Lattices in Computer Arithmetic

AriC team (previously Arénaire team)
Nicolas Brisebarre

École de Printemps d'Informatique Théorique
Autrans, March 17-22, 2013

Contributors

This talk presents some works by:
Dan Boneh, N.B., Thomas Caissard, Sylvain Chevillard, Glenn Durfee, Silviu Filip, Mourad Gouicem, Guillaume Hanrot, William Kahan, Mioara Joldeș, Christoph Lauter, Vincent Lefèvre, Jean-Michel Muller, Damien Stehlé, Arnaud Tisserand, Serge Torres, Paul Zimmermann.

Computing

- Huge focus on “High Performance Computing”

Computing

- Huge focus on “High Performance Computing”
- Security issues regarding the data, the computations, the results: Cryptology

Computing

- Huge focus on "High Performance Computing"
- Security issues regarding the data, the computations, the results: Cryptology
- What about the accuracy and the reliability of these computations?

Floating Point (FP) Arithmetic

Given

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

A finite FP number x is represented by 2 integers:

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

such that

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

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

IEEE Precisions

See http://en.wikipedia.org/wiki/IEEE_floating_point or (older) http://babbage.cs.qc.edu/courses/cs341/IEEE-754references.html.

	precision	minimal exponent	maximal exponent
single (binary 32)	24	-126	127
double (binary 64)	53	-1022	1023
extended double	64	-16382	16383
quadruple (binary 128)	113	-16382	16383

Various bugs

McCullough, B. D. and Vinod, H. D. The numerical reliability of econometric software. J. Economic Lit. 37, pp. 633-665, June 1999.

> *Improper attention to the method of rounding can produce disastrous results. The Wall Street Journal (November 8, 1983, p. 37) reported on the Vancouver Stock Exchange, which created an index much like the Dow-Jones Index. It began with a nominal value of 1,000.000 and was recalculated after each recorded transaction by calculation to four decimal places, the last place being truncated so that three decimal places were reported. Truncating the fourth decimal of a number measured to approximately 103 might seem innocuous. Yet, within a few months the index had fallen to 520, while there was no general downturn in economic activity. The problem, of course, was insufficient attention given to the method of rounding. When recalculated properly, the index was found to be 1098.892 (Toronto Star, November 29, 1983).*

Various bugs

S. Rump's example (1988). Consider

$$f(a,b) = 333.75b^6 + a^2\left(11a^2b^2 - b^6 - 121b^4 - 2\right) + 5.5b^8 + \frac{a}{2b},$$

and try to compute $f(a,b)$ for $a = 77617.0$ and $b = 33096.0$. On an IBM 370 computer:

- 1.172603 in single precision;
- 1.1726039400531 in double precision; and
- 1.172603940053178 in extended precision.

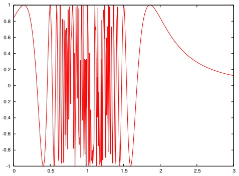
And yet, the exact result is $-0.8273960599\cdots$.
What about more recent systems? On a Pentium4 (gcc, Linux), Rump's C program returns

- 5.960604×10^{20} in single precision;
- 2.0317×10^{29} in double precision;
- $-9.38724 \times 10^{-323}$ in extended precision.

Various bugs

M. Joldeș. Rigorous Polynomial Approximations and Applications. PhD thesis, ENS Lyon, 2011.

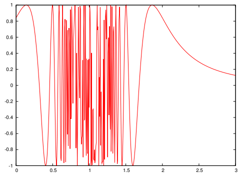
Let $J = \int_0^3 \sin\left(\frac{1}{(10^{-3}+(1-x)^2)^{3/2}}\right) \mathrm{d}x$.



Various bugs

M. Joldeș. Rigorous Polynomial Approximations and Applications. PhD thesis, ENS Lyon, 2011.

Let $J = \int_0^3 \sin\left(\frac{1}{(10^{-3} + (1-x)^2)^{3/2}}\right) \mathrm{d}x$.



- Maple15: 0.7499743685;
- Pari/GP: 0.7927730971479080755;
- Mathematica, Chebfun fail to answer;
- Chen, '06: 0.7578918118.

  WHAT IS THE CORRECT ANSWER?

Various bugs

W. Tucker. Validated Numerics. Princeton University Press, 2011.

Let $I = \int_0^8 \sin(x + e^x)\mathrm{d}x$. Let's evaluate it using MATLAB.

```
fcn_str = 'sin(x+exp(x))';
f = vectorize(inline(fcn_str));
a = 0; b = 8;
>> q = quad(f,a,b)
q =
    0.251102722027180
```

Actually, $I \in [0.3474, 0.3475]$...

A statement by Alston Householder

“It makes me nervous to fly on airplanes since I know they are designed using floating-point arithmetic.” A. Householder

A statement by Alston Householder

“It makes me nervous to fly on airplanes since I know they are designed using floating-point arithmetic.” A. Householder

Well, the situation is not completely tragic! There are some useful computations that we can perform in a fast and certified way.

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f.

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f.
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f.
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 0. Computation of hardest-to-round cases: V. Lefèvre and J.-M. Muller.
- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f.
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Applications

- specific hardware implementations in low precision (~ 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).

Scientific Framework and Tools

Computer Arithmetic

Scientific Framework and Tools

Computer Arithmetic

- Numerical Analysis, Approximation Theory, Interval Analysis, Fine-tuned Implementation

Scientific Framework and Tools

Computer Arithmetic

- Numerical Analysis, Approximation Theory, Interval Analysis, Fine-tuned Implementation
- Algorithmic Number Theory, Computer Algebra, Functional Analysis, Complex Analysis, Logic, Formal Proof

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 0. Computation of hardest-to-round cases: V. Lefèvre and J.-M. Muller.
- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f.
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.

Argument reduction

- In order to evaluate $\varphi(x)$ with x a floating-point number, we transform x into x^*, the reduced argument.
- x^* belongs to the convergence domain of an elementary function f.
- We know how to get $\varphi(x)$ from $f(x^*)$.

Worst Cases for Argument Reduction

W. Kahan. "Minimizing $q * m - n$", available at
`http://www.cs.berkeley.edu/~wkahan/testpi/`, text at the beginning of nearpi.c

Example: The cos function (additive argument reduction)

The goal is to evaluate $\cos$. The convergence domain of the evaluation algorithm of $\sin$ and $\cos$ contains $[-\pi/4, +\pi/4]$.

Example: The cos function (additive argument reduction)

The goal is to evaluate $\cos$. The convergence domain of the evaluation algorithm of $\sin$ and $\cos$ contains $[-\pi/4, +\pi/4]$. We decompose the computation of $\cos(x)$ into three steps:

- we compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - k\pi/2$;

Example: The cos function (additive argument reduction)

The goal is to evaluate $\cos$. The convergence domain of the evaluation algorithm of $\sin$ and $\cos$ contains $[-\pi/4, +\pi/4]$. We decompose the computation of $\cos(x)$ into three steps:

- we compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - k\pi/2$;
- we compute $g(x^*, k) =$

$$\begin{cases} \cos(x^*) & \text{if} \quad k \text{ mod } 4 = 0, \\ -\sin(x^*) & \text{if} \quad k \text{ mod } 4 = 1, \\ -\cos(x^*) & \text{if} \quad k \text{ mod } 4 = 2, \\ \sin(x^*) & \text{if} \quad k \text{ mod } 4 = 3; \end{cases}$$

Example: The cos function (additive argument reduction)

The goal is to evaluate $\cos$. The convergence domain of the evaluation algorithm of $\sin$ and $\cos$ contains $[-\pi/4, +\pi/4]$. We decompose the computation of $\cos(x)$ into three steps:

- we compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - k\pi/2$;
- we compute $g(x^*, k) =$

$$\begin{cases} \cos(x^*) & \text{if} \quad k \text{ mod } 4 = 0, \\ -\sin(x^*) & \text{if} \quad k \text{ mod } 4 = 1, \\ -\cos(x^*) & \text{if} \quad k \text{ mod } 4 = 2, \\ \sin(x^*) & \text{if} \quad k \text{ mod } 4 = 3; \end{cases}$$

- we obtain $\cos(x) = g(x^*, k)$.

Argument reduction: loss of accuracy problems

Reminder. We compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - k\pi/2$.

Argument reduction: loss of accuracy problems

Reminder. We compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - k\pi/2$.
Extreme caution required!!! We can encounter terrible loss of accuracy problems.

Argument reduction: loss of accuracy problems

Reminder. We compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - k\pi/2$.
Extreme caution required!!! We can encounter terrible loss of accuracy problems.

Naive method: compute

$$\begin{aligned} k &= \left\lfloor \frac{x}{\pi/2} \right\rfloor, \\ x^* &= x - k\pi/2, \end{aligned}$$

using machine precision.

