Bachelor thesis - Lattice Basis Reduction

L.M. (Leanne) Scheepers

Supervisor: Benne de Weger
Faculteit Wiskunde en Informatica
Technische Universiteit Eindhoven

December 1, 2011
# Contents

1 Introduction 5

2 Lecture notes 7

2.1 Lattice basis reduction, n-dimensional 7

2.1.1 Example with the algorithm computed on the columns 8

2.1.2 Mathematica Function 10

2.1.3 Interesting properties 11

2.2 Interesting applications 11

2.2.1 Computing $|a\pi + bc + c\log 2| \approx \text{small}$ 11

2.2.2 Finding roots of a function 12

2.2.3 Diophantine Approximation 13

2.2.4 Simultaneous Diophantine Approximation 14

2.2.5 Inhomogeneous Diophantine approximation 17

2.2.6 Minimization of a convex function over $\mathbb{Z}^2$ 19

2.3 Exercises 22

2.4 Answers 23
Chapter 1

Introduction

For the course Discrete Mathematics (2WC15) in our current Bachelor program, there are lecture notes ([1]) in which different subjects are treated. As an extension Benne de Weger already wrote a part about continued fractions, diophantine approximation and lattices. In the latter part some basics of lattices are explained and the concept of lattice basis reduction in the 2-dimensional case. This could and should be extended to lecture notes with the n-dimensional case included as well.

Therefore the goal of my bachelor thesis was writing this extension of the lecture notes about lattice basis reduction in the n-dimensional case.

In the next chapter this extension can be found, which is not only a description of the n-dimensional case, but also treats some interesting applications.

General references for that chapter are: [3, Chapters 16 and 17] and [4, Chapter 2].
Chapter 2

Lecture notes

2.1 Lattice basis reduction, $n$-dimensional

The goal of lattice basis reduction is to change a given lattice basis (without changing the lattice) in order to improve the basis. A given basis can be improved if the basis vectors are shortened and the vectors are made more orthogonal.

The following definition tells us when a basis is LLL-reduced.

**Definition 2.1** Let $(b_1, b_2, \ldots, b_n)$ be an ordered basis for a lattice. Denote by $(b^*_1, b^*_2, \ldots, b^*_n)$ the Gram-Schmidt orthogonalisation and write $B_i = \|b^*_i\|^2 = \langle b^*_i, b^*_i \rangle$. Let $\mu_{i,j} = \frac{\langle b_i, b^*_j \rangle}{\langle b^*_j, b^*_j \rangle}$ for $1 \leq j < i \leq n$ be the coefficients from the Gram-Schmidt process. Fix $\frac{1}{4} < \delta < 1$. The (ordered) basis is LLL-reduced (with factor $\delta$) if the following conditions hold:

- **(Size reduced)** $|\mu_{i,j}| \leq \frac{1}{2}$ for $1 \leq j < i \leq n$
- **(Lovász condition)** $B_i \geq (\delta - \mu_{i,i}^2)B_{i-1}$ for $2 \leq i \leq n$

It is traditional to choose $\delta = \frac{3}{4}$

As seen above it is important to know what the Gram-Schmidt orthogonalisation is. Some of its properties are:

**Lemma 2.2** Let $(b_1, b_2, \ldots, b_n)$ be linearly independent in $\mathbb{R}^n$ and let $(b^*_1, b^*_2, \ldots, b^*_n)$ be the Gram-Schmidt orthogonalisation.

1) $\|b^*_i\| \leq \|b_i\|$ for $1 \leq i \leq n$

2) $\langle b_i, b^*_i \rangle = \langle b^*_i, b^*_i \rangle$ for $1 \leq i \leq n$

3) Denote the closest integer to $\mu_{k,j}$ by $[\mu_{k,j}]$; i.e. $[\frac{1}{2}] = 1$ and $[-\frac{1}{2}] = -1$. If $b'_k = b_k - [\mu_{k,j}]b_j$ for $1 \leq k \leq n$ and $1 \leq j < k$ and if $\mu'_{k,j} = \frac{\langle b'_k, b^*_j \rangle}{\langle b^*_j, b^*_j \rangle}$ then $|\mu'_{k,j}| \leq \frac{1}{2}$.

**Note:** The $b^*_i$ are orthogonal to each other; the $b_i$ not necessarily. Furthermore the $b_i$ are in the lattice, the $b^*_i$ usually not.
And a way to compute the Gram-Schmidt orthogonalisation:

**Algorithm 2.1 (Gram-Schmidt algorithm)**

**Input:** a basis \( \{b_1, b_2, \ldots, b_n\} \)

**Output:** \( \{b_1^*, b_2^*, \ldots, b_n^*\} \); the Gram-Schmidt orthogonalisation of \( \{b_1, b_2, \ldots, b_n\} \)

**Step 1:** \( b_1^* = b_1 \)

**Step 2:** for \( i \) from 2 to \( n \) do

\[ v = b_i \]

for \( j := i - 1 \) downto 1 do

\[ \mu_{i,j} = \frac{\langle b_i, b_j^* \rangle}{\langle b_j^*, b_j^* \rangle} \]

\[ b_i^* = v - \mu_{i,j} b_j^* \]

end for

end for

**Step 3:** output \( \{b_1^*, b_2^*, \ldots, b_n^*\} \)

**Algorithm 2.2 (LLL algorithm with Euclidean norm)**

**Input:** a basis \( \{b_1, b_2, \ldots, b_n\} \in \mathbb{Z}^n \)

**Output:** LLL reduced basis \( \{b_1, b_2, \ldots, b_n\} \)

**Step 1:** Compute the Gram-Schmidt basis \( b_1^*, \ldots, b_n^* \) and coefficients \( \mu_{i,j} \) for \( 1 \leq j < i \leq n \)

