# An application of the algorithmic of euclidean lattices to computer arithmetic : machine-efficient polynomial approximants

N. Brisebarre, S. Chevillard, G. Hanrot, J.-M. Muller, A. Tisserand and S. Torres
Arénaire, LIP, É.N.S. Lyon

Interactions Mathématiques & Informatique à Montpellier
20/01/2010

# Function evaluation on a machine

Problem: evaluation of a function $\varphi$ over $\mathbb{R}$ or a subset of $\mathbb{R}$.

We wish to only use additions, subtractions, multiplications (we should avoid divisions) $\Rightarrow$ use of polynomials.

Most of the algorithms for evaluating elementary functions ($\exp, \ln, \cos, \sin,$ $\arctan, \sqrt{\ }, \ldots$) use polynomial approximants.

# Floating Point (FP) Arithmetic

Given

$$\begin{cases} \text{a radix} & \beta \geq 2, \\ \text{a precision} & n \geq 1, \\ \text{a set of exponents} & E_{\text{min}} \cdots E_{\text{max}}. \end{cases}$$

A finite FP number $x$ is represented by $2$ integers:

- integer mantissa : $M$, $\beta^{n-1} \leq |M| \leq \beta^n - 1$;
- exponent $E$, $E_{\text{min}} \leq e \leq E_{\text{max}}$

such that

$$x = \frac{M}{\beta^{n-1}} \times \beta^e.$$

We call real mantissa, or mantissa of $x$ the number $m = M \times \beta^{1-n}$, such that $x = m \times \beta^e$.

We assume binary FP arithmetic (that is to say $\beta = 2$.)

# IEEE precisions

`http://babbage.cs.qc.edu/courses/cs341/IEEE-754references.html`

| | precision | minimal exponent | maximal exponent |
|---|---|---|---|
| single | $24$ | $-126$ | $127$ |
| double | $53$ | $-1022$ | $1023$ |
| extended double | $64$ | $-16382$ | $16383$ |
| quadruple | $113$ | $-16382$ | $16383$ |

# Evaluation of elementary functions

$\exp, \ln, \cos, \sin, \arctan, \sqrt{\ }, \ldots$

First step. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function $\varphi$ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function $f$ over $[a, b]$.

Second step. Polynomial approximation of $f$:

- least square approximation;
- minimax approximation.

# Minimax Approximation

Reminder. Let $g : [a, b] \to \mathbb{R}$, $||g||_{[a,b]} = \sup_{a \leq x \leq b} |g(x)|$.

We denote $\mathbb{R}_n[X] = \{p \in \mathbb{R}[X]; \deg p \leq n\}$.

Minimax approximation: let $f : [a, b] \to \mathbb{R}$, $n \in \mathbb{N}$, we search for $p \in \mathbb{R}_n[X]$ s.t.

$$||p - f||_{[a,b]} = \inf_{q \in \mathbb{R}_n[X]} ||q - f||_{[a,b]}.$$

An algorithm by Remez gives $p$ (`minimax` function in Maple, also available in Sollya `http://sollya.gforge.inria.fr/`).

Problem: we can't directly use minimax approx. in a computer since the coefficients of $p$ can't be represented on a finite number of bits.

Our context: the coefficients of the polynomials must be written on a finite (imposed) number of bits.

Let $m = (m_i)_{0 \le i \le n}$ a finite sequence of rational integers.

Let $q(x) = q_0 + q_1 x + \cdots + q_n x^n \in \mathbb{R}_n[x]$. Each $q_i$ must be an integer multiple of $2^{-m_i}$: $q_i = a_i / 2^{m_i}$ with $a_i \in \mathbb{Z}$.

# Truncated Polynomials

Let $m = (m_i)_{0 \le i \le n}$ a finite sequence of rational integers. Let

$$\mathcal{P}_n^m = \{q = q_0 + q_1 x + \cdots + q_n x^n \in \mathbb{R}_n[X]; q_i \text{ integer multiple of } 2^{-m_i}, \forall i\}.$$

First idea. Remez $\longrightarrow p(x) = p_0 + p_1 x + \cdots + p_n x^n$. Every $p_i$ rounded to $\hat{a}_i/2^{m_i}$, the nearest integer multiple of $2^{-m_i} \longrightarrow \hat{p}(x) = \frac{\hat{a}_0}{2^{m_0}} + \frac{\hat{a}_1}{2^{m_1}} x + \cdots + \frac{\hat{a}_n}{2^{m_n}} x^n$.

Problem: $\hat{p}$ not necessarily a minimax approx. of $f$ among the polynomials of $\mathcal{P}_n^m$.

# Approximation of the function $\cos$ over $[0, \pi/4]$ by a degree-$3$ polynomial

Maple or Sollya tell us that the polynomial

$$p = 0.9998864206 + 0.00469021603x - 0.5303088665x^2 + 0.06304636099x^3$$

is $\sim$ the best approximant to $\cos$. We have $\varepsilon = ||\cos - p||_{[0,\pi/4]} = 0.0001135879...$.

We look for $a_0, a_1, a_2, a_3 \in \mathbb{Z}$ such that

$$\max_{0 \le x \le \pi/4} \left| \cos x - \left( \frac{a_0}{2^{12}} + \frac{a_1}{2^{10}} x + \frac{a_2}{2^6} x^2 + \frac{a_3}{2^4} x^3 \right) \right|$$

is minimal.

# Approximation of the function $\cos$ over $[0, \pi/4]$ by a degree-$3$ polynomial

Maple or Sollya tell us that the polynomial

$$p = 0.9998864206 + 0.00469021603x - 0.5303088665x^2 + 0.06304636099x^3$$

is $\sim$ the best approximant to $\cos$. We have $\varepsilon = ||\cos - p||_{[0,\pi/4]} = 0.0001135879...$.

We look for $a_0, a_1, a_2, a_3 \in \mathbb{Z}$ such that

$$\max_{0 \le x \le \pi/4} \left| \cos x - \left( \frac{a_0}{2^{12}} + \frac{a_1}{2^{10}}x + \frac{a_2}{2^6}x^2 + \frac{a_3}{2^4}x^3 \right) \right|$$

is minimal.