Argument reduction: loss of accuracy problems

Reminder. We compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - k\pi/2$.
Extreme caution required!!! We can encounter terrible loss of accuracy problems.

Naive method: compute

$$\begin{aligned} k &= \left\lfloor \frac{x}{\pi/2} \right\rfloor, \\ x^* &= x - k\pi/2, \end{aligned}$$

using machine precision.

Problem: when $k\pi/2$ near x, almost all (or all) the accuracy is lost when computing $x - k\pi/2$.

Argument Reduction: loss of accuracy problems

Example : if $x = 8248.251512$, right value of $x^* = -2.14758367 \cdots \times 10^{-12}$, and $k = 5251$.

Argument Reduction: loss of accuracy problems

Example : if $x = 8248.251512$, right value of $x^* = -2.14758367 \cdots \times 10^{-12}$, and $k = 5251$.

Computation of $x - k\pi/2$ on a pocket calculator (10 digits) using rounding to nearest mode and a rounding of $\pi/2$. We get -1.0×10^{-6}.

Argument Reduction: loss of accuracy problems

Example : if $x = 8248.251512$, right value of $x^* = -2.14758367 \cdots \times 10^{-12}$, and $k = 5251$.

Computation of $x - k\pi/2$ on a pocket calculator (10 digits) using rounding to nearest mode and a rounding of $\pi/2$. We get -1.0×10^{-6}.

Conclusion: the computed value of $\cos(x)$ is -1.0×10^{-6}, but the correct value is $-2.14758367 \cdots \times 10^{-12}$.

Argument Reduction: loss of accuracy problems

First solution: multiple precision. Problems: slow, difficult to know the necessary accuracy beforehand.

Argument Reduction: loss of accuracy problems

First solution: multiple precision. Problems: slow, difficult to know the necessary accuracy beforehand.

Second solution: computing the worst cases, i.e. those which are the hardest to round. Method due to W. Kahan.

Rational Approximation

Let $p/q \in \mathbb{Q}$, we define the height of p/q by $h(p/q) = \max(|p|, |q|)$.

Property

$\mathbb{Q}$ is dense in $\mathbb{R}$: for all $x \in \mathbb{R}$, for all $\varepsilon > 0$, there exists $p/q \in \mathbb{Q}$ s.t.

$$\left|x - \frac{p}{q}\right| < \varepsilon.$$

Rational Approximation

Let $p/q \in \mathbb{Q}$, we define the height of p/q by $h(p/q) = \max(|p|, |q|)$.

Property

$\mathbb{Q}$ is dense in $\mathbb{R}$: for all $x \in \mathbb{R}$, for all $\varepsilon > 0$, there exists $p/q \in \mathbb{Q}$ s.t.

$$\left| x - \frac{p}{q} \right| < \varepsilon.$$

Not precise enough to be useful to us: we'd rather have $h(p, q)$ as small as possible.

Rational Approximation

Let $p/q \in \mathbb{Q}$, we define the height of p/q by $h(p/q) = \max(|p|, |q|)$.

Property

$\mathbb{Q}$ *is dense in* $\mathbb{R}$ *: for all* $x \in \mathbb{R}$*, for all* $\varepsilon > 0$*, there exists* $p/q \in \mathbb{Q}$ *s.t.*

$$\left| x - \frac{p}{q} \right| < \varepsilon.$$

Not precise enough to be useful to us: we'd rather have $h(p, q)$ as small as possible.
More precisely: given $x \in \mathbb{R}$ and $\varepsilon > 0$, we want to have $|x - p/q| < \varepsilon$ or $|qx - p| < \varepsilon$ with $p \in \mathbb{Z}$, $q \in \mathbb{N} \setminus \{0\}$ as small as possible.

Notations

Let $x \in \mathbb{R}$. We denote $\lfloor x \rfloor$ the floor part of x, and $\|x\| = \min_{n \in \mathbb{Z}} |x - n|$ the distance of x to the nearest integer.

Notations

Let $x \in \mathbb{R}$. We denote $\lfloor x \rfloor$ the floor part of x, and $\|x\| = \min_{n \in \mathbb{Z}} |x - n|$ the distance of x to the nearest integer.

We want to have $|x - p/q|$ or rather $|qx - p|$ small with $p \in \mathbb{Z}$, $q \in \mathbb{N} \setminus \{0\}$ as small as possible.

Notations

Let $x \in \mathbb{R}$. We denote $\lfloor x \rfloor$ the floor part of x, and $\|x\| = \min_{n\in\mathbb{Z}} |x - n|$ the distance of x to the nearest integer.

We want to have $|x - p/q|$ or rather $|qx - p|$ small with $p \in \mathbb{Z}$, $q \in \mathbb{N} \setminus \{0\}$ as small as possible.

We consider $\|qx\|$ for $q \in \mathbb{N}^*$ in the sequel.

Best Approximations to a Real Number

Let $x \in \mathbb{R}$.

Definition

A fraction p/q ($p \in \mathbb{Z}$ and $q \in \mathbb{N}$, $q \neq 0$) is a best approximation to x if and only if

$$\begin{cases} \|qx\| = |qx - p| & \text{and} \\ \|q'x\| > \|qx\| & \text{for } 0 < q' < q. \end{cases}$$

Best Approximations to a Real Number

Theorem (Dirichlet)

Let $x \in \mathbb{R}$ and $Q \in]1, +\infty[$. Then there exists $q \in \mathbb{N} \cap]0, Q[$ s. t. $\|qx\| \leq Q^{-1}$.

Best Approximations to a Real Number

Theorem (Dirichlet)

Let $x \in \mathbb{R}$ and $Q \in]1, +\infty[$. Then there exists $q \in \mathbb{N} \cap]0, Q[$ s. t. $\|qx\| \leq Q^{-1}$.

We set $q_1 = 1$. Let p_1 s.t.. $\|q_1 x\| = \|x\| = |x - p_1|$. p_1/q_1 is a best rational approximation to x.

Best Approximations to a Real Number

Theorem (Dirichlet)

Let $x \in \mathbb{R}$ and $Q \in]1, +\infty[$. Then there exists $q \in \mathbb{N} \cap]0, Q[$ s. t. $\|qx\| \leq Q^{-1}$.

We set $q_1 = 1$. Let p_1 s.t.. $\|q_1 x\| = \|x\| = |x - p_1|$. p_1/q_1 is a best rational approximation to x.

If $\|q_1 x\| = 0$, $x \in \mathbb{Z}$ and p_1/q_1 is its only one best approx.

Best Approximations to a Real Number

Theorem (Dirichlet)

Let $x \in \mathbb{R}$ and $Q \in]1, +\infty[$. Then there exists $q \in \mathbb{N} \cap]0, Q[$ s. t. $\|qx\| \leq Q^{-1}$.

We set $q_1 = 1$. Let p_1 s.t.. $\|q_1 x\| = \|x\| = |x - p_1|$. p_1/q_1 is a best rational approximation to x.

If $\|q_1 x\| = 0$, $x \in \mathbb{Z}$ and p_1/q_1 is its only one best approx.

If $\|q_1 x\| > 0$, Dirichlet Theorem with $Q > \|q_1 x\|^{-1} \Rightarrow$ there exists $q \in \mathbb{N}$ s.t.. $\|qx\| \leq Q^{-1} < \|q_1 x\|$.

Best Approximations to a Real Number

Theorem (Dirichlet)

Let $x \in \mathbb{R}$ and $Q \in]1, +\infty[$. Then there exists $q \in \mathbb{N} \cap]0, Q[$ s. t. $\|qx\| \leq Q^{-1}$.

We set $q_1 = 1$. Let p_1 s.t.. $\|q_1 x\| = \|x\| = |x - p_1|$. p_1/q_1 is a best rational approximation to x.

If $\|q_1 x\| = 0$, $x \in \mathbb{Z}$ and p_1/q_1 is its only one best approx.

If $\|q_1 x\| > 0$, Dirichlet Theorem with $Q > \|q_1 x\|^{-1} \Rightarrow$ there exists $q \in \mathbb{N}$ s.t.. $\|qx\| \leq Q^{-1} < \|q_1 x\|$.

Let q_2 be the smallest integer q s.t. $\|qx\| < \|q_1 x\|$. Let p_2 s.t. $\|q_2 x\| = |q_2 x - p_2|$.

Best Approximations to a Real Number

Theorem (Dirichlet)

Let $x \in \mathbb{R}$ and $Q \in]1, +\infty[$. Then there exists $q \in \mathbb{N} \cap]0, Q[$ s. t. $\|qx\| \leq Q^{-1}$.

We set $q_1 = 1$. Let p_1 s.t.. $\|q_1 x\| = \|x\| = |x - p_1|$. p_1/q_1 is a best rational approximation to x.

If $\|q_1 x\| = 0$, $x \in \mathbb{Z}$ and p_1/q_1 is its only one best approx.