Compute \( B_i = \|b_i^*\|^2 = \langle b_i^*, b_i^* \rangle \) for \( 1 \leq i \leq n \)

\( k = 2 \)

**Step 2:** while \( k \leq n \) do

for \( j = (k-1) \) downto 1 do

Let \( q_k = \lfloor \mu_{k,j} \rfloor \) and set \( b_k = b_k - q_j b_j \)

Update the values \( \mu_{k,j} \) for \( 1 \leq j < k \)

end for

if \( B_k \geq (\delta - \mu_{k,k-1}^2) B_{k-1} \) then

\( k = k + 1 \)

else

Swap \( b_k \) with \( b_{k-1} \),

Update the values \( B_k, B_{k-1}, \mu_{k-1,j} \) and \( \mu_{k,j} \) for \( 1 \leq j < k \) and \( \mu_{i,k}, \mu_{i,k-1} \) for \( k < i \leq n \)

\( k = \min\{2, k-1\} \)

end if

end while

### 2.1.1 Example with the algorithm computed on the columns

Consider the matrix \( L_0 \) and the trivial transition matrix \( T = T_0 \):

\[
L_0 = \begin{pmatrix}
1 & -1 & 3 \\
1 & 0 & 5 \\
1 & 2 & 6 \\
\end{pmatrix}, \quad T_0 = \begin{pmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\end{pmatrix}
\]

On \( L_0 \) we will use Algorithm 2.2. When applying the algorithm we start with step 1; computing the Gram-Schmidt basis \( b_1^*, \ldots, b_n^* \) and coefficients \( \mu_{i,j} \) for \( 1 \leq j < i \leq n \) and computing the \( B_i = \|b_i^*\|^2 = \langle b_i^*, b_i^* \rangle \) for \( 1 \leq i \leq n \).

Then \( b_1^* = b_1 \) and \( B_1 = \|b_1^*\|^2 = \langle b_1^*, b_1^* \rangle = 3 \)

\[ \mu_{2,1} = \frac{\langle b_2, b_1^* \rangle}{B_1} = \frac{1}{3}, \text{ so } b_2^* = b_2 - \mu_{2,1} b_1^* = b_2 - \frac{1}{3} b_1^* = \left( -\frac{1}{3}, \frac{2}{3} \right) \]

and \( B_2 = \frac{42}{9} = \frac{14}{3} \)
2.1 Lattice basis reduction, \( n \)-dimensional

\[ \mu_{3,1} = \left\langle \frac{b_3, b_1^*}{B_1} \right\rangle = \frac{14}{3}, \] so \( b_3^* = b_3 - \mu_{3,1} b_1^* = b_3 - \frac{14}{3} b_1^* = \left( \begin{array}{c} -\frac{1}{3} \\ \frac{1}{3} \end{array} \right) \) and \( \mu_{3,2} = \frac{\langle b_3, b_2^* \rangle}{B_2} = \frac{13}{14}, \) so

\[ b_3^* = b_3 - \mu_{3,2} b_2^* = b_3 - \frac{13}{14} b_2^* = \left( \begin{array}{c} \frac{6}{14} \\ -\frac{13}{14} \end{array} \right) \) and \( B_3 = \frac{9}{14}, \)

\( k = 2 \)

\( q_1 = [\mu_{2,1}] = 0 \Rightarrow b_2 = b_2 - q_1 b_1 = b_2. \) As \( b_2 \) stays the same, we don’t need to update any values.

Then we arrive at the if-statement, so we are checking if: \( B_2 \geq (\delta - \mu_{2,1}) B_1. \)

\[ B_2 = \frac{14}{3} = \frac{56}{12} \]

\( (\delta - \mu_{2,1}) B_1 = (\frac{3}{4} - (\frac{1}{3})^2)3 = \frac{69}{36} = \frac{12}{12} \)

Whereas \( B_2 \) satisfies the condition, \( k = k + 1 = 3 \)

\( k = 3 \)

\( q_2 = [\mu_{3,2}] = 1 \Rightarrow b_3 = b_3 - q_2 b_3 = \left( \begin{array}{c} 4 \\ 5 \\ 4 \end{array} \right) \)

Now we update the values \( \mu_{3,1} \) and \( \mu_{3,2}: \)

\[ \mu_{3,1} = \frac{\langle b_3, b_1^* \rangle}{B_1} = \frac{13}{3}, \]

\[ \mu_{3,2} = \frac{\langle b_3, b_1^* \rangle}{B_1} = \frac{1}{14}, \]

\[ B_3 = \frac{9}{14} < (\frac{3}{4} - \mu_{3,2}) B_2 \Rightarrow \text{swap } b_3 \text{ with } b_2. \]

This results in the following matrix:

\[ L_1 = \left( \begin{array}{ccc} 1 & 4 & -1 \\ 1 & 5 & 0 \\ 1 & 4 & 2 \end{array} \right) \]

