Construction of Dense Lattice Packings in Prime Dimensions

Michael Angel

San Diego State University

15 May 2020

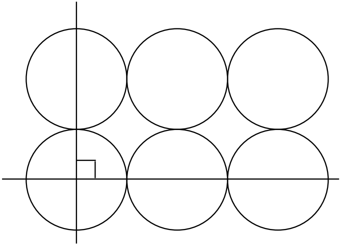
Presentation Agenda

Sphere Packing

Goal:

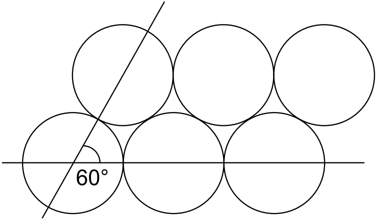
Find arrangements of identical spheres in $\mathbb{R}^n$ with high density.
Metrics: Proportion of total volume or density.

Example: Spheres in 2 dimensions (circles):



less dense (78.54%)

$\delta = .25$



denser (90.69%)

$\delta = .29$

Applications

Various practical applications for dense sphere packings:

- Error correcting codes
- Channel coding with Gaussian noise
- Coding of a Rayleigh fading channel
- Stable state of crystals/quasicrystals

This thesis...

- ... algebraically constructs sphere packings.
- ... proposes novel search technique based on the constructions.
- ... implements search and discovers lattices with high density in dimensions 3, 5, 7, 11 and 13.

Lattice Packings

A **lattice**, that is, a discrete subgroup of $\mathbb{R}^n$, can describe a sphere packing. Lattice points are sphere centers. Points can be generated by generator matrix, $M = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$, where $b_1, \ldots, b_n$ form a basis for $\mathbb{R}^n$.

Example: 2-dimensional Lattice $= \{a_1 b_1 + a_2 b_2 : \forall a_1, a_2 \in \mathbb{Z}\}$

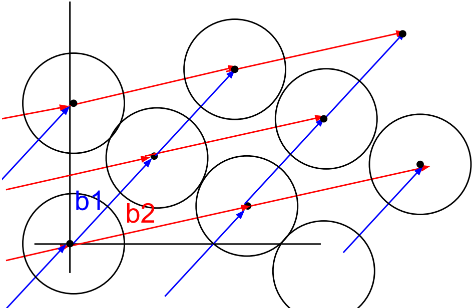


Table of Densest Known Packings:

Dimension	Center Density, δ
3*	0.17678 (lattice)
4	0.12500 (lattice)
5	0.08839 (lattice)
6	0.07217 (lattice)
7	0.06250 (lattice)
8*	0.06250 (lattice)
9	0.04419 (lattice)
10	0.03906 (non-lattice)
11	0.03516 (non-lattice)

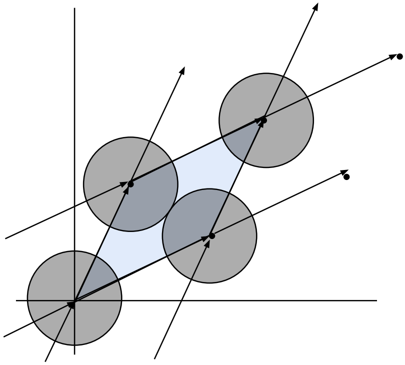
- * Hexagonal packing and Leech lattice packing in dimensions 3 and 8 respectively are proven to be optimally dense.
- Dimensions 4, 5, 6 and 7 cannot have denser lattice packings.

Lattice Fundamental Parallelotope

Given a lattice, Λ, with basis B, the **fundamental parallelotope** of the lattice, $\mathcal{P}(\Lambda)$, is:

$$\{Bx \mid x \in \mathbb{R}^n, \forall i : 0 \leq i < 1\}$$

The fundamental parallelotope contains the volume of one sphere.



Lattice Gram Matrix

Given a lattice with generator matrix M, the **Gram matrix**, G, of the lattice is MM^{tr}.