If $\|q_1 x\| > 0$, Dirichlet Theorem with $Q > \|q_1 x\|^{-1}$ $\Rightarrow$ there exists $q \in \mathbb{N}$ s.t.. $\|qx\| \leq Q^{-1} < \|q_1 x\|$.

Let q_2 be the smallest integer q s.t. $\|qx\| < \|q_1 x\|$. Let p_2 s.t. $\|q_2 x\| = |q_2 x - p_2|$.

By construction, p_2/q_2 is a best rational approximation to x, with the smallest denominator greater than q_1.

Best Approximations to a Real Number

We iterate this process. We get, by induction, a strictly increasing sequence (finite if $x \in \mathbb{Q}$) of integers $1 = q_1 < q_2 < \cdots$ and a sequence of integers $p_1, p_2, \ldots$ s. t. :

$$\|q_n x\| = |q_n x - p_n|\,, \tag{1}$$

$$\|q_{n+1} x\| < \|q_n x\|\,, \tag{2}$$

$$\|q x\| \geq \|q_n x\| \quad \text{pour } 0 < q < q_{n+1}. \tag{3}$$

From (1-3), the p_n/q_n are best approximations to x.

Best Approximations to a Real Number

By construction, p_1/q_1 is a best approximation, and p_{n+1}/q_{n+1} is the best approximation of x with the smallest denominator greater than q_n. We proved

Theorem

The p_n/q_n are the best approximations to x, sorted by increasing size of the denominators.

If x is a rational number

If $x \in \mathbb{Q}$ i.e. $x = a/b$ with $a \in \mathbb{Z}$, $b \in \mathbb{N}^*$ and $\mathrm{pgcd}(a,b) = 1$, then a/b is a best approximation to x.

There exists $N \in \mathbb{N}$ s.t. $x = p_N/q_N$ and, as $\|q_N x\| = 0$, the process stops at the order N : the number of best approximations is finite.

If x is an irrational number

If $x \notin \mathbb{Q}$, then the sequence p_n/q_n converges to x. From the previous results, we get

$$q_n \|q_n x\| < q_{n+1} \|q_n x\| \leq 1\,.$$

If x is an irrational number

If $x \notin \mathbb{Q}$, then the sequence p_n/q_n converges to x. From the previous results, we get

$$q_n\|q_n x\| < q_{n+1}\|q_n x\| \le 1\,.$$

Hence, we have

$$\left|x - \frac{p_n}{q_n}\right| = \frac{1}{q_n}\|q_n x\| \leqslant \frac{1}{q_n q_{n+1}} < \frac{1}{q_n^2}\,,$$

which shows $\lim_{n\to\infty} p_n/q_n = x$.

If x is an irrational number

If $x \notin \mathbb{Q}$, we have, for all $n \geqslant 0$,

$$\left| x - \frac{p_n}{q_n} \right| \leqslant \frac{1}{q_n q_{n+1}} \left(\leqslant \frac{1}{q_n^2} \right).$$

If x is an irrational number

If $x \notin \mathbb{Q}$, we have, for all $n \geqslant 0$,

$$\left| x - \frac{p_n}{q_n} \right| \leqslant \frac{1}{q_n q_{n+1}} \left(\leqslant \frac{1}{q_n^2} \right).$$

Note that we can also show, for all $n \geqslant 0$,

$$\frac{1}{q_n(q_n + q_{n+1})} \leqslant \left| x - \frac{p_n}{q_n} \right|.$$

An Algorithm for Computing Best Rational Approximations to x

Theorem

Let $x \in \mathbb{R}$. We define the sequences (finite or infinite) (p_n), (q_n) and (a_n) by the initial conditions $a_0 = \lfloor x \rfloor$, $(p_0, q_0, p_1, q_1) = (1, 0, \lfloor x \rfloor, 1)$ and the recurrence relations defined for $n \in \mathbb{N}^*$

$$\begin{cases} p_{n+1} = a_n p_n + p_{n-1}\,, \\ q_{n+1} = a_n q_n + q_{n-1}\,, \end{cases}$$

where

$$a_n = \left\lfloor \frac{|q_{n-1}x - p_{n-1}|}{|q_n x - p_n|} \right\rfloor$$

if $q_n x \neq p_n$ (the sequences are finite if $q_n x = p_n$). Then the p_n/q_n are best rational approximations to x for $n \geq 1$ if $a_1 \geq 2$, and for $n \geq 2$ if $a_1 = 1$.

An Algorithm for Computing Best Rational Approximations to x

Theorem

Let $x \in \mathbb{R}$. We define the sequences (finite or infinite) (p_n), (q_n) and (a_n) by the initial conditions $a_0 = \lfloor x \rfloor$, $(p_0, q_0, p_1, q_1) = (1, 0, \lfloor x \rfloor, 1)$ and the recurrence relations defined for $n \in \mathbb{N}^*$

$$\begin{cases} p_{n+1} = a_n p_n + p_{n-1} , \\ q_{n+1} = a_n q_n + q_{n-1} , \end{cases}$$

where

$$a_n = \left\lfloor \frac{|q_{n-1}x - p_{n-1}|}{|q_n x - p_n|} \right\rfloor$$

if $q_n x \neq p_n$ (the sequences are finite if $q_n x = p_n$). Then the p_n/q_n are best rational approximations to x for $n \geq 1$ if $a_1 \geq 2$, and for $n \geq 2$ if $a_1 = 1$.

If we put $r_0 = x$, $r_n = \dfrac{1}{r_{n-1} - a_{n-1}}$, we have $a_n = \lfloor r_n \rfloor$.

An Algorithm for Computing Best Rational Approximations to x

The fractions p_n/q_n are called the convergents and the a_n the partial quotients of the continued fraction expansion of x.

Note that if $x \in \mathbb{Q}$, it is exactly the Euclidean algorithm.

Since the knowledge of the sequence (a_n) is equivalent to the one of x ($x = p_N/q_N$ if $x \in \mathbb{Q}$ and $x = \lim_{n\to\infty} p_n/q_n$ if $x \notin \mathbb{Q}$), we will use the notation $x = [a_0; a_1, a_2, \ldots]$ (the number of terms between brackets can be finite).

“Continued Fraction”?

Let, for $n \in \mathbb{N}^*$,

$$x_n = \frac{|q_n x - p_n|}{|q_{n-1} x - p_{n-1}|},$$

so that we have

$$x_0 = x^{-1} \text{ and } x_n^{-1} = a_n + x_{n+1}.$$

So we get

$$x = \frac{1}{x_0} = a_0 + x_1 = a_0 + \cfrac{1}{a_1 + x_2} = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + x_3}} = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots}}}.$$

This writing explains the name “continued fraction".

Examples: When We're Lucky

We have

$$\sqrt{2} = 1 + \cfrac{1}{2 + \cfrac{1}{2 + \cfrac{1}{2 + \cfrac{1}{2 + \dots}}}}$$

which we can denote $\sqrt{2} = [1, \overline{2}]$.

Examples: When We're Lucky

We have

$$\sqrt{2} = 1 + \cfrac{1}{2 + \cfrac{1}{2 + \cfrac{1}{2 + \cfrac{1}{2 + \dots}}}}$$

which we can denote $\sqrt{2} = [1, \overline{2}]$. We also have $\sqrt{3} = [1, \overline{1, 2}]$, $\sqrt{5} = [2, \overline{4}]$, $\sqrt{7} = [2, \overline{1, 1, 1, 4}]$.
A quadratic number has a periodic continued fraction expansion: there exist k and $L \in \mathbb{N}$ s.t. $a_l = a_{l+k}$ for all $l \geqslant L$.

Euler : $e = [2, 1, 2, 1, 1, 4, 1, 1, 6, 1, \dots, 1, 2n, 1, \dots]$.

Examples: When We're Lucky

We have

$$\sqrt{2} = 1 + \cfrac{1}{2 + \cfrac{1}{2 + \cfrac{1}{2 + \cfrac{1}{2 + \dots}}}}$$

which we can denote $\sqrt{2} = [1, \overline{2}]$. We also have $\sqrt{3} = [1, \overline{1, 2}]$, $\sqrt{5} = [2, \overline{4}]$, $\sqrt{7} = [2, \overline{1, 1, 1, 4}]$.
A quadratic number has a periodic continued fraction expansion: there exist k and $L \in \mathbb{N}$ s.t. $a_l = a_{l+k}$ for all $l \geqslant L$.

Euler : $e = [2, 1, 2, 1, 1, 4, 1, 1, 6, 1, \dots, 1, 2n, 1, \dots]$.

It is fairly uncommon to encounter such closed expression for continued fraction expansions (c.f.e).