With transition matrix \( T_1 \) so \( L_1 = L_0 T_1 \)

\[ T_1 = \left( \begin{array}{ccc} 1 & 0 & 0 \\ 0 & -1 & 1 \\ 0 & 1 & 0 \end{array} \right) \]

We call \( T \) the total transition matrix; i.e. \( L_0 T = L_1, \) and here \( T := T_0 T_1 = T_1. \)

\( k=2 \)

Now \( \mu_{2,1} = \frac{12}{3} \) and \( [\mu_{2,1}] = 4. \) Then \( b_2 = b_2 - 4b_1 = \left( \begin{array}{c} 0 \\ 1 \\ 0 \end{array} \right) \)

\[ B_2 = \frac{2}{3} < (\frac{3}{4} - \frac{1}{2})3 \Rightarrow \text{swap } b_2 \text{ and } b_1. \]

This results in the following matrix:

\[ L_2 = \left( \begin{array}{ccc} 0 & 1 & -1 \\ 1 & 1 & 0 \\ 0 & 1 & 2 \end{array} \right) \]
With transition matrix $T_2$ so $L_2 = L_1 T_2$

$$T_2 = \begin{pmatrix} -4 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

And the total transition matrix $T := T_2^{-1} = \begin{pmatrix} -4 & 1 & 0 \\ -1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$. It is easy to compute the "total" transition matrix $T_1 T_2$; apply to $T_1$ the same transformation as done to $L_1$.

$k=2$

Now $\mu_{2,1} = \frac{1}{4} = 1$ and $|\mu_{2,1}| = 1$. Then $b_2 = b_2 - b_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

$B_2 = 2 > \left( \frac{3}{4} - 1 \right) l \Rightarrow k = k + 1 = 3$ and we now have the following matrix:

$$L_3 = \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & 2 \end{pmatrix}$$

With transition matrix $T_3$ so $L_4 = L_2 T_3$

$$T_3 = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

And total transition matrix $T := T_3^{-1} = \begin{pmatrix} -4 & 5 & 0 \\ -1 & 1 & 1 \\ 1 & -1 & 0 \end{pmatrix}$

$k=3$

$\mu_{3,1} = 0$

$\mu_{3,2} = 0$

So our algorithm ends.

The LLL-reduced matrix:

$$L_{red} = \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & 2 \end{pmatrix}$$

And $L_{red} = L_0 T_0 T_1 T_2 T_3 = L_0 T$. As noted above it is quite easy to compute the "total" transition matrix. Therefore it is ‘cheaper’ to keep track of the "total" transition matrix during the computation, than computing an inverse of $L_0$ and computing $T = \text{Inverse}[L_0] L_{red}$.

2.1.2 Mathematica Function

As stated above: computing an LLL-reduced basis by hand is quite an operation. Luckily there is a function in Mathematica: \text{LatticeReduce[]}, that can compute an LLL-reduced basis for us. This command works on the rows of a matrix. A downside of the command is that it can only work with integers. If you want to use the command on the columns of a matrix $m$, just use: \text{Transpose[LatticeReduce[Transpose[m]]]}.

\textbf{Note:} Computations in Mathematica might give different reduced bases, because the implementation in Mathematica might be slightly different from the algorithm used here.
2.2 Interesting applications

2.1.3 Interesting properties

There are some interesting properties of an LLL-reduced basis \((b_1, \ldots, b_n)\).

- Let \(\alpha = \frac{1}{\sqrt{2}}\). Then \(\|b_1\| \leq \alpha \frac{\sqrt{n}}{n-1} (\det L)^{\frac{1}{2}}\). So we have an upper bound for the length of the first reduced vector of our basis. Furthermore \(\|b_1\| \times \ldots \times \|b_n\| \leq \alpha ^{\frac{n(n-1)}{2}} (\det L)\).

- The determinant, or volume of the generator cell, is preserved up to sign; \(|\det(A)| = |\det(B)|\).

- If we look at the previous example and in particular at the transition matrices, it is obvious that \(\det(T_0) = \det(T_3) = 1\) and \(\det(T_1) = \det(T_2) = -1\). Whereas \(L_{\text{red}} = L_0 . T_0 \cdot T_1 \cdot T_2 \cdot T_3\) and we know that \(\det(AB) = \det(A) \det(B)\), we see that in this example \(\det(L_{\text{red}}) = \det(L_0) = -3\).

- The product of the norms will decrease.

- If a lattice is 'decent', then a reduced basis will be nearly orthogonal and all \(n\) basis vectors will nearly be of the same length. And we expect them to be of length \(\det \frac{1}{2}\). Let’s take a non-'decent' lattice characterized by: \(m = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ K\pi & K\varepsilon & K \log 2 \end{pmatrix}\), then \(m_{\text{red}} = \begin{pmatrix} -2 & 25 \\ 2 & 25 \end{pmatrix}\). The expected length of the basis vectors of the reduced basis is \(\det(m)^{\frac{1}{2}} = 10\). Whereas the length of the smallest basis vector here is way smaller than you would expect.

2.2 Interesting applications

Now you’ve read this whole part about LLL-reducing a basis, I can imagine that you have started to wonder if it has any interesting applications or purposes. In the upcoming subsections you can find several interesting examples.