The **determinant** of the lattice, $\det(\Lambda)$, is the square of volume of the fundamental parallelotope.
$\mathrm{vol}(\Lambda)^2 = \det(\Lambda) = \det(G)$.

- The density of the lattice is the volume of one sphere divided by the volume of the fundamental parallelotope
- Center density, δ, is the volume density divided by the volume of an n-dimensional unit sphere, $\frac{r^n}{\sqrt{\det(G)}}$, were r is the sphere radius

Lenstra–Lenstra–Lovász (LLL) Basis Reduction Algorithm

Center density calculation requires finding the shortest distance between two lattice points ("Shortest Vector Problem", NP-Complete).

The **LLL algorithm** is a relatively simple algorithm that finds a short vector in **polynomial time**. LLL theoretically returns n-dim vector within $2^{(n+1)/2}$ times the actual shortest vector. In practice the algorithm almost always produces shortest vector.

LLL Pseudocode

Input: b = $\{b_1, b_2, \ldots, b_n\}$

Output: reduced b

B := gram_schmidt(b)

k := 1

while k < n do:

  for j in from k−1 to 0 do:

    $u_{k,j} := \frac{\langle b_k, B_j \rangle}{\langle B_k, B_j \rangle}$

    if $|u_{k,j}| > \frac{1}{2}$:

      $b_k = b_k - b_j * u_{k,j}$

      B = gram_schmidt(b)

  if $\langle B_k \rangle \geq (\frac{3}{4} - (u_{k-1,k})^2) * \langle B_{k-1} \rangle$:

    k = k + 1

  else:

    swap b_k and b_{k-1}

    B = gram_schmidt(b)

    k = max(k − 1, 1)

return b

LLL Python Code

```python
def lll_reduction(basis, delta):
    n = len(basis)
    basis = list(map(Vector, basis))
    orthogonal = gram_schmidt(basis)

    def mu(i: int, j: int) -> Rational:
        return orthogonal[j].proj_coeff(basis[i])

    k = 1
    while k < n:
        for j in range(k - 1, -1, -1):
            mu_kj = mu(k, j)
            if abs(mu_kj) > 0.5:
                basis[k] = basis[k] - basis[j] * round(mu_kj)
                orthogonal = gram_schmidt(basis)
        if orthogonal[k].sdot() >= (delta - mu(k, k - 1) ** 2) * orthogonal[k - 1].sdot():
            k += 1
        else:
            basis[k], basis[k - 1] = basis[k - 1], basis[k]
            orthogonal = gram_schmidt(basis)
            k = max(k - 1, 1)
    return basis
```

Quadratic Forms

A **quadratic form** is a polynomial with every term having degree 2.
A quadratic form can be represented with a symmetric matrix, S.

Example:

$$q(x_1, x_2) = 7x_1^2 + 6x_1x_2 + 5x_2^2 = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 7 & 3 \\ 3 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

$$q(x) = xSx^{tr}, \; x = (x_1, x_2)$$

Quadratic Forms $\longleftrightarrow$ Lattices

Quadratic Form, $q \longleftrightarrow$ Symmetric Matrix, S

$$q = xSx^{tr}$$

Symmetric Matrix, $S \longleftrightarrow$ Gram Matrix, G

$$S = G,$$

(positive definite implies lattice is full rank)

Gram Matrix, $G \longleftrightarrow$ Generator Matrix, M

$$G = MM^{tr}$$

Generator Matrix, $M \longleftrightarrow$ Lattice, Λ

$$\Lambda = \{aM : \forall a \in \mathbb{Z}^n\}$$

Number Field Definitions and Theorems

- **Number field**: a finite degree field extension of the field of rational numbers.
- The **ring of integers**, $\mathcal{O}_K$, of a number field, K, is the set of all elements in K that are roots of monic polynomials with integer coefficients.
- The ring of integers is a finitely generated $\mathbb{Z}$-module and thus has an **integral basis**, $b_1, \ldots, b_n$, where n is the degree of field extension $K/\mathbb{Q}$.
- A field extension is **abelian/cyclic** if its Galois group is abelian/cyclic.