The naive approach gives the polynomial $\hat{p} = \frac{2^{12}}{2^{12}} + \frac{5}{2^{10}}x - \frac{34}{2^6}x^2 + \frac{1}{2^4}x^3$. We have $\hat{\varepsilon} = ||\cos - \hat{p}||_{[0,\pi/4]} = 0.00069397...$.

# Applications

Two targets:

- specific hardware implementations in low precision ($\sim 15$ bits). Reduce the cost (time and silicon area) keeping a correct accuracy;
- single or double IEEE precision software implementations. Get very high accuracy keeping an acceptable cost (time and memory).

# Statement of the problem

Let $f : [a, b] \to \mathbb{R}, n \in \mathbb{N},\ \ m = (m_i)_{0 \le i \le n}$ a finite sequence of rational integers, $p(x) = p_0 + p_1 x + \cdots + p_n x^n$ the minimax approx. of $f$ over $[a, b]$ (Remez). Let

$$\mathcal{P}_n^m = \left\{ q(x) = \frac{a_0}{2^{m_0}} + \frac{a_1}{2^{m_1}} x + \cdots + \frac{a_n}{2^{m_n}} x^n ; a_i \in \mathbb{Z}, \forall i \right\}.$$

Every $p_i$ rounded to $\hat{a}_i / 2^{m_i}$, the nearest integer multiple of $2^{-m_i}$ $\longrightarrow$ $\hat{p}(x) = \frac{\hat{a}_0}{2^{m_0}} + \frac{\hat{a}_1}{2^{m_1}} x + \cdots + \frac{\hat{a}_n}{2^{m_n}} x^n$.

# Statement of the problem

Let $f : [a, b] \to \mathbb{R}, n \in \mathbb{N}$, $m = (m_i)_{0 \leq i \leq n}$ a finite sequence of rational integers, $p(x) = p_0 + p_1 x + \cdots + p_n x^n$ the minimax approx. of $f$ over $[a, b]$ (Remez). Let

$$\mathcal{P}_n^m = \left\{ q(x) = \frac{a_0}{2^{m_0}} + \frac{a_1}{2^{m_1}} x + \cdots + \frac{a_n}{2^{m_n}} x^n ; a_i \in \mathbb{Z}, \forall i \right\}.$$

Every $p_i$ rounded to $\hat{a}_i / 2^{m_i}$, the nearest integer multiple of $2^{-m_i}$ $\longrightarrow$ $\hat{p}(x) = \frac{\hat{a}_0}{2^{m_0}} + \frac{\hat{a}_1}{2^{m_1}} x + \cdots + \frac{\hat{a}_n}{2^{m_n}} x^n$.

Let

$$\varepsilon = ||f - p||_{[a,b]} \text{ and } \hat{\varepsilon} = ||f - \hat{p}||_{[a,b]}.$$

We compare $\varepsilon$ to $\hat{\varepsilon}$.

Let

$$\varepsilon = ||f - p||_{[a,b]} \text{ and } \hat{\varepsilon} = ||f - \hat{p}||_{[a,b]}.$$

We compare $\varepsilon$ to $\hat{\varepsilon}$.

Given $K \in [\varepsilon, \hat{\varepsilon}]$. We search for a truncated polynomial $p^\star \in \mathcal{P}_n^m$ s.t.

$$||f - p^\star||_{[a,b]} = \min_{q \in \mathcal{P}_n^m} ||f - q||_{[a,b]}$$

and

$$||f - p^\star||_{[a,b]} \leq K.$$

# A first approach

We put $p^\star(x) = \frac{a_0^\star}{2^{m_0}} + \frac{a_1^\star}{2^{m_1}}x + \cdots + \frac{a_n^\star}{2^{m_n}}x^n$ ($a_0^\star, \ldots, a_n^\star \in \mathbb{Z}$ are the unknowns).

1. We find relations satisfied by the $a_i^\star$ $\Rightarrow$ finite number of candidate polynomials.

2. If this number is small enough, we perform an exhaustive search: computation of the norms $||f - q||_{[a,b]}$, $q$ running among the candidate polynomials.

# A tool for realizing this approach: polytopes

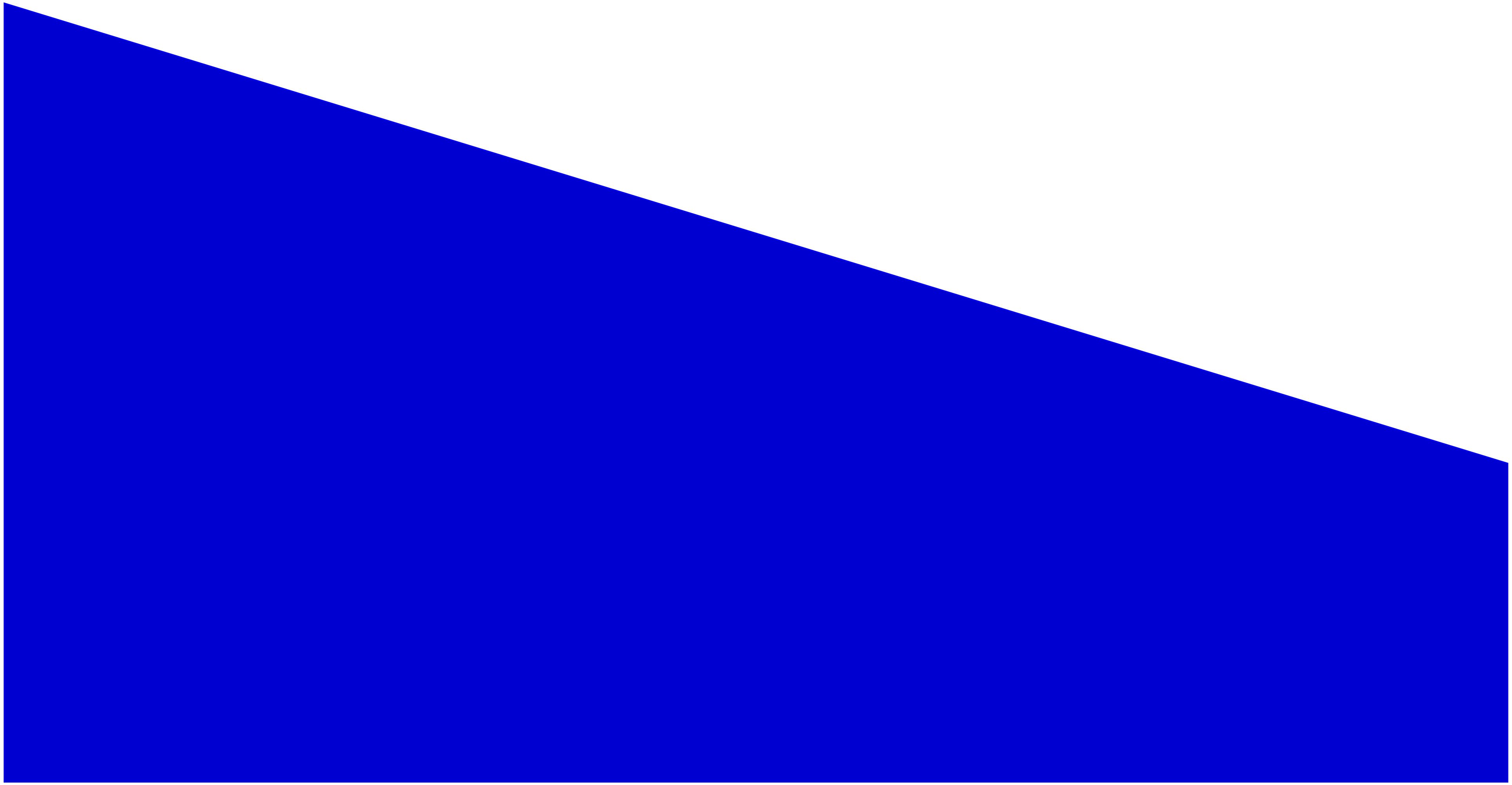
**Definitions .** *Let* $k \in \mathbb{N}$*.*