2.2.1 Computing \(|a\pi + be + c \log 2| \approx \text{small}\)

We want a linear combination of given numbers to be small, with \(a\), \(b\) and \(c\) not too big. In formula: \(|ad + be + cf| \approx \text{small}\), with \(d, e\) and \(f\) known. How we can solve this problem will be explained in the next example.

We want to determine values for \(a\), \(b\) and \(c\) for which \(|a\pi + be + c \log 2| \approx \text{small}\). Furthermore we want \(|a|, |b|, |c| \lesssim 1000\).

We choose \(K = 10^9\) and define a lattice by the matrix \(m = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ K\pi & K\varepsilon & K \log 2 \end{pmatrix}\).

If we now multiply this matrix with a vector \(\begin{pmatrix} a \\ b \\ c \end{pmatrix}\), then this results in the following equation.

\[
\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ K\pi & K\varepsilon & K \log 2 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} a \\ b \\ K(a\pi + be + c \log 2) \end{pmatrix}
\]

We see that the determinant of the matrix is \(K \log 2 \approx K\). Therefore the length of our shortest vector should be approximately \(K^{\frac{1}{2}}\). Therefore \(|a| \approx K^{\frac{1}{2}} = 10^3\) and \(|b| \approx K^{\frac{1}{2}} = 10^3\), \(K|a\pi + be + c \log 2| \approx K^{\frac{1}{2}} = 10^3\), so \(|a\pi + be + c \log 2| \approx K^{\frac{1}{2}} = 10^3\). Note that then also
2.2 Interesting applications

\[ c \approx \frac{1}{\log 2} (-a \pi - b e) \approx K^\frac{1}{3} = 10^3. \]

So with this method we should be able to get values for \(a, b\) and \(c\) that fulfil our conditions.

In Mathematica we use the LatticeReduce command again. Whereas it only works on integers we take the Floor of the third row. With the following commands we compute a transition matrix \(T\) in which the values for \(a, b\) and \(c\) are.

\[
K = 10^9;
\]
\[
m = \{\{1, 0, 0\}, \{0, 1, 0\}, \text{Floor}[\{K \ast Pi, K \ast E, K \ast Log[2]\}]\}
\]
\[
\text{lll} = \text{Transpose}[\text{LatticeReduce}[\text{Transpose}[m]]]
\]
\[
T = \text{Inverse}[m].\text{lll}
\]

The columns of \(T\) represent the values of \(a, b\) and \(c\).

Now we check if this indeed gives us a solution of our problem:

\[ |a \pi + b e + c \log 2| = \left| 113 \pi + 2e - 520 \log 2 \right| = 3.78607 \ast 10^{-7}, \text{ which is indeed small compared to the size of the coefficient.} \]

Probably with \(\max\{|a|, |b|, |c|\}\) of the same size we cannot do better. Because if we could have, we would have found a shorter lattice vector.

### 2.2.2 Finding roots of a function

Imagine you have a number, (that has 20 digits after the comma), and you are wondering if it is some root of a nice function. Normally you would search for a smart guess. However we can use the LatticeReduce command in Mathematica to find such functions.

Let’s take \( a = 1.4142135623730950488 \). Now we are wondering of which polynomial equations of degree \( \leq 5 \) with integer coefficients \(a\) could be a solution. Then we compute a matrix \( m \) with the following entries. Where \(m_{6,i} = [10^{15} \ast a^{i-1}]\).

\[
m = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 \\
565685429492380 & 400000000000000 & 2828427124746190 & 200000000000000 & 1414213562373095 & 100000000000000
\end{pmatrix}
\]

Then we compute \(m_1 = \text{Transpose}[\text{LatticeReduce}[\text{Transpose}[m]]]\).
2.2 Interesting applications

Now we compute a Transition matrix \( T \): \( T = \text{Inverse}[m].m_1 \).

\[
T = \begin{pmatrix}
0 & 0 & -1 & 0 & -7355041 & 10401599 \\
0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 1 & -3677520 & 5200799 \\
1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -2 & -1838761 & 2600399 \\
-2 & -4 & 0 & 0 & 54608393 & -77227930
\end{pmatrix}
\]

In the matrix \( T \) now each column represents an equation of which \( a \) is an approximate solution. When we take for example the first column, we see that probably: \( a^2 - 2 = 0 \). So \( a = \sqrt{2} \).

\( a \) is also a solution of: \(-7355041a^5 - 3677520a^3 + 1838761a + 54608393 = -9.16 \times 10^{-9} \approx 0.\) However we can’t classify this as a nice function anymore.

This outcome suggests that doing the same with a lower dimension would also yield a result. Let’s take dimension 3 and compute the same:

\[
m = \begin{pmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
2000000000000000 & 1414213562373095 & 1000000000000000
\end{pmatrix}
\]

and,

\[
T = \begin{pmatrix}
-1 & 0 & 0 \\
0 & -15994428 & 22619537 \\
2 & 22619537 & -31988856
\end{pmatrix}
\]

In the matrix \( T \) now each column represents an equation of which \( a \) is an approximate solution. When we take for example the first column, we see that: \(-a^2 + 2 = 0 \). Which is the same result as in the computation with dimension 6. Only the higher dimension gives more 'nice' equations of which \( a \) is an approximate solution.

2.2.3 Diophantine Approximation

In Section 7.3 of [1] we introduced the concept of Diophantine approximation. So suppose we want to check whether a given number \( a \) can be approximated by a fraction, i.e. we want to compute \( p \) and \( q \) for which \( f = a - \frac{p}{q} \approx \text{small} \). How we can solve this problem by using the LLL-algorithm will be explained in the next example.

