \{\emptyset\} \\ & K \in C(\mathcal{I}_t \times \mathcal{I}_t)_{\succeq 0} & \end{array}$$

Let $C \subset S^{n-1}$ be a independent set and K a feasible kernel. Then

$$0 \leq \sum_{\substack{J,J' \subseteq C \\ |J|,|J'| \leq t}} K(J,J') = \sum_{\substack{S \subseteq C \\ |S| \leq 2t}} A_tK(S) \leq K(\emptyset,\emptyset) - |C|$$

so $K(\emptyset,\emptyset) \geq |C|$.

Complementary slackness

If $K(\varnothing, \varnothing) = |C|$, all inequalities in the proof need to be equalities.

Complementary slackness

If $K(\emptyset, \emptyset) = |C|$, all inequalities in the proof need to be equalities. In particular,

$$A_t(K)(\{x, y\}) = 0 \qquad \forall\, x \neq y \in C.$$

Complementary slackness

If $K(\varnothing,\varnothing) = |C|$, all inequalities in the proof need to be equalities. In particular,

$$A_t(K)(\{x,y\}) = 0 \qquad \forall\, x \neq y \in C.$$

So sharp bounds give information about possible configurations!

Optimality and uniqueness of the D_4 root system

By complementary slackness, our exact optimal solution to $\mathrm{las}_2(4)$ shows that for all sets C of size 24,

$$x \cdot y \in \{-1, -1/2, 0, 1/2\} \quad \text{for all } x \neq y \in C.$$

Optimality and uniqueness of the D_4 root system

By complementary slackness, our exact optimal solution to $\mathrm{las}_2(4)$ shows that for all sets C of size 24,

$$x \cdot y \in \{-1, -1/2, 0, 1/2\} \quad \text{for all } x \neq y \in C.$$

This can be used to show that

- the D_4 root system is the unique optimal kissing configuration in $\mathbb{R}^4$.

Optimality and uniqueness of the D_4 root system

By complementary slackness, our exact optimal solution to $\mathrm{las}_2(4)$ shows that for all sets C of size 24,

$$x \cdot y \in \{-1, -1/2, 0, 1/2\} \quad \text{for all } x \neq y \in C.$$

This can be used to show that

- the D_4 root system is the unique optimal kissing configuration in $\mathbb{R}^4$.
- the D_4 root system is the unique optimal spherical code with 24 points in dimension 4.

Approach for computing an exact solution to las_2

- Write the kernels $K \in C(I_t \times I_t)^{O(n)}_{\succeq 0}$ in terms of positive semidefinite matrices.

Approach for computing an exact solution to las_2

- Write the kernels $K \in C(I_t \times I_t)_{\geq 0}^{O(n)}$ in terms of positive semidefinite matrices.
- Use sum-of-squares polynomials to write the problem as a semidefinite program.

Approach for computing an exact solution to las_2

- Write the kernels $K \in C(I_t \times I_t)^{O(n)}_{\geq 0}$ in terms of positive semidefinite matrices.

- Use sum-of-squares polynomials to write the problem as a semidefinite program.
- Solve the semidefinite program up to high enough precision.

Approach for computing an exact solution to las_2

- Write the kernels $K \in C(I_t \times I_t)_{\geq 0}^{O(n)}$ in terms of positive semidefinite matrices.
- Use sum-of-squares polynomials to write the problem as a semidefinite program.
- Solve the semidefinite program up to high enough precision.
- Round the solution to an exact optimal solution.

Approach for computing an exact solution to las_2

- Write the kernels $K \in C(I_t \times I_t)_{\succeq 0}^{O(n)}$ in terms of positive semidefinite matrices. Time: 2 days
- Use sum-of-squares polynomials to write the problem as a semidefinite program.
- Solve the semidefinite program up to high enough precision.
- Round the solution to an exact optimal solution.

Approach for computing an exact solution to las_2

- Write the kernels $K \in C(I_t \times I_t)_{\geq 0}^{O(n)}$ in terms of positive semidefinite matrices. Time: 2 days
- Use sum-of-squares polynomials to write the problem as a semidefinite program.
- Solve the semidefinite program up to high enough precision. Time: 2 weeks
- Round the solution to an exact optimal solution.