Kronecker–Weber Theorem and Conductor

- **Kronecker–Weber Theorem**: K finite abelian number field $\Rightarrow K \subset \mathbb{Q}(\zeta_n)$, for some n, where ζ_n is an n-th root of unity.
- **Conductor**, f, of K is the smallest n such that $K \subset \mathbb{Q}(\zeta_n)$
- When $K/\mathbb{Q}$ is unramified and has degree p, for prime p, the conductor f is of the form $\prod_{i=1} p_i$ for distinct primes p_i, $p_i \equiv 1 \pmod{p}$.
- When f is prime itself, f is the smallest prime such that $f \equiv 1 \pmod{p}$ and $K \subset \mathbb{Q}(\zeta_f)$.

Restrictions on Field K

Field K is a cyclic number field with degree p, where p is an odd, unramified prime in $K/\mathbb{Q}$. Then:

- $K \subset \mathbb{Q}(\zeta_n)$ (Kronecker–Weber Theorem);
- K is totally real,
  i.e., all of K's embeddings into $\mathbb{C}$ are real;
- Field Discriminant of K, $\mathrm{disc}(K)$, is f^{p-1}.

Trace Form of Number Field

For number field K and $\alpha \in K$ and let $\sigma_1(\alpha), \sigma_2(\alpha), \ldots, \sigma_n(\alpha)$ be the roots of the minimal polynomial of α over $\mathbb{Q}$. The **field trace**, $\mathrm{Tr}_{K/\mathbb{Q}}(\alpha)$, is defined as:

$$\mathrm{Tr}_{K/\mathbb{Q}}(\alpha) = \sum_{i=1}^{n} \sigma_i(\alpha).$$

Note: $\mathrm{Tr}_{K/\mathbb{Q}}(\alpha)$ is always a rational number and is an integer when α is an algebraic integer.

We define the **trace form** as the map from $K \times K$ to $\mathbb{Q}$ sending (x, y) to $\mathrm{Tr}_{K/\mathbb{Q}}(xy)$. The trace form is a quadratic form.

Canonical Embedding (a.k.a. the Minkowski embedding)

Number field K; degree n; $\sigma_1, \sigma_2, \ldots, \sigma_n$ are n distinct embeddings into $\mathbb{C}$.
$\sigma_1, \sigma_2, \ldots, \sigma_{s_1}$: real
$\sigma_{s_1+1}, \ldots, \sigma_n$: complex
$n - s_1 = 2s_2$ complex embeddings paired such that $\sigma_{s_1+i} = \overline{\sigma_{s_1+s_2+i}}$ for $0 \leq i \leq s_2$. The **canonical embedding**, σ_K, of K to $\mathbb{R}^n$ is:

$$\sigma_K(x) = \Big(\sigma_1(x), \sigma_2(x), \ldots, \sigma_{s_1}(x),$$

$$\mathrm{Re}(\sigma_{s_1+1}(x)), \mathrm{Im}(\sigma_{s_1+1}(x)), \ldots, \mathrm{Re}(\sigma_{s_1+s_2}(x)), \mathrm{Im}(\sigma_{s_1+s_2}(x))\Big).$$

If $\mathcal{M}$ is a $\mathbb{Z}$-submodule of $\mathcal{O}_K$ of full rank, then $\sigma_K(\mathcal{M})$ is a full lattice (lattice of dimension n).