We take \( a = \pi \), which gives us the following equation to solve: \( f = \pi - \frac{p}{q} \approx \text{small} \).

Let’s try to solve this in the most basic way: we just pick numbers for \( p \) and \( q \) and compute the above with those. We take \( q = 10^6 \) and \( p = 3141592 \). This results in \( f = 6.5359 \times 10^{-7} \).

This is a solution, but we want a better way to solve this problem. We will keep \( q \) around \( 10^6 \), to be able to compare our results with the previous example.

We choose \( K = 10^{12} \) and define a lattice by the matrix \( m = \begin{pmatrix} 1 & 0 \\ K\pi & K \end{pmatrix} \). If we now multiply this with the following vector \( \begin{pmatrix} q \\ -p \end{pmatrix} \), then this results in the following equation:
2.2 Interesting applications

\[
\begin{pmatrix}
1 & 0 \\
K\pi & K
\end{pmatrix}
\begin{pmatrix}
q \\
-p
\end{pmatrix}
= 
\begin{pmatrix}
q \\
K(\pi q - p)
\end{pmatrix}
\]

We see that the determinant of the matrix is \(K\). Therefore the length of our shortest vector should be approximately \(K^{\frac{1}{2}}\). Therefore \(q \approx K^{\frac{1}{2}} = 10^6\).

\[
K(q\pi - p) \approx K^{\frac{1}{2}} \\
\Leftrightarrow q\pi - p \approx K^{-\frac{1}{2}} \\
\Leftrightarrow \pi - \frac{q}{p} \approx \frac{K^{-\frac{1}{2}}}{K} = K^{-1} = \frac{1}{K} = 10^{-12}
\]

So with this method we can get better values for \(f\), while \(p,q\) remain approximately \(10^6\).

In Mathematica we use the LatticeReduce command again. Whereas it only works on integers we take the Floor of the second row. With the following commands we compute a transition matrix \(T\) in which the values for \(p\) and \(q\) are.

\[
K = 10^{12};
\]
\[
m = \{\{1, 0\}, \text{Floor}[\{K \times Pi, K\}]\}
\]
\[
m1 = \text{Transpose}[\text{LatticeReduce}[\text{Transpose}[m]]]
\]
\[
T = \text{Inverse}[m].m1
\]

\[
T = 
\begin{pmatrix}
-364913 & 995207 \\
1146408 & -3126535
\end{pmatrix}
\]

The columns of \(T\) represent the values of \(p\) and \(q\).

Now we check if this indeed gives a better solution than the simple guess above. We take \(q = -364913\) and \(p = -1146408\). This results in: \(f = -1.61071 \times 10^{-12}\).

This indeed gives a smaller value for \(f\). One can also notice that this value is of the same size as expected above.

This is an equivalent of the Continued Fraction algorithm.

2.2.4 Simultaneous Diophantine Approximation

When we extend the problem of the previous section to a case with the approximation of 2 numbers; \(a\) and \(b\), we want to solve a simultaneous Diophantine approximation problem. Whereas we want to solve the following set of equations for \(p_1,p_2\) and \(q\) (and not for \(q_1\) and \(q_2\), whereas this yields solving two equations as in the previous section):

- \(f_1 = a - \frac{p_1}{q} \approx \text{small}\)
- \(f_2 = b - \frac{p_2}{q} \approx \text{small}\)

How we can solve this problem by using the LLL-algorithm will be shown by the next example. Suppose we want to solve the following set of equations for \(p_1,p_2\) and \(q\):

- \(f_1 = \pi - \frac{p_1}{q} \approx \text{small}\)
• \( f_2 = e - \frac{p_2}{q} \approx \text{small} \)

Let us again try to solve this in the most basic way: we just pick numbers for \( p_1, p_2 \) and \( q \) and compute the above with those. We take \( q = 10^6 \), \( p_1 = 3141592 \) and \( p_2 = 2718281 \). This results in:

• \( f_1 = 6.5359 \times 10^{-7} \)
• \( f_2 = 8.28459 \times 10^{-7} \)

This is a solution, but we want a better way to solve this problem. We will keep \( q \) around \( 10^6 \), to be able to compare our results with the previous example.

In a previous subsection we had an analogue example, in which we wanted to make \( |a\pi + be + c\log2| \) small. In this problem we have two equations with in total three unknowns.

We choose \( K = 10^6 \) and define a lattice by the matrix \( m = \begin{pmatrix} 1 & 0 & 0 \\ K\pi & K & 0 \\ Ke & 0 & K \end{pmatrix} \). If we now multiply this matrix with the vector \( \begin{pmatrix} q \\ -p_1 \\ -p_2 \end{pmatrix} \), then this results in the following equation:

\[
\begin{pmatrix} 1 & 0 & 0 \\ K\pi & K & 0 \\ Ke & 0 & K \end{pmatrix} \begin{pmatrix} q \\ -p_1 \\ -p_2 \end{pmatrix} = \begin{pmatrix} q \\ K(\pi q - p_1) \\ K(eq - p_2) \end{pmatrix}
\]