Approach for computing an exact solution to las_2

- Write the kernels $K \in C(I_t \times I_t)_{\geq 0}^{O(n)}$ in terms of positive semidefinite matrices. Time: 2 days
- Use sum-of-squares polynomials to write the problem as a semidefinite program.
- Solve the semidefinite program up to high enough precision. Time: 2 weeks
- Round the solution to an exact optimal solution. Time: 4 hours

Approach for computing an exact solution to las_2

- Write the kernels $K \in C(I_t \times I_t)_{\geq 0}^{O(n)}$ in terms of positive semidefinite matrices. Time: 2 days
- Use sum-of-squares polynomials to write the problem as a semidefinite program.
- Solve the semidefinite program up to high enough precision. Time: 2 weeks
- **Round the solution to an exact optimal solution.** Time: 4 hours

Rounding SDP solutions

What is rounding?

$$\begin{aligned} \text{maximize} \quad & \langle C, X \rangle \\ \text{subject to} \quad & \langle A_i, X \rangle = b_i, \quad i = 1, \ldots, m \\ & X \succeq 0 \end{aligned}$$

What is rounding?

$$\begin{aligned} \text{maximize} \quad & \langle C, X \rangle \\ \text{subject to} \quad & \langle A_i, X \rangle = b_i, \quad i = 1, \ldots, m \\ & X \succeq 0 \end{aligned}$$

A solver gives $X^* \succeq -\varepsilon I$ such that

$$\langle A_i, X \rangle \approx b_i$$

and $\langle C, X^* \rangle$ is approximately optimal. We want an exact optimal solution.

What is rounding?

$$\begin{aligned} \text{maximize} \quad & \langle C, X \rangle \\ \text{subject to} \quad & \langle A_i, X \rangle = b_i, \quad i = 1, \dots, m \\ & X \succeq 0 \end{aligned}$$

A solver gives $X^* \succeq -\varepsilon I$ such that

$$\langle A_i, X \rangle \approx b_i$$

and $\langle C, X^* \rangle$ is approximately optimal. We want an exact optimal solution.
Assumptions:

- Everything can be done over algebraic fields of low degree (in this talk we use $\mathbb{Q}$ for convenience)
- There is a basis of the kernel of an optimal solution X of small bitsize

The rounding procedure

Procedure:

1. Find the kernel vectors

Intuition:

The rounding procedure

Procedure:

1. Find the kernel vectors
2. Do a basis change (facial reduction)

Intuition:

The rounding procedure

Procedure:

1. Find the kernel vectors
2. Do a basis change (facial reduction)

Intuition:

1. We first set the almost 0 eigenvalues to exactly 0.

The rounding procedure

Procedure:

1. Find the kernel vectors
2. Do a basis change (facial reduction)
3. Find an exact solution close to the original solution in the new basis

Intuition:

1. We first set the almost 0 eigenvalues to exactly 0.

The rounding procedure

Procedure:

1. Find the kernel vectors
2. Do a basis change (facial reduction)
3. Find an exact solution close to the original solution in the new basis

Intuition:

1. We first set the almost 0 eigenvalues to exactly 0.
2. Then we modify the remaining eigenvectors and eigenvalues a little bit to satisfy the constraints, so that strictly positive eigenvalues stay strictly positive.

Finding the kernel vectors

We want to find exact kernel vectors using the numerical kernel vectors X^*. Previous approaches used the LLL-algorithm for this ($\mathcal{O}(n^6)$).

Finding the kernel vectors

We want to find exact kernel vectors using the numerical kernel vectors X^*. Previous approaches used the LLL-algorithm for this ($\mathcal{O}(n^6)$).

Let V contain the (numerical) kernel vectors of X^* as rows. Let V_R be the row-reduced echelon form of V. Now round V_R to $\mathbb{Q}$ entry-wise.

Finding the kernel vectors

We want to find exact kernel vectors using the numerical kernel vectors X^*. Previous approaches used the LLL-algorithm for this ($\mathcal{O}(n^6)$).

Let V contain the (numerical) kernel vectors of X^* as rows. Let V_R be the row-reduced echelon form of V. Now round V_R to $\mathbb{Q}$ entry-wise. Since there is a basis of the kernel of low bitsize by assumption, this gives relatively nice kernel vectors.

Transformations

Let the columns of B form a basis of $\mathbb{R}^n$ with the kernel vectors v_i as the first few basis vectors.