Examples

For instance, we don't know anything regarding the c.f.e. of $\pi = 3.14159265358979...$.

$$\pi = 3 + \cfrac{1}{7 + \cfrac{1}{15 + \cfrac{1}{1 + \cfrac{1}{292 + \dots,}}}}$$

Examples

For instance, we don't know anything regarding the c.f.e. of $\pi = 3.14159265358979\ldots$.

$$\pi = 3 + \cfrac{1}{7 + \cfrac{1}{15 + \cfrac{1}{1 + \cfrac{1}{292 + \ldots,}}}}$$

Its first convergents are $\frac{p_1}{q_1} = 3$, $\frac{p_2}{q_2} = \frac{22}{7} = 3.142\ldots$, $\frac{p_3}{q_3} = \frac{333}{106} = 3.14150\ldots$, $\frac{p_4}{q_4} = \frac{355}{113} = 3.1415929\ldots$, $\frac{p_5}{q_5} = \frac{103993}{33102} = 3.1415926530\ldots$.

Two classical results

Reminder. The best approximations p_n/q_n of a real number x satisfy $q_n|q_nx - p_n| < 1$.

Theorem

Among two consecutive best approximations to x, one at least satisfies the inequality $q\|qx\| < 1/2$.

Two classical results

Reminder. The best approximations p_n/q_n of a real number x satisfy $q_n|q_nx - p_n| < 1$.

Theorem

Among two consecutive best approximations to x, one at least satisfies the inequality $q\|qx\| < 1/2$.

And the reciprocal:

Theorem (Legendre)

If $q|qx - p| < 1/2$, then p/q is a convergent of (the continued fraction expansion of) x.

We still address the $\cos$ example with a reduction mod $\pi/2$. Let $C = \pi/2$.

Reminder. We compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - kC$.

We use radix 2, with mantissas over n bits and exponents between e_{min} and e_{max}.
Our x has the following form

$$x = x_0.x_1x_2x_3 \cdots x_{n-1} \times 2^E, \text{ with } x_0 \neq 0$$

or

$$x = M \times 2^{E-n+1},$$

with $M = x_0x_1x_2x_3 \cdots x_{n-1}$, $2^{n-1} \leqslant M \leqslant 2^n - 1.$

Reminder. We compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - kC$.

We search for $p \in \mathbb{Z}$ and $s \in \mathbb{R}$, $|s| < 1/2$ s.t. $\dfrac{x}{C} = p + s.$

Reminder. We compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - kC$.

We search for $p \in \mathbb{Z}$ and $s \in \mathbb{R}$, $|s| < 1/2$ s.t. $\dfrac{x}{C} = p + s.$
$\Rightarrow$ x^* is equal to sC.

Reminder. We compute x^* and k s. t. $x^* \in [-\pi/4, +\pi/4]$ and $x^* = x - kC$.

We search for $p \in \mathbb{Z}$ and $s \in \mathbb{R}$, $|s| < 1/2$ s.t. $\dfrac{x}{C} = p + s.$
$\Rightarrow$ x^* is equal to sC.
Remember that $x = M \times 2^{E-n+1}$. Therefore we search for

$$\frac{2^{E-n+1}}{C} = \frac{p}{M} + \frac{s}{M}.$$

x^* is very small: p/M is a very good approximation to $2^{E-n+1}/C$.

Let $(p_n/q_n)_{n\geqslant 0}$ the sequence of convergents of $\dfrac{2^{E-n+1}}{C}$.
Let $j = \max\{k \in \mathbb{N} \text{ t.q. } q_k \leqslant 2^n - 1\}$.
We have

$$\left|p - M\frac{2^{E-n+1}}{C}\right| \geqslant \left|p_j - q_j\frac{2^{E-n+1}}{C}\right|.$$

Let $(p_n/q_n)_{n\geqslant 0}$ the sequence of convergents of $\dfrac{2^{E-n+1}}{C}$.
Let $j = \max\{k \in \mathbb{N} \text{ t.q. } q_k \leqslant 2^n - 1\}$.
We have

$$\left|p - M\frac{2^{E-n+1}}{C}\right| \geqslant \left|p_j - q_j\frac{2^{E-n+1}}{C}\right|.$$

Therefore

$$\left|pC - M2^{E-n+1}\right| \geqslant \varepsilon_E := C\left|p_j - q_j\frac{2^{E-n+1}}{C}\right|.$$

Let $(p_n/q_n)_{n\geqslant 0}$ the sequence of convergents of $\dfrac{2^{E-n+1}}{C}$.
Let $j = \max\{k \in \mathbb{N} \text{ t.q. } q_k \leqslant 2^n - 1\}$.
We have

$$\left|p - M\frac{2^{E-n+1}}{C}\right| \geqslant \left|p_j - q_j\frac{2^{E-n+1}}{C}\right|.$$

Therefore

$$\left|pC - M2^{E-n+1}\right| \geqslant \varepsilon_E := C\left|p_j - q_j\frac{2^{E-n+1}}{C}\right|.$$

Let $\varepsilon = \min_{e_{\text{min}}\leqslant E\leqslant e_{\text{max}}} \varepsilon_E$. The number $-\log_2(\varepsilon)$ makes it possible to know the precision accuracy which is necessary to perform a safe reduction argument.

Worst Cases for the Additive Argument Reduction for Several Floating-Points Systems and Constants C

r	n	C	e_{max}	Worst case	$-\log_r(\varepsilon)$
2	24	$\pi/2$	127	$16367173 \times 2^{+72}$	29.2
2	24	$\ln(2)$	127	8885060×2^{-11}	31.6
10	10	$\pi/2$	99	$8248251512 \times 10^{-6}$	11.7
10	10	$\pi/4$	99	$4124125756 \times 10^{-6}$	11.9
10	10	$\ln(10)$	99	$7908257897 \times 10^{+30}$	11.7
2	53	$\pi/2$	1023	$6381956970095103 \times 2^{+797}$	60.9
2	53	$\ln(2)$	1023	$5261692873635770 \times 2^{+499}$	66.8
2	113	$\pi/2$	1024	$614799 \cdots 1953734 \times 2^{+797}$	122.79

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 0. Computation of hardest-to-round cases: V. Lefèvre and J.-M. Muller.
- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f.
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f.

Minimax Approximation

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

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

Minimax Approximation

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

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

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

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

Minimax Approximation

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

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

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 due to Remez gives p (`minimax` function in Maple, also available in Sollya `http://sollya.gforge.inria.fr/`).

Minimax Approximation

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

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

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 due to 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.

Truncated Polynomials

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

Truncated Polynomials

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

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

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

Truncated Polynomials

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

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

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

Question: find $p^\star \in \mathcal{P}_n^m$ which minimizes $||f - q||$, $q \in \mathcal{P}_n^m$.

Truncated Polynomials

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

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

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

Question: find $p^\star \in \mathcal{P}_n^m$ which minimizes $||f - q||$, $q \in \mathcal{P}_n^m$.

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

Truncated Polynomials

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

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

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

Question: find $p^\star \in \mathcal{P}_n^m$ which minimizes $||f - q||$, $q \in \mathcal{P}_n^m$.

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

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

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

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
$\varepsilon = ||\cos - p||_{[0,\pi/4]} = 0.0001135879\ldots$.

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

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

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 $\hat{\varepsilon} = ||\cos - \hat{p}||_{[0,\pi/4]} = 0.00069397\ldots$.

Approximation of the Function $\cos$ over $[0, \pi/4]$ by a Degree-3 Polynomial

Maple or Sollya computes a polynomial p which 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

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

is minimal.
The naive approach gives the polynomial $\hat{p}$ and
$\hat{\varepsilon} = ||\cos - \hat{p}||_{[0,\pi/4]} = 0.00069397...$

Approximation of the Function $\cos$ over $[0, \pi/4]$ by a Degree-3 Polynomial

Maple or Sollya computes a polynomial p which is $\sim$ the best approximant to $\cos$. We have $\varepsilon = ||\cos - p||_{[0,\pi/4]} = 0.0001135879\ldots$.
We look for $a_0, a_1, a_2, a_3 \in \mathbb{Z}$ such that

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

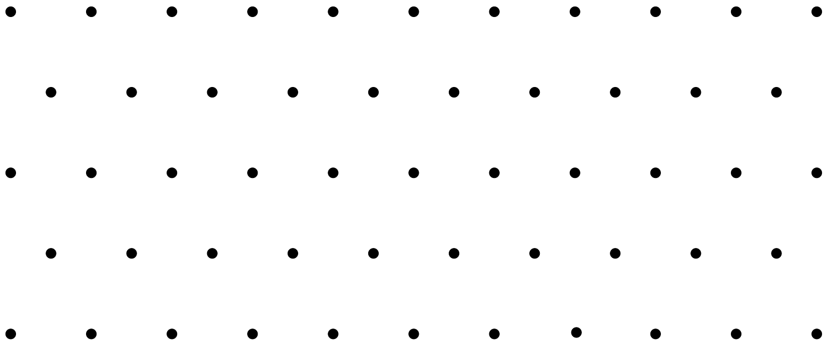
is minimal.
The naive approach gives the polynomial $\hat{p}$ and
$\hat{\varepsilon} = ||\cos - \hat{p}||_{[0,\pi/4]} = 0.00069397\ldots$ But the best "truncated" approximant:

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

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

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

An Approach based on Lattice Basis Reduction



An Approach based on Lattice Basis Reduction

Definition

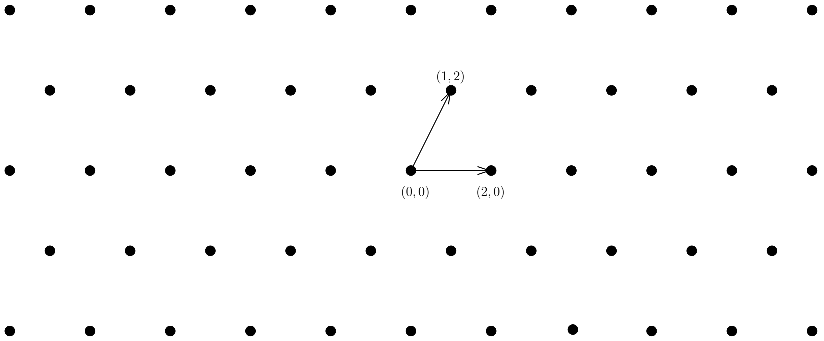
Let L be a nonempty subset of $\mathbb{R}^d$, L is a lattice iff there exists a set of vectors $b_1, \ldots, b_k$ $\mathbb{R}$-linearly independent such that

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

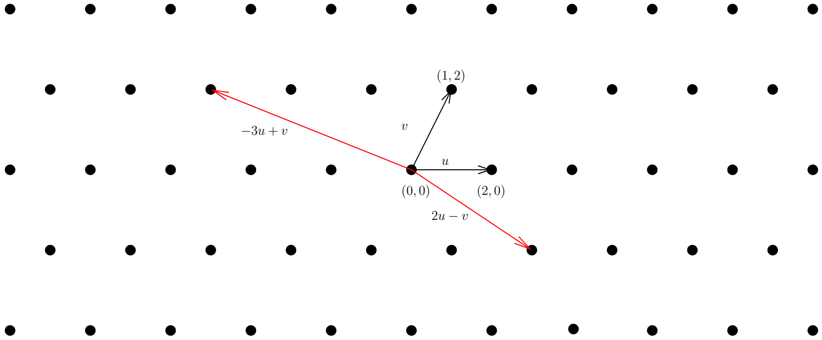
$(b_1, \ldots, b_k)$ is a basis of the lattice L.

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

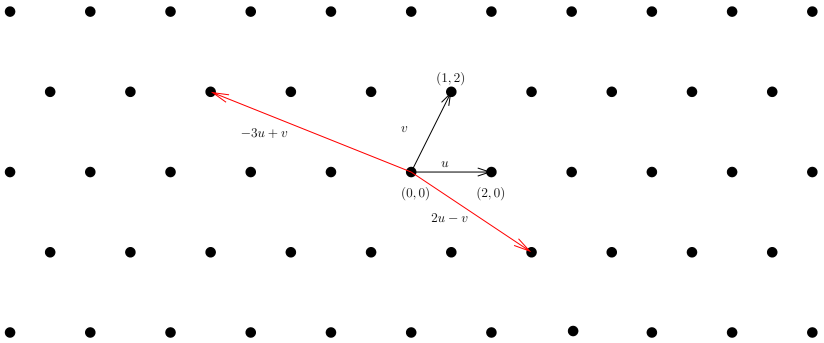
Example: The Lattice $\mathbb{Z}(2,0) \oplus \mathbb{Z}(1,2)$



Example: The Lattice $\mathbb{Z}(2,0) \oplus \mathbb{Z}(1,2)$

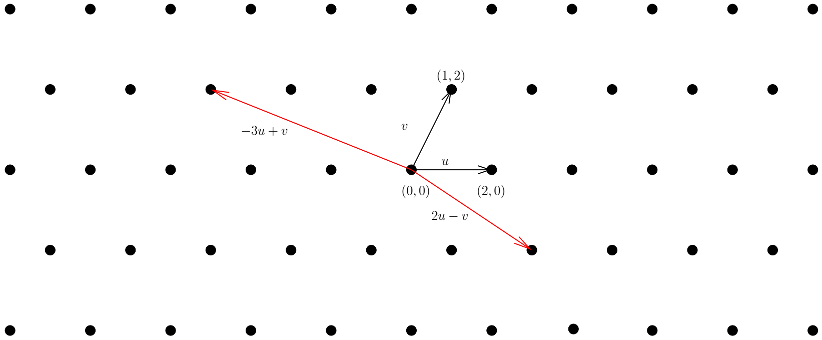


Example: The Lattice $\mathbb{Z}(2,0) \oplus \mathbb{Z}(1,2)$



SVP (Shortest Vector Problem) and CVP (Closest Vector Problem)

Example: The Lattice $\mathbb{Z}(2,0) \oplus \mathbb{Z}(1,2)$



SVP (Shortest Vector Problem) and CVP (Closest Vector Problem) are NP-hard.

Lenstra-Lenstra-Lovász Algorithm

SVP (Shortest Vector Problem) and CVP (Closest Vector Problem) are NP-hard.
Factoring Polynomials with Rational Coefficients, A. K. Lenstra, H. W. Lenstra and L. Lovász, Math. Annalen **261**, 515-534, 1982.

The LLL algorithm gives an approximate solution to SVP in polynomial time.

Babai's algorithm (based on LLL) gives an approximate solution to CVP in polynomial time.

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.

Summary

We can use the method we developed in two directions

- it is able to give us a smaller (in term of degree and/or size of the coefficients) polynomial providing the same accuracy.
- 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.

We illustrate the second item with an example taken 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 $\mathrm{0xBFBC28F800009107} \leqslant z \leqslant \mathrm{0x3FBC28F7FFFF6EF1}$ (i.e. approximately $-0.110 \leqslant z \leqslant 0.110$) and $m = \mathrm{0x3FBC28F80000910F} \simeq 0.110$.

Data

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

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

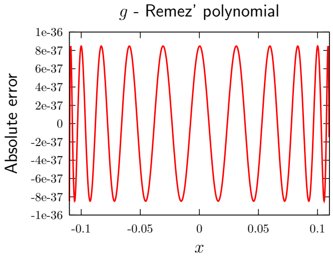
where the p_i are either double precision numbers (d.), a sum of two double precision numbers (d.d.), a sum of two double precision numbers (t.d.).

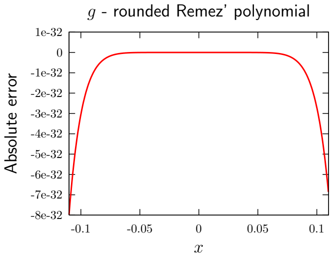
Figure: binary logarithm of the absolute error of several approximants

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

Exact Minimax, Rounded Minimax, our Polynomial

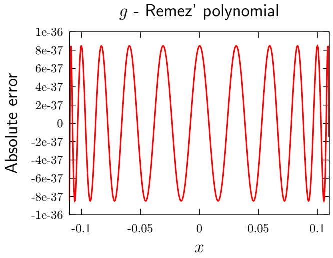
We save 16 bits with our method.

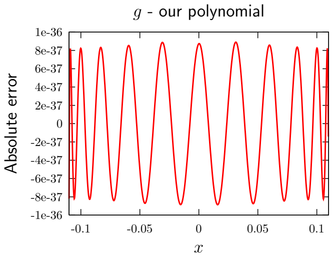




Exact Minimax, Rounded Minimax, our Polynomial

We save 16 bits with our method.





Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 0. Computation of hardest-to-round cases: V. Lefèvre and J.-M. Muller.
- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f .
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f .
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 0. Computation of hardest-to-round cases: V. Lefèvre and J.-M. Muller.
- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f .
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 0. Computation of hardest-to-round cases: V. Lefèvre and J.-M. Muller.

Binary Floating Point (FP) Arithmetic

Given

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

A finite FP number x is represented by 2 integers:

- integer mantissa : M, $2^{n-1} \leqslant |M| \leqslant 2^n - 1$;
- exponent E, $E_{\text{min}} \leqslant E \leqslant E_{\text{max}}$

such that

$$x = \frac{M}{2^{n-1}} \times 2^E.$$

IEEE Precisions

See http://en.wikipedia.org/wiki/IEEE_floating_point or (older) http://babbage.cs.qc.edu/courses/cs341/IEEE-754references.html.

	precision	minimal exponent	maximal exponent
single (binary 32)	24	-126	127
double (binary 64)	53	-1022	1023
extended double	64	-16382	16383
quadruple (binary 128)	113	-16382	16383

Correct rounding

In the IEEE 754 standard, the user defines an *active rounding mode* (or *rounding direction attribute*) among:

- round to the nearest (default). If $x \in \mathbb{R}$, $\mathsf{RN}(x)$ is the floating-point number that is the closest to x. In case of a tie, value whose integral significand is even.
- round towards $+\infty$.
- round towards $-\infty$.
- round towards zero.