We see that the determinant of the matrix is \( K^2 \). Therefore the length of our shortest vector should be approximately \( K^{\frac{2}{3}} \). Therefore \( q \approx K^{\frac{2}{3}} = 10^6 \).

\[
K(q\pi - p_1) \approx K^{\frac{2}{3}} \\
⇔ q\pi - p_1 \approx K^{-\frac{1}{3}} \\
⇔ \pi - \frac{q}{p_1} \approx K^{\frac{1}{3}} \approx \frac{1}{K} = 10^{-9}
\]

\[
K(eq - p_2) \approx K^{\frac{2}{3}} \\
⇔ eq - p_2 \approx K^{-\frac{1}{3}} \\
⇔ e - \frac{q}{p_2} \approx K^{\frac{1}{3}} \approx \frac{1}{K} = 10^{-9}
\]

So with this method we can get better values for \( f_1 \) and \( f_2 \), while \( q, p_1 \) and \( p_2 \) remain approximately \( 10^6 \).

In Mathematica we use the LatticeReduce command again. Whereas it only works on integers we take the Floor of the second and the third row. With the following commands we compute a transition matrix \( T \) in which the values for \( p_1, p_2 \) and \( q \) are.

\[
K = 10^6; \\
m = \{\{1, 0, 0\}, \text{Floor}\{K*Pi, K, 0\}, \text{Floor}\{K*E, 0, K\}\} \\
m1 = \text{Inverse}[\text{LatticeReduce}[\text{Transpose}[m]]] \\
T = \text{Inverse}[m].m1
\]
2.2 Interesting applications

\[ T = \begin{pmatrix}
-596401 & 432134 & 657746 \\
1873649 & -1357589 & -2066370 \\
1621186 & -1174662 & -1787939 \\
\end{pmatrix} \]

The columns of \( T \) represent the values of \( p_1, p_2 \) and \( q \).

Now we check if this indeed gives a better solution than the simple guess above. We take \( q = -596401, p_1 = -1873649 \) and \( p_2 = -1621186 \). This results in:

- \( f_1 = 3.24624 \times 10^{-10} \)
- \( f_2 = 1.29913 \times 10^{-9} \)

This indeed gives smaller values for \( f_1 \) and \( f_2 \). We also see that these values are of the same size as expected above.

Suppose we want to solve the following equations:

- \( f_1 = \pi - \frac{p_1}{q} \approx 10^{-9} \)
- \( f_2 = e - \frac{p_2}{q} \approx 10^{-12} \)

Herefore we can use the same method as above, with a \( K_1 = 10^9 \) and a \( K_2 = 10^{12} \). In the second row of the matrix we will now use \( K_1 \) instead of \( K \) and in the third row \( K_2 \) instead of \( K \).

\[
\begin{pmatrix}
1 & 0 & 0 \\
K_1 \pi & K_1 & 0 \\
eK_2 & 0 & K_2 \\
\end{pmatrix}
\begin{pmatrix}
q \\
-p_1 \\
-p_2 \\
\end{pmatrix}
= 
\begin{pmatrix}
q \\
K_1 (\pi q - p_1) \\
K_2 (eq - p_2) \\
\end{pmatrix}
\]

And this indeed yields the required result.

- \( f_1 = -7.97861 \times 10^{-9} \)
- \( f_2 = 4.81837 \times 10^{-13} \)

When we want to solve an extra equation, we need to add an extra dimension to the matrix, whereas an extra equation means an extra unknown.
2.2 Interesting applications

2.2.5 Inhomogeneous Diophantine approximation

Suppose you have a lattice in $\mathbb{R}^2$ and you pick a random point in $\mathbb{R}^2$. Now what is the lattice point closest to your point? How to solve this will be shown by the upcoming example.

As a lattice we take the lattice represented by $L = \begin{pmatrix} 1 & 0 \\ 144 & 89 \end{pmatrix}$. We already used this lattice in Section 7.4.3 of [1] and there we saw that $L_{\text{red}} = \begin{pmatrix} 5 & -8 \\ 8 & 5 \end{pmatrix}$ is its reduced basis. As a point we pick $x = \begin{pmatrix} 20 \\ 20 \end{pmatrix}$.

[Diagram showing a lattice with a not-reduced red basis, a reduced green basis and two closest vectors]

We want to solve: $Ly = x$, which is equivalent to solving: $y = L^{-1}x$. In here we want $y$ to represent the closest point with respect to the given lattice representation, however we want the 'closest' lattice point in $\mathbb{R}^2$. Therefore we compute $x_1 = L[y]$, which will give us a lattice point.

This gives us for $L$ and $x$ as above: $x_1 = \begin{pmatrix} 20 \\ 32 \end{pmatrix}$.

If we compute $\|x - x_1\| = 12$. This difference is quite large compared to $\det^\frac{1}{2}$, so $x_1$ might not be the closest lattice point. Therefore we will repeat the computation with $L_{\text{red}}$ and $x$. This gives
2.2 Interesting applications

us: \( x_2 = \begin{pmatrix} 19 \\ 23 \end{pmatrix} \). And \( \| x - x_2 \| = 3.16 \). So the LLL-reduced basis gives a better solution; and thus a more accurate closest point. This is visualized in the figure below.

In the remainder of this section \( x_1 \) will represent the ‘closest’ lattice point computed with the original basis and \( x_2 \) will represent the ‘closest’ lattice point computed with the LLL-reduced basis.