Transformations

Let the columns of B form a basis of $\mathbb{R}^n$ with the kernel vectors v_i as the first few basis vectors. Then for any optimal matrix X,

$$B^{\mathsf{T}} X B = \begin{pmatrix} 0 & 0 \\ 0 & \widehat{X} \end{pmatrix}.$$

Transformations

Let the columns of B form a basis of $\mathbb{R}^n$ with the kernel vectors v_i as the first few basis vectors. Then for any optimal matrix X,

$$B^\mathsf{T} X B = \begin{pmatrix} 0 & 0 \\ 0 & \widehat{X} \end{pmatrix}.$$

Let $\widehat{A}_i$ be the block of $B^{-1} A_i B^{-T}$ corresponding to $\widehat{X}$. Then

$$\langle \widehat{A}_i, \widehat{X} \rangle = \langle A_i, X \rangle = b_i.$$

Transformations

Let the columns of B form a basis of $\mathbb{R}^n$ with the kernel vectors v_i as the first few basis vectors. Then for any optimal matrix X,

$$B^{\mathsf{T}} X B = \begin{pmatrix} 0 & 0 \\ 0 & \widehat{X} \end{pmatrix}.$$

Let $\widehat{A}_i$ be the block of $B^{-1} A_i B^{-T}$ corresponding to $\widehat{X}$. Then

$$\langle \widehat{A}_i, \widehat{X} \rangle = \langle A_i, X \rangle = b_i.$$

So the optimal face is described by

$$\hat{F} = \{\widehat{X} \succeq 0 : \langle \widehat{A}_i, \widehat{X} \rangle = b_i, i = 1, \dots, m\}.$$

Getting an exact solution

Write the affine constraint $\langle \widehat{A}_i, \widehat{X} \rangle = b_i$ as $Ax = b$ by vectorizing $\widehat{X}$. Goal: find a solution x close to the numerical solution x^*.

Getting an exact solution

Write the affine constraint $\langle \widehat{A}_i, \widehat{X} \rangle = b_i$ as $Ax = b$ by vectorizing $\widehat{X}$. Goal: find a solution x close to the numerical solution x^*.

Idea: interpolate between the closest solution and a solution modifying few entries of x^*. Use the pseudoinverse (which gives the closest solution) to solve

$$\tilde{A}\tilde{x} = b - Ax^*$$

Getting an exact solution

Write the affine constraint $\langle \widehat{A}_i, \widehat{X} \rangle = b_i$ as $Ax = b$ by vectorizing $\widehat{X}$. Goal: find a solution x close to the numerical solution x^*.

Idea: interpolate between the closest solution and a solution modifying few entries of x^*. Use the pseudoinverse (which gives the closest solution) to solve

$$\tilde{A}\tilde{x} = b - Ax^*$$

where $\tilde{A}$ contains

Getting an exact solution

Write the affine constraint $\langle \widehat{A}_i, \widehat{X} \rangle = b_i$ as $Ax = b$ by vectorizing $\widehat{X}$. Goal: find a solution x close to the numerical solution x^*.

Idea: interpolate between the closest solution and a solution modifying few entries of x^*. Use the pseudoinverse (which gives the closest solution) to solve

$$\tilde{A}\tilde{x} = b - Ax^*$$

where $\tilde{A}$ contains

- a basis of the column space

Getting an exact solution

Write the affine constraint $\langle \widehat{A}_i, \widehat{X} \rangle = b_i$ as $Ax = b$ by vectorizing $\widehat{X}$. Goal: find a solution x close to the numerical solution x^*.

Idea: interpolate between the closest solution and a solution modifying few entries of x^*. Use the pseudoinverse (which gives the closest solution) to solve

$$\tilde{A}\tilde{x} = b - Ax^*$$

where $\tilde{A}$ contains

- a basis of the column space
- a small number of extra columns (say 10% more columns than strictly needed)

Getting an exact solution

Write the affine constraint $\langle \widehat{A}_i, \widehat{X} \rangle = b_i$ as $Ax = b$ by vectorizing $\widehat{X}$. Goal: find a solution x close to the numerical solution x^*.

Idea: interpolate between the closest solution and a solution modifying few entries of x^*. Use the pseudoinverse (which gives the closest solution) to solve

$$\tilde{A}\tilde{x} = b - Ax^*$$

where $\tilde{A}$ contains

- a basis of the column space
- a small number of extra columns (say 10% more columns than strictly needed)

This is both fast and gives solutions of small size.

Questions?

D. de Laat, N. M. Leijenhorst, and W. H. H. de Muinck Keizer. *Optimality and uniqueness of the D_4 root system*. arXiv:2404.18794. Apr. 2024
H. Cohn, D. de Laat, and N. Leijenhorst. *Optimality of spherical codes via exact semidefinite programming bounds*. arXiv:2403.16874. Mar. 2024

The rounding procedure is available as part of the Julia SDP solver `ClusteredLowRankSolver.jl`.