Correct rounding

In the IEEE 754 standard, the user defines an *active rounding mode* (or *rounding direction attribute*) among:

- round to the nearest (default). If $x \in \mathbb{R}$, $\mathrm{RN}(x)$ is the floating-point number that is the closest to x. In case of a tie, value whose integral significand is even.
- round towards $+\infty$.
- round towards $-\infty$.
- round towards zero.

A correctly-rounded operation whose entries are FP numbers must return what we would get by infinitely precise operation followed by rounding.

Correct rounding

In the IEEE 754 standard, the user defines an *active rounding mode* (or *rounding direction attribute*) among:

- round to the nearest (default). If $x \in \mathbb{R}$, $\mathrm{RN}(x)$ is the floating-point number that is the closest to x. In case of a tie, value whose integral significand is even.
- round towards $+\infty$.
- round towards $-\infty$.
- round towards zero.

A correctly-rounded operation whose entries are FP numbers must return what we would get by infinitely precise operation followed by rounding.

In the sequel, we focus on rounding to nearest.

Correct rounding

IEEE-754 (1985): Correct rounding for $+$, $-$, $\times$, $\div$, $\sqrt{}$ and some conversions. Advantages:

- if the result of an operation is exactly representable, we get it;
- if we just use the 4 arith. operations and $\sqrt{}$, deterministic arithmetic: one can elaborate algorithms and proofs that use the specifications;
- accuracy and portability are improved;
- playing with rounding towards $+\infty$ and $-\infty$ $\rightarrow$ certain lower and upper bounds.

FP arithmetic becomes a structure in itself, that can be studied.
IEEE-754 (2008): suggests correct rounding for some elementary functions.

The Table Maker's Dilemma

Let $f : \mathbb{R} \to \mathbb{R}$ a function. Let x be a FP number. We want to compute $\mathsf{RN}(f(x))$.

Most of the time, we use an approximated value $\widetilde{f(x)}$ of $f(x)$. For instance, $f = \exp$ over $[0,1]$ and P is a polynomial approximant to f, then $\widetilde{f(x)} = P(x)$ for $x \in [0,1]$.

Question: how to make sure that $\mathsf{RN}(f(x)) = \mathsf{RN}\left(\widetilde{f(x)}\right)$?

The Table Maker's Dilemma

Let $f : \mathbb{R} \to \mathbb{R}$ a function. Let x be a FP number. We want to compute $\mathsf{RN}(f(x))$.

Most of the time, we use an approximated value $\widetilde{f(x)}$ of $f(x)$. For instance, $f = \exp$ over $[0,1]$ and P is a polynomial approximant to f, then $\widetilde{f(x)} = P(x)$ for $x \in [0,1]$.

Question: how to make sure that $\mathsf{RN}(f(x)) = \mathsf{RN}\left(\widetilde{f(x)}\right)$?

TMD: how to make sure that, for all FP number x,

$$\mathsf{RN}(f(x)) = \mathsf{RN}\left(\widetilde{f(x)}\right)?$$

The Table Maker's Dilemma

A breakpoint is a point where the rounding function changes.

Restatement of the question: determine $m_x \in \mathbb{N}$, s.t.

- $f(x) \in (\widetilde{f(x)} - 2^{-m_x}, \widetilde{f(x)} + 2^{-m_x})$;
- there is no breakpoint in $(\widetilde{f(x)} - 2^{-m_x}, \widetilde{f(x)} + 2^{-m_x})$.

The Table Maker's Dilemma

A breakpoint is a point where the rounding function changes.

Restatement of the question: determine $m_x \in \mathbb{N}$, s.t.

- $f(x) \in (\widetilde{f(x)} - 2^{-m_x}, \widetilde{f(x)} + 2^{-m_x})$;
- there is no breakpoint in $(\widetilde{f(x)} - 2^{-m_x}, \widetilde{f(x)} + 2^{-m_x})$.

Two important additional issues:

- $m_x \in \mathbb{N}$ should be as small as possible (otherwise the cost of evaluation of $\widetilde{f(x)}$ will get prohibitive!);
- well, are we sure that m_x does exist?

The Table Maker's Dilemma

A breakpoint is a point where the rounding function changes.

Restatement of the question: determine $m_x \in \mathbb{N}$, s.t.

- $f(x) \in (\widetilde{f(x)} - 2^{-m_x}, \widetilde{f(x)} + 2^{-m_x})$;
- there is no breakpoint in $(\widetilde{f(x)} - 2^{-m_x}, \widetilde{f(x)} + 2^{-m_x})$.

Two important additional issues:

- $m_x \in \mathbb{N}$ should be as small as possible (otherwise the cost of evaluation of $\widetilde{f(x)}$ will get prohibitive!);
- well, are we sure that m_x does exist? No: if $f(x)$ is a breakpoint!

The Table Maker's Dilemma

A breakpoint is a point where the rounding function changes.

Restatement of the question: determine $m_x \in \mathbb{N}$, s.t.

- $f(x) \in (\widetilde{f(x)} - 2^{-m_x}, \widetilde{f(x)} + 2^{-m_x})$;
- there is no breakpoint in $(\widetilde{f(x)} - 2^{-m_x}, \widetilde{f(x)} + 2^{-m_x})$.

Two important additional issues:

- $m_x \in \mathbb{N}$ should be as small as possible (otherwise the cost of evaluation of $\widetilde{f(x)}$ will get prohibitive!);
- well, are we sure that m_x does exist? No: if $f(x)$ is a breakpoint!

TMD: how to make sure that, for all FP number x,

$$\mathsf{RN}(f(x)) = \mathsf{RN}\left(\widetilde{f(x)}\right)?$$

The Table Maker's Dilemma

A breakpoint is a point where the rounding function changes. In this talk, it is the middle of two consecutive FP numbers.

Two-step challenge:

- Determine the set BP_f of all the FP numbers x such that $f(x)$ is a breakpoint;

The Table Maker's Dilemma

A breakpoint is a point where the rounding function changes. In this talk, it is the middle of two consecutive FP numbers.

Two-step challenge:

- Determine the set BP_f of all the FP numbers x such that $f(x)$ is a breakpoint;
- Find $m \in \mathbb{N}$, as small as possible, such that for all FP number $y \notin BP_f$, if we use an internal precision of m bits to evaluate the $f(y)$'s, then we will always get $\mathrm{RN}(f(y))$.

The Table Maker's Dilemma: an Example

Consider the function 2^x and the binary64/double precision FP number (base $2, n = 53$)

$$x = \frac{8520761231538509}{2^{62}}$$

We have

$$\begin{aligned} 2^{53+x} &= 9018742077413030.9999999999999999998805 \cdots (\text{decimal}) \\ &= 1 \cdots 0.\underbrace{1 \cdots\cdots\cdots\cdots 1}_{60 consecutive 1's}0 \cdots (\text{binary}). \end{aligned}$$

The Table Maker's Dilemma: an Example

Consider the function 2^x and the binary64/double precision FP number (base $2, n = 53$)

$$x = \frac{8520761231538509}{2^{62}}$$

We have

$$\begin{aligned} 2^{53+x} &= 9018742077413030.999999999999999998805\cdots(\text{decimal}) \\ &= 1\cdots0.\underbrace{1\cdots\cdots\cdots\cdots1}_{60 consecutive 1's}0\cdots(\text{binary}). \end{aligned}$$

Hardest-to-round (HR) case for function 2^x and double precision FP numbers. The value of m is 113 here.

The Table Maker's Dilemma: an Example

Consider the function 2^x and the binary64/double precision FP number (base $2, n = 53$)

$$x = \frac{8520761231538509}{2^{62}}$$

We have

$$\begin{aligned} 2^{53+x} &= 9018742077413030.9999999999999999998805\cdots(\text{decimal}) \\ &= 1\cdots0.\underbrace{1\cdots\cdots\cdots\cdots1}_{60 consecutive 1's}0\cdots(\text{binary}). \end{aligned}$$

Hardest-to-round (HR) case for function 2^x and double precision FP numbers. The value of m is 113 here.

Function f: $\sqrt[n]{}$, $\sin$, $\cos$, $\arcsin$, $\arccos$, $\tan$, $\arctan$, $\exp$, $\log$, $\sinh$, $\cosh$...

The Table Maker's Dilemma

A breakpoint is a point where the rounding function changes. In this talk, it is the middle of two consecutive FP numbers.

Two-step challenge:

- Determine the set BP_f of all the FP numbers x such that $f(x)$ is a breakpoint;

The Table Maker's Dilemma

A breakpoint is a point where the rounding function changes. In this talk, it is the middle of two consecutive FP numbers.