*A polyhedron is a subset* $\mathfrak{P}$ *of* $\mathbb{R}^k$ *s.t. there exists a matrix* $A \in \mathcal{M}_{m,k}(\mathbb{R})$ *and a vector* $b \in \mathbb{R}^m$ *(with* $m \geq 0$*) s.t.*

$$\mathfrak{P} = \{x \in \mathbb{R}^k | Ax \leq b\}.$$

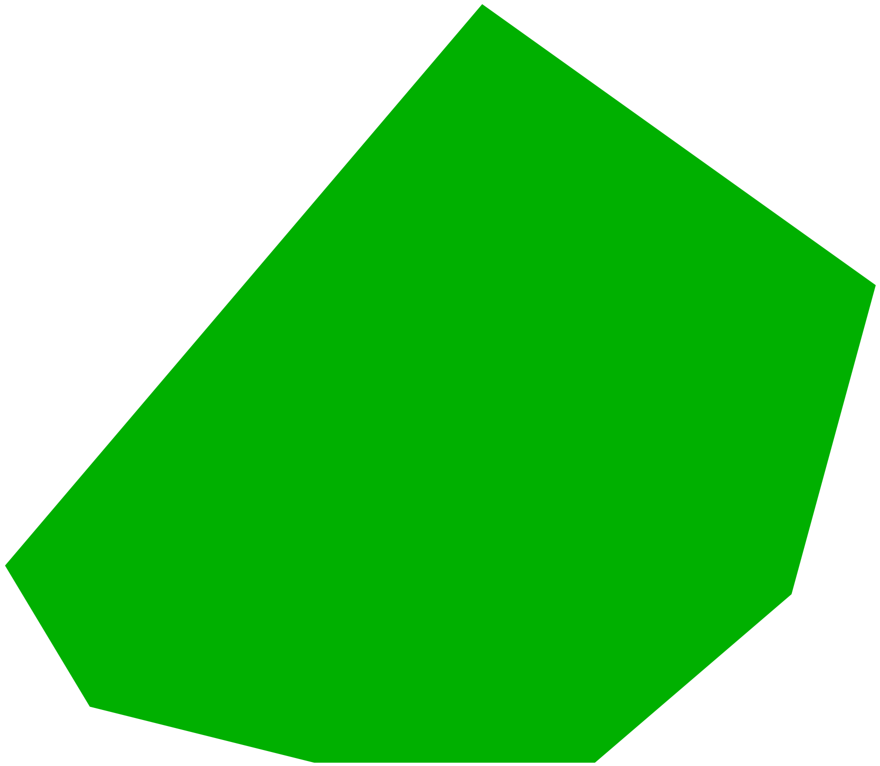
*A polytope is a bounded polyhedron.*

*A polyhedron (resp. polytope)* $\mathfrak{P}$ *is rational if it is defined by a matrix and a vector with rational coefficients.*



An example of polyhedron: $\{(x, y) \in \mathbb{R}^2 : x + 3y \leq 2\}$ (half-plane in $\mathbb{R}^2$).

An example of polyhedron: $\{(x, y) \in \mathbb{R}^2 : 2x - 3y \leq 10, x + 3y \geq 1\}$ (cone in $\mathbb{R}^2$).



An example of polytope.

# Reminder of the problem

We put

$$\varepsilon = ||f - p||_{[a,b]} \text{ and } \hat{\varepsilon} = ||f - \hat{p}||_{[a,b]}.$$

We compare $\varepsilon$ to $\hat{\varepsilon}$.

Given $K \in [\varepsilon, \hat{\varepsilon}]$. We search for a truncated polynomial $p^\star \in \mathcal{P}_n^m$ s.t.

$$||f - p^\star||_{[a,b]} = \min_{q \in \mathcal{P}_n^m} ||f - q||_{[a,b]}$$

and

$$||f - p^\star||_{[a,b]} \leq K.$$

For all $x \in [a, b]$, we must have

$$f(x) - K \leq \sum_{i=0}^{n} \frac{a_i^\star}{2^{m_i}} x^i \leq f(x) + K, \tag{1}$$

($a_0^\star, \ldots, a_n^\star \in \mathbb{Z}$ are the unknowns).