Note that a visualization might give you a wrong idea; i.e. look at the following two figures.

Here we represent the same vectors in a different coordinate system. In the first figure you see the ‘closest vectors’ with respect to the not-reduced basis and in the second figure you see the ‘closest vectors’ with respect to the reduced basis. The first figure suggests that \( x_1 \) is our best solution, whereas we’ve just shown that \( x_2 \) yields a better solution, which is shown in the second figure.
2.2 Interesting applications

In general the LLL-reduced basis gives the same or a better solution. I.e. if you take $x = \begin{pmatrix} 300 \\ 300 \end{pmatrix}$, $x_1$ and $x_2$ will be the same point.

But why does this reduced matrix give a better solution? When we compare $x_1$ and $x_2$ we see that $x_1$ is very accurate in the first entry, but that is compensated by the second entry. The entries in $x_2$ are both a bit 'off', but only a bit. This is due to the fact that the reduced matrix is closer to orthogonality than the original matrix is.

Note: These solutions don’t need to be the 'best' solution. To determine what the best solution is, we need a bit more checking.

2.2.6 Minimization of a convex function over $\mathbb{Z}^2$

In [2] the minimization of a convex function over $\mathbb{Z}^2$ is discussed. The authors also show that for a given vector $v \in \mathbb{Q}^2$ and a rational $2 \times 2$ nonsingular matrix $A$, a vector $x^*$ can be found which minimizes $\|Ax - v\|^2$. In other words, it minimizes the distance between $Ax$ and our given vector $v$. This is essentially the same as we have seen in the previous subsection, only with a slightly different method it seems at first. Let us take a look at the following example:

$$A = \begin{pmatrix} 1 & 0 \\ 144 & 89 \end{pmatrix}, \text{ and } v = \begin{pmatrix} 20 \\ 20 \end{pmatrix}.$$

For trying to minimize $f(x) = \|Ax - v\|^2$ we will start with a basic triple $a, b, c \in \mathbb{Z}^2$ so that $f(a) \leq f(b) \leq f(c) < \infty$. We will take: $a = (0, 0), b = (0, 1)$ and $c = (1, 0)$.

$$f((0,0)) = 800$$
$$f((0,1)) = 5161$$
$$f((1,0)) = 15737$$

So our basic triple meets $f(a) \leq f(b) \leq f(c) < \infty$.

Now we are going to repeat the following as long as this yields an improvement:

1. let $a'$ be a minimizer of $\min\{f(a + \lambda(b - a))|\lambda \in \mathbb{Z}\}$
2. let $b'$ be a minimizer of $\min\{f(a' + \sigma(b - a))|\sigma \in \{-1, 1\}\}$
3. let $c'$ be a minimizer of $\min\{f(a + \lambda(b - a)) + \sigma(c - a)|\lambda \in \mathbb{Z}, \sigma \in \{-1, 1\}\}$
4. replace $(a, b, c)$ by $(a', b', c')$
5. reorder $a, b, c$ so that $f(a) \leq f(b) \leq f(c)$

In this case we need six steps (shown in the table below); the 6th step does not yield an additional improvement, so our algorithm ends.

December 1, 2011 Bachelor thesis - Lattice Basis Reduction
## 2.2 Interesting Applications

<table>
<thead>
<tr>
<th>Step</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>f(a)</th>
<th>f(b)</th>
<th>f(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(0,0)</td>
<td>(0,1)</td>
<td>(1,0)</td>
<td>800</td>
<td>5161</td>
<td>15737</td>
</tr>
<tr>
<td>1</td>
<td>(-1.2)</td>
<td>(0,0)</td>
<td>(0,1)</td>
<td>637</td>
<td>800</td>
<td>5161</td>
</tr>
<tr>
<td>2</td>
<td>(2,-3)</td>
<td>(-1.2)</td>
<td>(0,0)</td>
<td>325</td>
<td>637</td>
<td>800</td>
</tr>
<tr>
<td>3</td>
<td>(10,-16)</td>
<td>(2,-3)</td>
<td>(5,-8)</td>
<td>116</td>
<td>325</td>
<td>369</td>
</tr>
<tr>
<td>4</td>
<td>(23,-37)</td>
<td>(18,-29)</td>
<td>(10,-16)</td>
<td>10</td>
<td>85</td>
<td>116</td>
</tr>
<tr>
<td>5</td>
<td>(23,-37)</td>
<td>(15,-24)</td>
<td>(18,-29)</td>
<td>10</td>
<td>41</td>
<td>85</td>
</tr>
<tr>
<td>6</td>
<td>(23,-37)</td>
<td>(15,-24)</td>
<td>(18,-29)</td>
<td>10</td>
<td>41</td>
<td>85</td>
</tr>
</tbody>
</table>

Then our basic triple $a, b, c$ becomes: $a = (23, -37)$, $b = (15, -24)$, $c = (18, -29)$. Now $Aa$ should represent the closest vector. $Aa = \begin{pmatrix} 23 \\ 19 \end{pmatrix}$, which is exactly the same vector as our previous algorithm gave. Furthermore $Ab, Ac$ should be a reduced basis for our matrix. $Ab = \begin{pmatrix} 15 \\ 24 \end{pmatrix}$, $Ac = \begin{pmatrix} 18 \\ 11 \end{pmatrix}$; these are indeed vectors in our lattice. But it does yield a different reduced basis than the LLL-algorithm.