Canonical Embedding (a.k.a. the Minkowski embedding)

Call K's embeddings $\sigma_1, \sigma_2, \ldots, \sigma_n$ (all real).
Let $b_1, \ldots, b_n$ be a basis for $\mathcal{O}_K$.
$\sigma_K(\mathcal{O}_K)$ has generator matrix:

$$M = \begin{bmatrix} \sigma_1(b_1) & \sigma_2(b_1) & \ldots & \sigma_n(b_1) \\ \sigma_1(b_2) & \sigma_2(b_2) & \ldots & \\ \vdots & & & \vdots \\ \sigma_1(b_n) & \ldots & & \sigma_n(b_n) \end{bmatrix}$$

The Gram matrix, $G = MM^{tr}$, has (i, j) entry $\mathrm{Tr}_{K/\mathbb{Q}}(b_i b_j)$, and thus the Gram matrix of $\sigma_K(\mathcal{O}_K)$ is the symmetric matrix of the trace form of K, $\mathrm{Tr}_{K/\mathbb{Q}}(x^2)$, for $x \in \mathcal{O}_K$.

Center Density of $\sigma_K(\mathcal{O}_K)$

$$\delta(\sigma_K(\mathcal{O}_K)) = \frac{d^p}{2^p \mathrm{vol}(\sigma_K(\mathcal{O}_K))} = \frac{d^p}{2^p \sqrt{|\mathrm{disc}(K)|}} = \frac{d^p}{2^p f^{\frac{p-1}{2}}}$$

G : Gram matrix of $\sigma_K(\mathcal{O}_K)$

d : shortest distance between lattice points in $\sigma_K(\mathcal{O}_K)$

Trace Form

K: Cyclic number field with degree p, an odd, unramified prime in $K/\mathbb{Q}$.
f: Conductor of K
$L = \mathbb{Q}(\zeta_f)$
Let θ be a generator of $\mathrm{Gal}(K/\mathbb{Q})$, and $t = \mathrm{Tr}_{L/K}(\zeta_f)$,
K has integral basis $\{t, \theta(t), \cdots, \theta^{p-1}(t)\}$,
Let $x \in \mathcal{O}_K$, $\quad x = \sum_{i=0}^{p-1} a_i\theta^i(t)$, then:

$$\mathrm{Tr}_{K/\mathbb{Q}}(x^2)|_{\mathcal{O}_K} = f \cdot \left(\sum_{i=0}^{p-1} a_i^2\right) - \frac{f-1}{p}\left(\sum_{i=0}^{p-1} a_i\right)^2$$

[E. L. d. Oliveira, J. C. Interlando, T. P. da Nóbrega Neto, and J. O. D. Lopes, *The integral trace form of cyclic extensions of odd prime degree*, Rocky Mountain Journal of Mathematics, 47 (2017), pp. 1075–1088.]

Submodules Defined By Linear Transformation Matrix, T

T : Matrix with rank p (full rank) and p-columns
$\mathcal{M}$: Submodule of $\mathcal{O}_K$ characterized by T
d: Minimum distance between lattice points

Lattice	Gram Matrix	Center Density (δ)
$\sigma_K(\mathcal{O}_K)$	$G = \text{SymmMat}(\text{Tr}_{K/\mathbb{Q}}(x^2)\vert_{\mathcal{O}_K})$	$\delta(\sigma_K(\mathcal{O}_K)) = \frac{d^p}{2^p f^{\frac{p-1}{2}}}$
$\sigma_K(\mathcal{M})$	TGT^{tr}	$\delta(\sigma_K(\mathcal{M})) = \frac{d^p}{2^p f^{\frac{p-1}{2}}[\mathcal{O}_K:\mathcal{M}]}$

Definition of $\mathcal{M}_H$

$$H := \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & \zeta & \zeta^2 & \zeta^3 & \dots & \zeta^{(p-1)} \\ 1 & \zeta^2 & \zeta^4 & \zeta^6 & \dots & \zeta^{2(p-1)} \\ \vdots & & & & & \\ 1 & \zeta^{\frac{(p-1)}{2}} & \zeta^{2\frac{(p-1)}{2}} & \zeta^{3\frac{(p-1)}{2}} & \dots & \zeta^{(p-1)\frac{(p-1)}{2}} \end{bmatrix}$$