Idea: plug a certain number of points of $[a, b]$ into (1) in order to construct a polytope $\mathfrak{P}$ which the points $(a_0^\star, \ldots, a_n^\star)$ belong to. Then scan the points of $\mathfrak{P} \cap \mathbb{Z}^{n+1}$.

If we want to use algorithmic tools, all the input data should belong to $\mathbb{Q}$.

For all $x \in [a, b]$, we must have

$$f(x) - K \leq \sum_{i=0}^{n} \frac{a_i^\star}{2^{m_i}} x^i \leq f(x) + K, \tag{2}$$

($a_0^\star, \ldots, a_n^\star \in \mathbb{Z}$ are the unknowns).

Let $x = r/s$ with $r \in \mathbb{Z}, s \in \mathbb{N}$. We have

$$f\left(\frac{r}{s}\right) - K \leq \sum_{i=0}^{n} \frac{a_i^\star}{2^{m_i}} \left(\frac{r}{s}\right)^i \leq f\left(\frac{r}{s}\right) + K.$$

We choose $m(\frac{r}{s})$ and $M(\frac{r}{s}) \in \mathbb{Q}$ such that $m(\frac{r}{s}) \leq f\left(\frac{r}{s}\right) - K$ and $f\left(\frac{r}{s}\right) + K \leq M(\frac{r}{s})$, $m(\frac{r}{s})$ "close" to $f\left(\frac{r}{s}\right) - K$ and $M(\frac{r}{s})$ "close" to $f\left(\frac{r}{s}\right) + K$.

We plug into (2) $d$ rational numbers from $[a, b]$ (choice is important).

We plug into (2) $d$ rational numbers from $[a, b]$ (choice is important).

If $d \geq n + 1 \Rightarrow$ we have a rational polytope whose the integers $a_i^\star$ are elements.

Perform exhaustive research by scanning the points with integer coordinates of the polytope.

We can use C libraries (s.t. PIP) designed for efficiently scanning the integer points of polytopes.

**Remark .** *Gives only candidates (but forgets none of them).*

Method works over any $[a, b]$.

We must have

$$f(x) - K \leq \sum_{i=0}^{n} \frac{a_i^\star}{2^{m_i}} x^i \leq f(x) + K \qquad (3)$$

for all $x \in [a, b]$.

1. Let $x_1, \ldots, x_d$ a finite sequence of $\mathbb{Q} \cap [a, b]$.
2. We plug the $x_k$ into (3). We compute rational approx. of the $f(x_k) - K$ and $f(x_k) + K$.

   $d \geq n + 1 \Rightarrow$ we have a rational polytope which the integers $a_i^\star$ belong to.
3. Perform exhaustive search by scanning the points with integer coord. of the polytope. To do so, we use C libraries (such as PIP) designed for efficiently scanning the integer points of polytopes.

# Approximation of the function $\cos$ over $[0, \pi/4]$ by a degree-$3$ polynomial

Maple or Sollya tell us that the polynomial

$$p = 0.9998864206 + 0.00469021603x - 0.5303088665x^2 + 0.06304636099x^3$$

is $\sim$ the best approximant to $\cos$. We have $\varepsilon = ||\cos - p||_{[0,\pi/4]} = 0.0001135879...$.

We look for $a_0, a_1, a_2, a_3 \in \mathbb{Z}$ such that

$$\max_{0 \le x \le \pi/4} \left| \cos x - \left( \frac{a_0}{2^{12}} + \frac{a_1}{2^{10}} x + \frac{a_2}{2^6} x^2 + \frac{a_3}{2^4} x^3 \right) \right|$$

is minimal.

The naive approach gives the polynomial $\hat{p} = \frac{2^{12}}{2^{12}} + \frac{5}{2^{10}} x - \frac{34}{2^6} x^2 + \frac{1}{2^4} x^3$. We have $\hat{\varepsilon} = ||\cos - \hat{p}||_{[0,\pi/4]} = 0.00069397...$.

Best approximant:

$$p^\star = \frac{4095}{2^{12}} + \frac{6}{2^{10}}\,x - \frac{34}{2^6}\,x^2 + \frac{1}{2^4}\,x^3$$

which gives a distance to $\cos$, $||\cos - p^\star||_{[0,\pi/4]}$, equal to $0.0002441406250$.

In this example, we gain $-\log_2(0.35) \approx 1.5$ bits of accuracy.

# The polytope method is flexible!

We can add some constraints (fix values of some coef. for instance) or use "weighted" infinite norms.

**Examples .**

- *We can restrict our search to odd truncated polynomials* $\sum_{i=0}^{n} \frac{a_i^\star}{2^{m_i}} x^{2i+1}$.

- *We can restrict our search to truncated polynomials whose constant term is* $1$, *we consider* $1 + \sum_{i=1}^{n} \frac{a_i^\star}{2^{m_i}} x_k^i$.

- *We can search for a best truncated polynomial for the relative error* $|| \cdot ||_{rel,[a,b]}$ *defined by*

$$||f - p||_{rel,[a,b]} = \sup_{a \leq x \leq b} \left| \frac{p(x)}{f(x)} - 1 \right|.$$

This method gives a best polynomial for a given sequence of $m_i$.

It should make it possible to tackle with degree-8 or 10 polynomials: this is nice for hardware-oriented applications but not satisfying for all software-oriented applications.

Another drawback: we need to have a good insight of the error $K$.

- if $K$ is underestimated, there won't be any solution found,
- if $K$ is overestimated, there might be far too many candidates: it becomes untractable.

We designed a tool for getting a relevant estimate of $K$.

This tool proved to give more than expected.