So why are these two methods similar to each other? For that we will look at the steps we take in each method. While performing the LLL algorithm we are first trying to make a vector shorter and then we will swap vectors to meet a certain condition (in this case the Lovász condition). While performing the Convex Minimization Method we are reducing $f(x) = \|Ax - v\|^2$ and after that we swap our vectors to meet a certain condition (in this case $f(a) \leq f(b) \leq f(c)$). So as

### Closest lattice vector

**Note:** We can do the same with our LLL-reduced basis. This yields exactly the same result, only less steps are required, which are shown in the table below.

<table>
<thead>
<tr>
<th>Step</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>f(a)</th>
<th>f(b)</th>
<th>f(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(1,0)</td>
<td>(0,0)</td>
<td>(0,1)</td>
<td>369</td>
<td>800</td>
<td>1009</td>
</tr>
<tr>
<td>1</td>
<td>(3,-1)</td>
<td>(3,0)</td>
<td>(2,0)</td>
<td>10</td>
<td>41</td>
<td>116</td>
</tr>
<tr>
<td>2</td>
<td>(3,-1)</td>
<td>(3,0)</td>
<td>(2,-1)</td>
<td>10</td>
<td>41</td>
<td>85</td>
</tr>
<tr>
<td>3</td>
<td>(3,-1)</td>
<td>(3,0)</td>
<td>(2,-1)</td>
<td>10</td>
<td>41</td>
<td>85</td>
</tr>
</tbody>
</table>
you may notice the outline of the two methods is similar, the results however don’t need to be (as already shown in the previous example).
2.3 Exercises

2.1. Which of the following matrices represent an LLL-reduced basis (with $\delta = \frac{3}{4}$)?

\[
\begin{pmatrix}
1 & 0 \\
0 & 3
\end{pmatrix},
\begin{pmatrix}
3 & 1 \\
1 & 2
\end{pmatrix},
\begin{pmatrix}
9 & 0 \\
0 & 17
\end{pmatrix},
\begin{pmatrix}
7 & 5 & 1 \\
7 & -2 & 8 \\
9 & -7 & 2
\end{pmatrix},
\begin{pmatrix}
1 & 0 & 0 \\
0 & 4 & 0 \\
0 & 0 & 1
\end{pmatrix},
\begin{pmatrix}
2 & 0 & 1 \\
3 & -2 & 4 \\
1 & 0 & 2
\end{pmatrix}
\]

2.2. Compute an LLL-reduced basis of the following matrices 'by hand'. Verify your solutions with Mathematica.

\[
\begin{pmatrix}
1 & 0 & 0 \\
4 & 2 & 15 \\
0 & 0 & 3
\end{pmatrix},
\begin{pmatrix}
2 & 3 & 7 \\
0 & 0 & 1 \\
5 & 0 & 3
\end{pmatrix},
\begin{pmatrix}
2 & 0 & 1 \\
1 & 0 & 2 \\
0 & -2 & 0
\end{pmatrix}
\]

2.3. Suppose we want $|\pi - \frac{x}{q}| < 10^{-k}$. How large should $q$ approximately be?

2.4. Can you find a polynomial for which $x = -12.03315274900869619514987591929$ is a solution?

2.5. There is a nice solution for $x_1 \arctan 1 + x_2 \arctan \frac{1}{5} + x_3 \arctan \frac{1}{239} = 0$. Can you find it?

2.6. Suppose $K = 10^9$ and $L = \begin{pmatrix} 1 & 0 & 0 \\ K\pi & K\varepsilon & K\sqrt{2} \end{pmatrix}$. Compute the closest lattice point you can get to $x = \begin{pmatrix} 0 \\ 0 \\ K\log 2 \end{pmatrix}$.
2.4 Answers

Exercise 2.1
Yes, No, Yes, Yes, No, No.

Exercise 2.2

'By hand': \[
\begin{pmatrix}
1 & 0 & 0 \\
0 & 2 & -1 \\
0 & 0 & 3
\end{pmatrix}, \quad 
\begin{pmatrix}
-1 & 0 & 3 \\
1 & 2 & 0 \\
-2 & 1 & 0
\end{pmatrix}, \quad 
\begin{pmatrix}
-1 & 2 & 0 \\
1 & 1 & 0 \\
0 & 0 & -2
\end{pmatrix}.
\]
Mathematica: \[
\begin{pmatrix}
1 & 0 & 0 \\
0 & 2 & -1 \\
0 & 0 & 3
\end{pmatrix}, \quad 
\begin{pmatrix}
-1 & 0 & 3 \\
1 & 2 & 0 \\
-2 & 1 & 0
\end{pmatrix}, \quad 
\begin{pmatrix}
-1 & 2 & 0 \\
1 & 1 & 0 \\
0 & 0 & -2
\end{pmatrix}.
\]

Exercise 2.3

$q$ should be approximately $10^{\frac{1}{2}}$

Exercise 2.4

\[f = x^5 + 12x^4 + x^3 + 17x^2 + 3x + 12\]

Exercise 2.5

Yes, \(x_1 = 1, x_2 = -4\) and \(x_3 = 1\). Also known as Machin’s formula.

Exercise 2.6

Closest lattice point: \[
\begin{pmatrix}
484 \\
-325 \\
693147052
\end{pmatrix}.
\]
Bibliography