Given $(a_0, ..., a_{p-1}) \in \mathbb{Z}^n$,
$\{t, \theta(t), \cdots, \theta^{p-1}(t)\}$: integral basis of $\mathcal{O}_K$,
submodule $\mathcal{M} \subset \mathcal{O}_K$ is defined as:

$$\mathcal{M}_H = \{a_0 t + a_1\theta(t) + \cdots + a_{p-1}\theta^{p-1}(t) \in \mathcal{O}_K : \\ (a_0, \dots, a_{p-1})H^{tr} \equiv (0, \dots, 0) \pmod{f}\}.$$

The rows of H represent congrunces mod f.

Transformation T defined by H

The rows of H represent congrunces mod f.

$T :=$ first p columns of kernel(t), where $t := \left[\begin{array}{c|c} H & f \cdot Id \end{array} \right]$

TGT^{tr} is the Gram matrix of $\sigma_K(\mathcal{M}_H)$.

Why $\sigma_K(\mathcal{M}_H)$?

$[\mathcal{O}_K : \mathcal{M}_H] = f^{\frac{p+1}{2}}$.
$\Rightarrow \delta(\sigma_K(\mathcal{M}_H)) = \frac{d^p}{2^p f^{\frac{p-1}{2}} [\mathcal{O}_K : \mathcal{M}_H]} = \frac{d^p}{2^p f^p}$.

$K/\mathbb{Q}$ totally real $\Rightarrow$
$d = \sqrt{\min(\mathrm{Tr}_{K/\mathbb{Q}}(x^2))}$

This thesis proves that for $\sigma_K(\mathcal{M}_H)$, f divides d
$\Rightarrow d = f\sqrt{d_2}$, for some integer d_2.

$\delta(\sigma_K(\mathcal{M}_H)) = \frac{\sqrt{d_2}^p}{2^p}$

Search Strategy

H has $\frac{p+1}{2}$ rows of congruences mod f.
Consider additional congruences. A congruence mod m, will add a factor of m to the index of the submodule.

Let $\mathcal{M}$ be a submodule defined by the congruences of H as well as additional congruences. Let i be the product of additional index factors from additional congruences and we have:

$\delta(\mathcal{M}) = \frac{\sqrt{d_2}^p}{2^p i}$

Overview of Search

$\delta = \frac{\sqrt{d_2}^p}{2^p i}$

1. Fix δ (target density).
2. Determine an i that yields the target density.
3. Test submodules that have additional index factor i

Example, Dimension 5

$p = 5$
Fix $\delta = \frac{1}{8\sqrt{2}}$, the highest possible density for dimension 5.
$\delta = \frac{1}{8\sqrt{2}} = \frac{\sqrt{d_2}^5}{2^5 i} \Rightarrow d^5 = 2^3 i^2$

$i = 2^1, d = 2^1$ satisfies the equality.

Index factor i has one factor of 2, therefore look for a lattice with center density $\frac{1}{8\sqrt{2}}$ by considering submodules constructed with the congruences of H as well as one additional congruence modulo 2: $\mathrm{J} = \begin{bmatrix} X & X & X & X & X \end{bmatrix}$, where $X = 0$ or 1.

Use matrices H and J to construct a lattice, for all possible J's. Calculate the density of all 2^5 lattices.

Example, Dimension 7

$p = 7$
$\delta = \frac{1}{2^4} = \frac{\sqrt{d}^7}{2^7 i} \Rightarrow d^7 = 2^6 i^2$

$i = 2^4$ satisfies the equality.

Index factor i has 4 factors of 2, therefore look for a lattice with center density $\frac{1}{2^4}$ by considering 4 additional congruences modulo 2:

$$\mathrm{J} = \begin{bmatrix} X & X & X & X & X & X & X \\ X & X & X & X & X & X & X \\ X & X & X & X & X & X & X \\ X & X & X & X & X & X & X \end{bmatrix}, \text{ where } X = 0 \text{ or } 1.$$

Size of the search space is 2^{28} lattices.