# A second approach through lattice basis reduction

# A reminder on lattice basis reduction

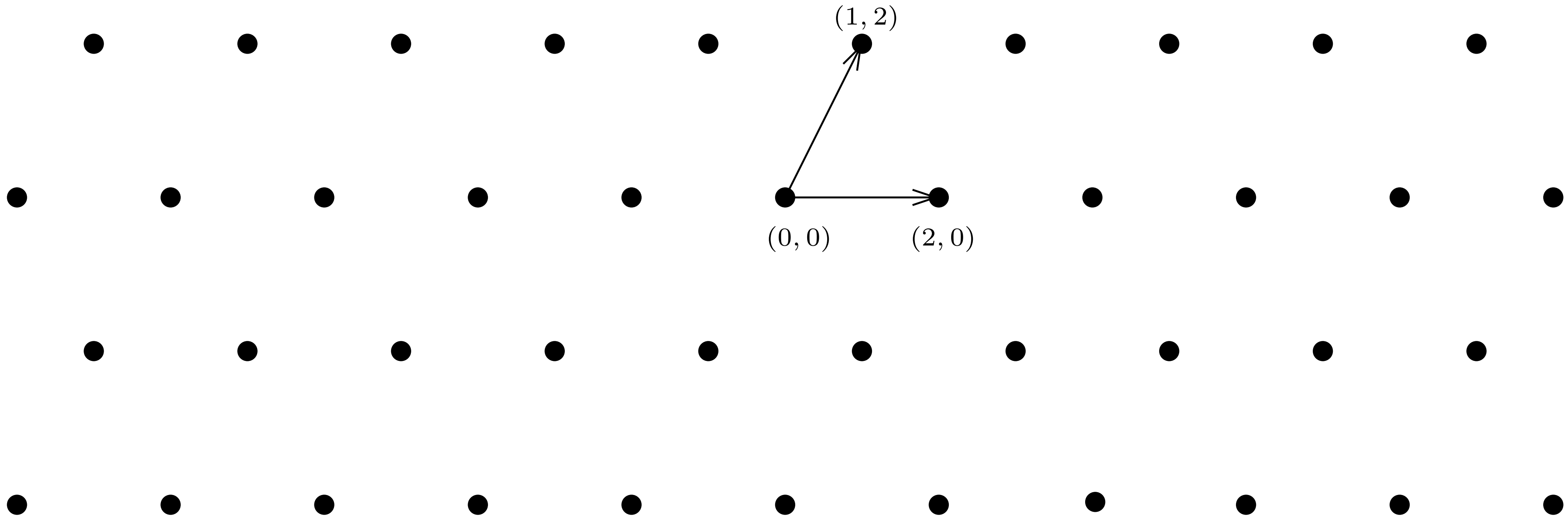
**Definition .** *Let $L$ be a nonempty subset of $\mathbb{R}^d$, $L$ is a lattice iff there exists a set of vectors $b_1, \ldots, b_k$ $\mathbb{R}$-linearly independent such that*

$$L = \mathbb{Z}.b_1 \oplus \cdots \oplus \mathbb{Z}.b_k.$$

$(b_1, \ldots, b_k)$ *is a basis of the lattice $L$.*

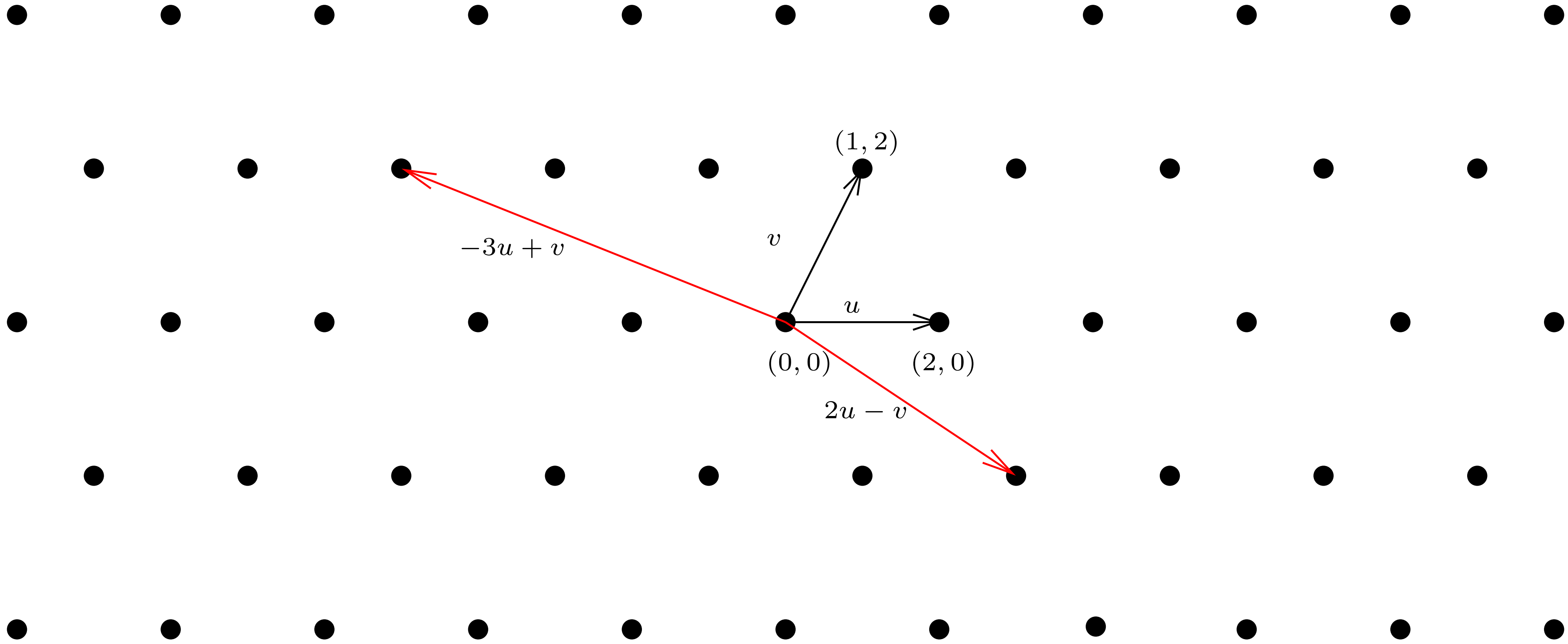
**Examples.** $\mathbb{Z}^d$, every subgroup of $\mathbb{Z}^d$.

**Remark .** *We say that a lattice $L$ is* integer *(resp.* rational*) when $L \in \mathbb{Z}^d$ (resp. $\mathbb{Q}^d$).*




The lattice $\mathbb{Z}(2,0) \oplus \mathbb{Z}(1,2)$.