Two-step challenge:

- Determine the set BP_f of all the FP numbers x such that $f(x)$ is a breakpoint;
- Find $m \in \mathbb{N}$, as small as possible, such that for all FP number $y \notin BP_f$, if we use an internal precision of m bits to evaluate the $f(y)$'s, then we will always get $\mathsf{RN}(f(y))$.

Finding m beyond which there is no problem ?

- Function f: $\sin$, $\cos$, $\arcsin$, $\arccos$, $\tan$, $\arctan$, $\exp$, $\log$, $\sinh$, $\cosh$.

Finding m beyond which there is no problem ?

- Function f: $\sin$, $\cos$, $\arcsin$, $\arccos$, $\tan$, $\arctan$, $\exp$, $\log$, $\sinh$, $\cosh$.
- Lindemann's theorem ($z \neq 0$ algebraic $\Rightarrow e^z$ transcendental) $\rightarrow$ except for straightforward cases (e^0, $\ln(1)$, $\sin(0)$, $\ldots$), if x is a FP number, there exists an m, say m_x, s.t. rounding the m_x-bit approximation $\Leftrightarrow$ rounding $f(x)$;

Finding m beyond which there is no problem ?

- Function f: $\sin$, $\cos$, $\arcsin$, $\arccos$, $\tan$, $\arctan$, $\exp$, $\log$, $\sinh$, $\cosh$.
- Lindemann's theorem ($z \neq 0$ algebraic $\Rightarrow e^z$ transcendental) $\rightarrow$ except for straightforward cases (e^0, $\ln(1)$, $\sin(0)$, $\ldots$), if x is a FP number, there exists an m, say m_x, s.t. rounding the m_x-bit approximation $\Leftrightarrow$ rounding $f(x)$;
- finite number of FP numbers $\rightarrow \exists m_{\mathsf{max}} = \max_x(m_x)$ s.t. $\forall x$, rounding the m_{max}-bit approximation to $f(x)$ is equivalent to rounding $f(x)$;

Finding m beyond which there is no problem ?

- Function f: $\sin$, $\cos$, $\arcsin$, $\arccos$, $\tan$, $\arctan$, $\exp$, $\log$, $\sinh$, $\cosh$.
- Lindemann's theorem ($z \neq 0$ algebraic $\Rightarrow e^z$ transcendental) $\rightarrow$ except for straightforward cases (e^0, $\ln(1)$, $\sin(0)$, $\ldots$), if x is a FP number, there exists an m, say m_x, s.t. rounding the m_x-bit approximation $\Leftrightarrow$ rounding $f(x)$;
- finite number of FP numbers $\rightarrow \exists m_{\text{max}} = \max_x(m_x)$ s.t. $\forall x$, rounding the m_{max}-bit approximation to $f(x)$ is equivalent to rounding $f(x)$;
- Questions: how large can m be? How to determine it?

Some insight (Warning: Hand-waving!)...

- the infinitely precise significand y of $f(x)$ has the form:

$$y = y_0.y_1y_2\cdots y_{n-1}\underbrace{01111111\cdots 11}_{k \text{ bits}}xxxxx\cdots$$

or — with $k \geqslant 1$.

$$y = y_0.y_1y_2\cdots y_{n-1}\overbrace{10000000\cdots 00}^{k \text{ bits}}xxxxx\cdots$$

- Assuming that after the k^{th} position the "1" and "0" are equally likely, the "probability" of having $k \geqslant k_0$ is 2^{1-k_0};
- if we consider N input FP numbers, around $N \times 2^{1-k_0}$ values for which $k \geqslant k_0$;

$\rightarrow$ no longer happens as soon as k_0 is significantly larger than $\log_2(N)$ (for one given value of the exponent, as soon as $k_0 \gg n$).

$\rightarrow$ roughly,

$$\text{"}m_{max} = \log_2(N) + n + o(1)\text{"}.$$

Hence, if $N = 2^{n-1}$, then m_{max} should be slightly larger than $2n$.

The TMD is a Diophantine Approximation Problem

We want to determine $m \in \mathbb{N}$ such that, for all FP number x,

- either $f(x)$ is a breakpoint (in this talk, the middle of two consecutive FP numbers);
- or there is no breakpoint in $(\widetilde{f(x)} - 2^{-m}, \widetilde{f(x)} + 2^{-m})$.

Second constraint restated: assume, w.l.o.g, that x and $f(x) \in [1, 2]$, we want to ensure that, for all $2^{n-1} \leqslant j \leqslant 2^n - 1$, we have $\left|\widetilde{f(x)} - \frac{2j+1}{2^n}\right| \geqslant 2^{-m}$.

The TMD is a Diophantine Approximation Problem

We want to determine $m \in \mathbb{N}$ such that, for all FP number x,

- either $f(x)$ is a breakpoint (in this talk, the middle of two consecutive FP numbers);
- or there is no breakpoint in $(\widetilde{f(x)} - 2^{-m}, \widetilde{f(x)} + 2^{-m})$.

Second constraint restated: assume, w.l.o.g, that x and $f(x) \in [1, 2]$, we want to ensure that, for all $2^{n-1} \leqslant j \leqslant 2^n - 1$, we have $\left|\widetilde{f(x)} - \frac{2j+1}{2^n}\right| \geqslant 2^{-m}$.

Assume that $\left|\widetilde{f(x)} - f(x)\right| < 2^{-m}$. Then if $\left|f(x) - \frac{2j+1}{2^n}\right| \geqslant 2^{-m+1}$, we get

$$\left|\widetilde{f(x)} - \frac{2j+1}{2^n}\right| \geqslant \left|f(x) - \frac{2j+1}{2^n}\right| - \left|\widetilde{f(x)} - f(x)\right| \geqslant 2^{-m}.$$

What can Diophantine Approximation do for us?

Theorem (Dirichlet)

For all $x \in \mathbb{R}$, the inequality $\left|x - \frac{p}{q}\right| < \frac{1}{q^2}$ has an infinite number of solutions.

What about the inequality $\left|x - \frac{p}{q}\right| < \frac{1}{q^\mu}$, with $\mu > 2$?

What can Diophantine Approximation do for us?

Theorem (Khinchine)

Let $\varphi : (0, +\infty) \to (0, +\infty)$*, continuous, s.t.* $x \mapsto x\varphi(x)$ *is non-increasing. Then*

1. $\left|x - \frac{p}{q}\right| < \frac{\varphi(q)}{q}$ *has, for almost all* $x \in \mathbb{R}$*, a finite number of solutions if for some* $c > 0$, $\int_c^{+\infty} \varphi(x)\mathrm{d}x < +\infty$.
2. $\left|x - \frac{p}{q}\right| < \frac{\varphi(q)}{q}$ *has, for almost all* $x \in \mathbb{R}$*, an infinite number of solutions if for some* $c > 0$, $\int_c^{+\infty} \varphi(x)\mathrm{d}x$ *diverges.*

Consider $\varphi(q) = 1/q, \varphi(q) = 1/q^{1+\varepsilon}, \varphi(q) = 1/(q(\log q)^{\varepsilon})$, for $\varepsilon > 0$.

What can Diophantine Approximation do for us?

Corollary

For almost all $x \in \mathbb{R}$, for all $\varepsilon > 0$,

$$\left| x - \frac{p}{q} \right| < \frac{1}{q^{2+\varepsilon}} \text{ has a finite number of solutions.}$$

Almost all or all? Well, for instance, the numbers $x_a = \sum_{n \in \mathbb{N}} a^{-n!}$ with $a \in \mathbb{N}, a \geq 2$ satisfy:

for all $n \in \mathbb{N}$, the inequality $\left| x - \frac{p}{q} \right| < \frac{1}{q^n}$ has an infinite number of solutions.

What can Diophantine Approximation do for us? Case of the Algebraic Numbers

Theorem (Roth, 1955)

If $x \in \mathbb{R}$, x algebraic, then,

$$\forall \varepsilon > 0, \quad \exists\, C(x,\varepsilon) \in \mathbb{R}^+ :$$
$$\forall (p,q) \in \mathbb{Z} \times \mathbb{N}^*, \quad \left| x - \frac{p}{q} \right| \geq \frac{C(x,\varepsilon)}{q^{2+\varepsilon}}\,.$$

Important remark: in our setting, q is a power of 2.

What can Diophantine Approximation do for us? Case of the S-integers

Theorem (Ridout, 1957)

If $x \in \mathbb{R}$, x *algebraic, then,*

$$\forall\, \varepsilon > 0, \quad \exists\, C(x,\varepsilon) \in \mathbb{R}^+ :$$
$$\forall (p,n) \in \mathbb{Z} \times \mathbb{N}^*, \quad \left| x - \frac{p}{2^n} \right| \geq \frac{C(x,\varepsilon)}{2^{(1+\varepsilon)n}}.$$

Big problem: the constant $C(x,\varepsilon)$ can't be effectively computed...