Search Implementation: Algorithm

```
Input:  p, f, m, search_size
Output:  search_size many densities, d

zeta := find_primitive_root(p, f)
h := make_h(zeta, p)
h = m * h
g := symmetric_matrix_of_trace_form(p)
i := 0
while i < search_size:
   j := make_search_matrix(i)
   j = f * j
   t := h.concatenate(j)
   t = augment_identity_times_factor(t, f*m)
   n := get_nullspace(t)
   n = n.matrix_from_columns(p)
   gram_matrix := n*g*n.transpose()
   d := get_density(gram_matrix)
   print(d)
   i = i + 1
```

github.com/mike006322/LatticePackings

```
def main():
    name = 'search_dim_5'
    data_filename = 'search_data/' + name + '.txt'
    p = 5  # prime dimension
    f = 11  # conductor
    m = 2
    zeta = find_primitive_root(p, f)
    h = m * make_h(zeta, p)
    # g is symmetric matrix of trace form
    g = matrix(DIM_5_TR_SYM_MATRIX)
    for i in range(2 ** 5):
        j = f * make_search_matrix(i)
        t = h.concatenate(j)
        t = augment_identity_times_factor(t, f*m)
        t = matrix(t)  # make t a SAGE matrix
        n = get_nullspace(t)
        n = n.matrix_from_columns(range(p))  # SAGE specific.
        gram_matrix = n * g * n.transpose()
        d = get_density(gram_matrix)
        add_to_file(str(d), data_filename)
```

Search Implementation

Mathematical Computation Requirements:

- Number Class (Floating point or infinite precision)
- Matrix Class (w/ Concatenation, Transpose, etc)
- Matrix Multiplication
- Get Matrix Nullspace
- Lattice Class
- Get Lattice Density
- Basis Reduction Algorithm (LLL)

Search Implementation

LLL Basis Reduction

- LLL original implemented in Python (slow)
- SAGE LLL using floating point algorithm, fplll, open source, C++ package at github.com/fplll/fplll
- MAGMA using floating point Nguyen and Stehlé LLL implementation

Higher Dimensions: Exponential Growth in Search Size

- Search size for $\delta = \frac{1}{8\sqrt{2}}$ in dimension 5: 2^5 lattices
- Search size for $\delta = \frac{1}{16}$ in dimension 7: 2^{21} lattices
- Search size for $\delta = \frac{1}{32}$ in dimension 11: 2^{24} lattices
- Search size for $\delta = \frac{1}{32}$ in dimension 13: 2^{32} lattices

Implementation For Non-binary Search Matrix Entries

Example: Search for $\delta = \frac{1}{18\sqrt{3}}$ in 11 Dimensions:

$i = 2 \cdot 3^8$

$X = 0$ or 1, $Y = 0, 1, 2$

Search Matrix, J:

$$J = \begin{bmatrix} X & X & X & X & X & X & X & X & X & X & X \\ Y & Y & Y & Y & Y & Y & Y & Y & Y & Y & Y \\ Y & Y & Y & Y & Y & Y & Y & Y & Y & Y & Y \\ Y & Y & Y & Y & Y & Y & Y & Y & Y & Y & Y \\ Y & Y & Y & Y & Y & Y & Y & Y & Y & Y & Y \\ Y & Y & Y & Y & Y & Y & Y & Y & Y & Y & Y \\ Y & Y & Y & Y & Y & Y & Y & Y & Y & Y & Y \\ Y & Y & Y & Y & Y & Y & Y & Y & Y & Y & Y \\ Y & Y & Y & Y & Y & Y & Y & Y & Y & Y & Y \end{bmatrix}$$

Maintain explicit order for progress tracking, multi-processing

Utilize Equivalence Classes

Search matrices J_1 and J_2 yield lattices with same density if they have the same rows or columns but permuted.

Accounting for this, we can reduce the size of the search space by a factorial factor.