**Proposition .** *If* $(e_1, \ldots, e_k)$ *and* $(f_1, \ldots, f_j)$ *are two free families that generate the same lattice, then* $k = j$ *(rank of the lattice) and there exists a* $k \times k$*-dimensional matrix* $M$*, with integer coefficients, and determinant equal to* $\pm 1$ *such that* $(e_i) = (f_i)M$*.*




The lattice $\mathbb{Z}(2,0) \oplus \mathbb{Z}(1,2)$.

**Proposition .** *If* $(e_1, \ldots, e_k)$ *and* $(f_1, \ldots, f_j)$ *are two free families that generate the same lattice, then* $k = j$ *(rank of the lattice) and there exists a* $k \times k$*-dimensional matrix* $M$*, with integer coefficients, and determinant equal to* $\pm 1$ *such that* $(e_i) = (f_i)M$*.*

There exists an infinity of bases (if $k \geq 2$) but some are more interesting than others.

Let $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d) \in \mathbb{R}^d$, then

$$(x|y) = x_1 y_1 + \cdots + x_d y_d.$$

We set $||x|| = (x|x)^{1/2} = (x_1^2 + \cdots + x_d^2)^{1/2}$ and $||x||_\infty = \max_{1 \le i \le d} |x_i|$.

There are several notions of what a “good” basis is but most of the time, it is required that it is made of short vectors.

# Shortest vector problem

**Problem .** **(SVP)** *Given a basis of a rational lattice $L$, find a shortest nonzero vector of $L$.*

*Associated approximation problem: find $v \in L \setminus \{0\}$ s.t. $||v|| \leq \gamma\lambda_1(L)$ where $\gamma \in \mathbb{R}$ is fixed and $\lambda_1(L)$ denotes the norm of a shortest nonzero vector of $L$.*

**Theorem .** *[Ajtai (1997), Miccianco (1998)] The problem of finding a vector $v$ s.t. $\|v\| = \lambda_1(L)$ is NP-hard under randomized polynomial reductions, and remains NP-hard if we tolerate an approximation factor $< \sqrt{2}$.*

# Closest vector problem

**Problem . (CVP)** *Given a basis of a rational lattice* $L$ *and* $x \in \mathbb{R}^d$, *find* $y \in L$ *s.t.* $||x - y|| = \mathrm{dist}(x, L)$.

*Associated approximation problem: find* $y \in L \setminus \{0\}$ *s.t.* $||x - y|| \leq \gamma \, \mathrm{dist}(x, L)$ *where* $\gamma \in \mathbb{R}$ *is fixed.*

Emde Boas (1981) : CVP is NP-hard

# Lenstra-Lenstra-Lovász algorithm

*Factoring Polynomials with Rational Coefficients*, A. K. Lenstra, H. W. Lenstra and L. Lovász, Math. Annalen **261**, 515-534, 1982.

**Theorem .** *Let $L$ a lattice of rank $k$.*

*LLL provides a basis $(b_1, \ldots, b_k)$ made of "pretty" short vectors. We have $||b_1|| \leq 2^{(k-1)/2}\lambda_1(L)$ where $\lambda_1(L)$ denotes the norm of a shortest nonzero vector of $L$.*

*LLL terminates in at most $O(k^6 \ln^3 B)$ operations with $B \geq ||b_i||^2$ for all $i$.*

**Remark .** *In practice, the returned basis is of better quality and given faster than expected.*

# Absolute error problem

We search for (one of the) best(s) polynomial of the form

$$p^\star = \frac{a_0^\star}{2^{m_0}} + \frac{a_1^\star}{2^{m_1}} X + \cdots + \frac{a_n^\star}{2^{m_n}} X^n$$

(where $a_i^\star \in \mathbb{Z}$ and $m_i \in \mathbb{Z}$ ) that minimizes $\|f - p\|_{[a,\, b]}$.

Discretize the continuous problem: we choose $x_1, \cdots, x_d$ points in $[a,\, b]$ such that $\frac{a_0^\star}{2^{m_0}} + \frac{a_1^\star}{2^{m_1}} x_i + \cdots + \frac{a_n^\star}{2^{m_n}} x_i^n$ as close as possible to $f(x_i)$ for all $i = 1, \ldots, d$.

That is to say we want the vectors

$$\begin{pmatrix} \frac{a_0^\star}{2^{m_0}} + \frac{a_1^\star}{2^{m_1}} x_1 + \cdots + \frac{a_n^\star}{2^{m_n}} x_1^n \\ \frac{a_0^\star}{2^{m_0}} + \frac{a_1^\star}{2^{m_1}} x_2 + \cdots + \frac{a_n^\star}{2^{m_n}} x_2^n \\ \vdots \\ \frac{a_0^\star}{2^{m_0}} + \frac{a_1^\star}{2^{m_1}} x_d + \cdots + \frac{a_n^\star}{2^{m_n}} x_d^n \end{pmatrix} \text{ and } \begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_d) \end{pmatrix}$$

to be as close as possible, which can be rewritten as: we want the vectors

$$a_0^\star \underbrace{\begin{pmatrix} \frac{1}{2^{m_0}} \\ \frac{1}{2^{m_0}} \\ \vdots \\ \frac{1}{2^{m_0}} \end{pmatrix}}_{\overrightarrow{v_0}} + a_1^\star \underbrace{\begin{pmatrix} \frac{x_1}{2^{m_1}} \\ \frac{x_2}{2^{m_1}} \\ \vdots \\ \frac{x_d}{2^{m_1}} \end{pmatrix}}_{\overrightarrow{v_1}} + \cdots + a_n^\star \underbrace{\begin{pmatrix} \frac{x_1^n}{2^{m_n}} \\ \frac{x_2^n}{2^{m_n}} \\ \vdots \\ \frac{x_d^n}{2^{m_n}} \end{pmatrix}}_{\overrightarrow{v_n}} \text{ and } \underbrace{\begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_d) \end{pmatrix}}_{\overrightarrow{y}}$$

to be as close as possible.

We have to minimize $\|a_0^\star \overrightarrow{v_0} + \cdots + a_n^\star \overrightarrow{v_n} - \overrightarrow{y}\|$.

We have to minimize $\|a_0^\star\overrightarrow{v_0} + \cdots + a_n^\star\overrightarrow{v_n} - \overrightarrow{y}\|$.

This is a closest vector problem in a lattice !

It is NP-hard : LLL algorithm gives an approximate solution.

# Focus on the method

We search for (one of the) best(s) polynomial of the form

$$p^\star = \frac{a_0^\star}{2^{m_0}} + \frac{a_1^\star}{2^{m_1}} X + \cdots + \frac{a_n^\star}{2^{m_n}} X^n$$

(where $a_i^\star \in \mathbb{Z}$ and $m_i \in \mathbb{Z}$ ) that minimizes $\|f - p\|_{[a,\, b]}$.

Choose $d$ points in $[a,\, b]$ : $x_1, \cdots, x_d$.

Our problem is to have $\frac{a_0^\star}{2^{m_0}} + \frac{a_1^\star}{2^{m_1}} x_i + \cdots + \frac{a_n^\star}{2^{m_n}} x_i^n$ as close as possible to $f(x_i)$ for all $i = 1, \ldots, d$.

Here again, the choice of the points is critical (it relies on some preliminary computations: linear programming and best polynomial approximation computation).

# Applying our method to Intel's erf code

erf is defined by $\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} \mathrm{d}t$ for all $x \in \mathbb{R}$.

- We looked at Intel's erf code on the interval $[1; 2]$ : it uses an argument reduction and the final problem is to approximate $\text{erf}(x + 1)$ on $[0; 1]$ with a polynomial to obtain an accuracy of $64$ bits.

- Intel uses a polynomial of degree $19$ with $20$ extended-double coefficients.

# How we can improve it

- We can't use a smaller degree because even the minimax polynomial of degree $18$ doesn't provide a sufficient accuracy. But we can reduce the size of the coefficients.

- We search for polynomials using the most possible number of double coefficients.

# Result

We get, almost instantaneously, a polynomial approximant

- with only two extended-double coefficients,
- that provides the same accuracy as the one with $20$ extended-double coefficients, currently used in Intel's code.
- This leads to smaller tables, faster cache loading time.

# Summary

- We've just seen that our method is able to give us a smaller (in term of degree and/or size of the coefficients) polynomial providing the same accuracy.

- But we can also use it to find a much better polynomial (in term of accuracy) with same precision for the coefficients than the rounded minimax.

- Let's look at an example from CRLibm.

# An example from CRlibm

- CRlibm is a library designed to compute correctly rounded functions in an efficient way (target : IEEE double precision).

  `http://lipforge.ens-lyon.fr/www/crlibm/`

- It uses specific formats such as double-double or triple-double.

- Here is an example we worked on with C. Lauter, and which is used to compute $\arcsin(x)$ on $[0.79; 1]$.

# Arcsine function

- After argument reduction we have the problem to approximate

$$g(z) = \frac{\arcsin(1-(z+m)) - \frac{\pi}{2}}{\sqrt{2\cdot(z+m)}}$$

where $0\text{xBFBC28F800009107} \le z \le 0\text{x3FBC28F7FFFF6EF1}$ (i.e. approximately $-0.110 \le z \le 0.110$) and $m = 0\text{x3FBC28F80000910F} \simeq 0.110$.

# Datas

Target accuracy to achieve correct rounding : $2^{-119}$.

The minimax of degree $21$ is sufficient (error = $2^{-119.83}$).

Each approximant is of the form

$$\underbrace{p_0}_{t.d.} + \underbrace{p_1}_{t.d.} x + \underbrace{p_2}_{d.d.} x^2 + \underbrace{\cdots}_{\cdots} + \underbrace{p_9}_{d.d.} x^9 + \underbrace{p_{10}}_{d.} x^{10} + \underbrace{\cdots}_{\cdots} + \underbrace{p_{21}}_{d.} x^{21}$$

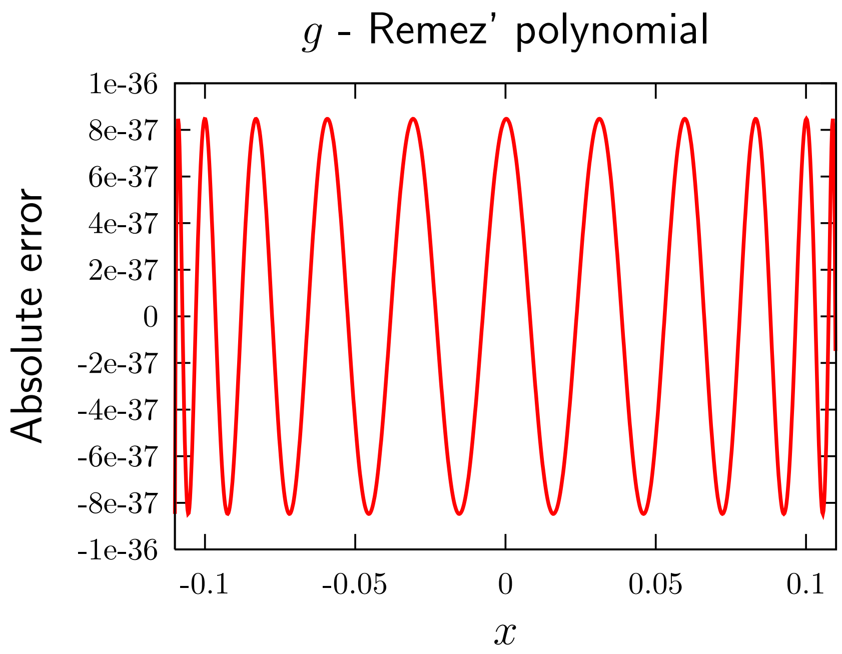
where the $p_i$ are either double precision numbers (d.), a sum of two double precision numbers (d.d.), a sum of two double precision numbers (t.d.).

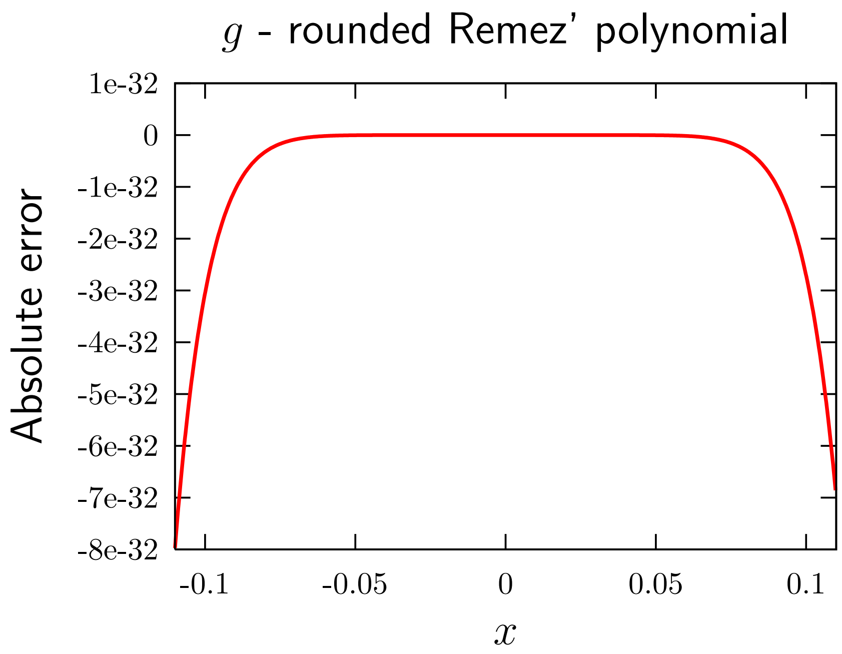
Figure 1: binary logarithm of the absolute error of several approximants

| | |
|---|---|
| Target | -119 |
| Minimax | -119.83 |
| Rounded minimax | -103.31 |
| Our polynomial | -119.77 |

# Exact minimax, rounded minimax, our polynomial

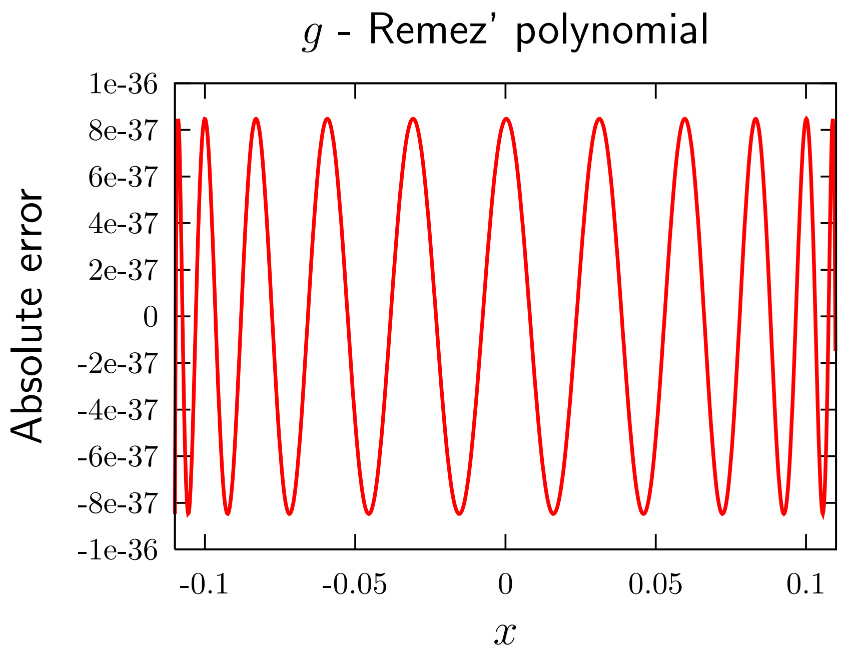
We save 16 bits with our method.

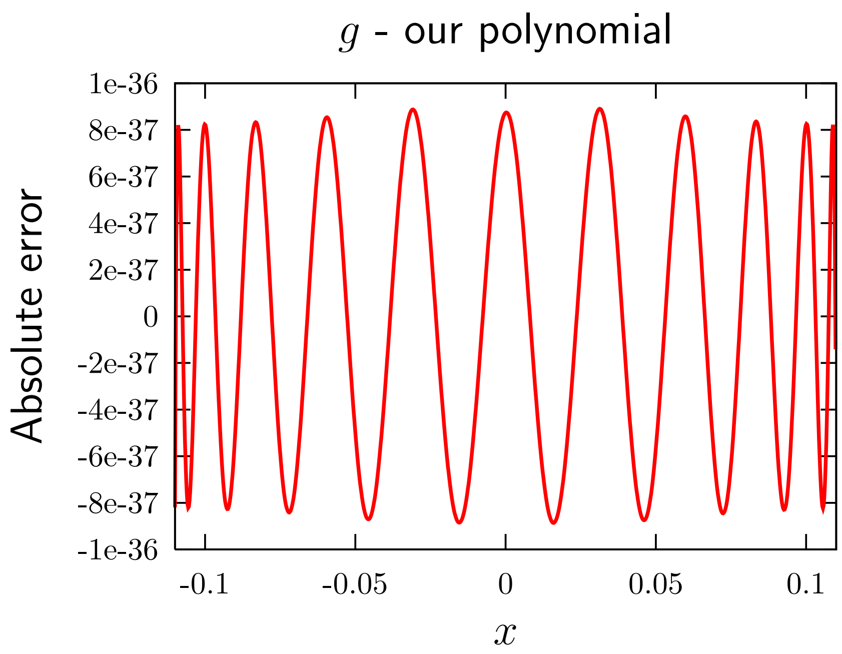


# Exact minimax, rounded minimax, our polynomial

We save $16$ bits with our method.



# Conclusion

- Two methods which improve the results provided by existing Remez' based method.

  The first method, based on linear programming, gives a best polynomial possible (for a given sequence of $m_i$).

  The second method, based on lattice basis reduction, much faster and more efficient than the first one, gives a very good approximant. We use linear programming to show that the error provided by this approach is tight.

  All these tools are or shall be part of the free software Sollya `http://sollya.gforge.inria.fr/`. Sollya is a tool environment for safe floating-point code development.

- Can be adapted to several kind of coefficients (fixed-point format, multi-double, classical floating point arithmetic with several precision formats).