What can Diophantine Approximation do for us? Case of the Algebraic Numbers

Theorem (Liouville, 1844 and 1851)

Let x is algebraic of degree d, and P its minimal polynomial over $\mathbb{Z}$, then,

$$\forall (p,q) \in \mathbb{Z} \times \mathbb{N}^*, \quad \left| x - \frac{p}{q} \right| \geq \frac{C(x)}{q^d},$$

with $C(x) = \dfrac{1}{\max\limits_{x-\frac{1}{2} \leqslant t \leqslant x+\frac{1}{2}} |P'(t)|}$.

What can Diophantine Approximation do for us? The $\exp$ function

From a study by Nesterenko and Waldschmidt (1995), we have, in the case of the double-precision/binary64 format (α and α' are binary64FP formats):

$$\left|e^{\alpha'} - \alpha\right| \geq 2^{-7290678}.$$

Computing with precision 7290678 is feasible, but far too expensive in practice.

Moreover, the hand-waving argument suggests that precision 120 should be enough for double-precision/binary64 arguments.

What can Diophantine Approximation do for us? The $\exp$ function

From a study by Nesterenko and Waldschmidt (1995), we have, in the case of the double-precision/binary64 format (α and α' are binary64FP formats):

$$\left|e^{\alpha'} - \alpha\right| \geq 2^{-7290678}.$$

Computing with precision 7290678 is feasible, but far too expensive in practice.

Moreover, the hand-waving argument suggests that precision 120 should be enough for double-precision/binary64 arguments.

We need an algorithmic approach!

The Algorithmic Approach

Assume, w.l.o.g., that f maps $[1,2]$ to $[1,2]$. Remember that we want to find $m \in \mathbb{N}$, as small as possible, s.t. for all $2^{n-1} \leqslant j \leqslant 2^n - 1$, we have

$$\left| f(x) - \frac{2j+1}{2^n} \right| \geqslant 2^{-m+1}.$$

The idea is to replace f with a polynomial $P \in \mathbb{Q}[x]$. Here again, if, for $y \in [1,2]$, if $|f(y) - P(y)| \leqslant 2^{-m+1}$ then

$$\left| P(x) - \frac{2j+1}{2^n} \right| \geqslant 2^{-m+1} \text{ will imply } \left| f(x) - \frac{2j+1}{2^n} \right| \geqslant 2^{-m+1}.$$

The Algorithmic Approach

Idea: replacing the function f with $P \in \mathbb{Q}[x]$, but there are issues:

- we have to work on subintervals I of $[1, 2]$ over which the approximation $|f(y) - P(y)| \leqslant 2^{-m+1}$ will be satisfied;
- the size of I is related to the degree of P;
- the degree of P has an impact on the solving of the inequalities

$$\left| P(x) - \frac{2j+1}{2^n} \right| \geqslant 2^{-m+1}.$$

The Algorithmic Approach

- If the approximation is of degree 1: algorithm by V. Lefèvre (actually, this kind of stuff was addressed before by Ostrowski, Cassels and other people);

The Algorithmic Approach

- If the approximation is of degree 1: algorithm by V. Lefèvre (actually, this kind of stuff was addressed before by Ostrowski, Cassels and other people);
- If the approximation is of degree $\geqslant 2$: an algorithm by Stehlé, Zimmermann and Lefèvre, relying on Coppersmith' method.

The Algorithmic Approach

- If the approximation is of degree 1: algorithm by V. Lefèvre (actually, this kind of stuff was addressed before by Ostrowski, Cassels and other people);

Degree 1 approximation: Cassels-Lefèvre-Ostrowski-... algorithm

The problem (more or less) boils down to, given α and $\beta \in \mathbb{R}$, minimizing the distance of $\alpha\mathbb{Z} + \beta$ to $\mathbb{Z}$.

More precisely, given α and $\beta \in \mathbb{R}$, $N \in \mathbb{N}$, we want to prove that

$$|q\alpha + \beta - p| \geqslant 2^{-m+1} \text{ for all } (p, q) \in \mathbb{Z} \times \mathbb{N} \text{ s.t. } q < N.$$

M. Gouicem did a really nice bibliographic search regarding this problem. Thanks also to Guillaume Hanrot and Thomas Caissard.

Ostrowski's numeration system

Let $\alpha \in (0,1)$ be an irrational number. Let $\alpha = [0; a_1, a_2, \cdots, a_n, \cdots]$ be its continued fraction expansion with convergents $p_n/q_n = [0; a_1, a_2, \cdots, a_n]$.

Proposition

Every integer N can be expanded uniquely in the form $N = \sum_{k=1}^{m} b_k q_{k-1}$, where $0 \leqslant b_1 \leqslant a_1 - 1, 0 \leqslant b_k \leqslant a_k$ for $k \geqslant 2, b_k = 0$ if $b_{k+1} = a_{k+1}$.

Ostrowski's numeration system

Real numbers are expanded according to the basis given by the sequence $(\theta_n)_{n\geqslant 0}$, where $\theta_n = q_n\alpha - p_n$.

Proposition (Lesca)

Every real number $-\alpha \leqslant \beta < 1-\alpha$ can be expanded uniquely in the form $\beta = \sum_{k=1}^{+\infty} c_k\theta_{k-1}$, where $0 \leqslant c_1 \leqslant a_1 - 1, 0 \leqslant c_k \leqslant a_k$ for $k \geqslant 2, c_k = 0$ if $c_{k+1} = a_{k+1}, c_k = a_k$ for infinitely many even and odd integers.

A modification of Euclid's algorithm computes this expansion.

Ostrowski's numeration system

Ostrowski's numeration system is used to approximate β modulo 1 by numbers of the form $N\alpha$, with $N \in N$.

Indeed, the sequence of integers $N_n = \sum_{k=1}^{n} c_k q_{k-1}$ provides good approximations of $\beta = \sum_{k=1}^{+\infty} c_k \theta_{k-1}$, since

$$N_n \alpha = \sum_{k=1}^{n} c_k q_{k-1} \alpha \equiv \sum_{k=1}^{n} c_k \underbrace{(q_{k-1}\alpha - p_{k-1})}_{\theta_{k-1}} \mod 1.$$

The Algorithmic Approach

- If the approximation is of degree 1: algorithm by V. Lefèvre (actually, this kind of stuff was addressed before by Ostrowski, Cassels and other people);
- If the approximation is of degree $\geqslant 2$: an algorithm by Stehlé, Zimmermann and Lefèvre, relying on a bivariate extension of Coppersmith' method.

The Algorithmic Approach

- If the approximation is of degree $\geqslant 2$: an algorithm by Stehlé, Zimmermann and Lefèvre, relying on a bivariate extension of Coppersmith' method.

The Algorithmic Approach

- If the approximation is of degree $\geqslant 2$: an algorithm by Stehlé, Zimmermann and Lefèvre, relying on a bivariate extension of Coppersmith' method.

It is slightly simpler (for me!) to present an earlier work by Boneh and Durfee, which was the first to use this bivariate extension.

Small roots modulo N

Works by Vallée, Girault, Toffin.

Works by Coppersmith and Howgrave-Graham.

Let $P \in \mathbb{Z}[X]$ monic, $N \in \mathbb{N}$, we search for the "small" $x_0 \in \mathbb{Z}$ s.t. $P(x_0) = 0 \bmod N$.

This method, based on LLL, works (and it's fast) as $|x_0| < N^{1/\deg P}$.
Consequences:

- Factorization of numbers of the form $p^r q$: Work by Boneh, Durfee and Howgrave-Graham.
- Attack of RSA with a small decryption exponent: Boneh and Durfee.

Cryptanalysis of RSA with a small decryption exponent

Reminder: Let p and q be prime, let $N = pq$.
One chooses e prime s.t. $\varphi(N) = (p-1)(q-1)$ and d s.t. $ed = 1 \bmod \varphi(N)$.
Encryption: $x = m^e \bmod N$.
Decryption: $x^d \bmod N = m \bmod N$.
N and e are public, d, p, q secret.

Modular exponentiation is expensive $\Rightarrow$ let's use a small d!

Boneh-Durfee : d should not be too small.

Conjecture

We can easily recover d if $d < N^{0.5}$.

Cryptanalysis of RSA with a small decryption exponent

Wiener (1990) : true if $d < N^{0.25}$ (continued fractions).

Boneh and Durfee (1999) : true if $d < N^{0.292}$.

Let $s = -(p+q)/2$ and $A = (N+1)/2$, there exists k s.t.

$$k(A+s) = 1 \bmod e :$$

A and e are known, k and s are unknown.

Assumption: e has the same order of magnitude as N.

We set $f(x,y) = x(A+y) - 1$. We search for (x_0, y_0) s.t. $f(x_0, y_0) = 0 \bmod e$ with $|x_0| \leqslant e^{\delta}$, $|y_0| < e^{0.5}$.

Cryptanalysis of RSA with a small decryption exponent

We set $f(x, y) = x(A + y) - 1$. We search for (x_0, y_0) s.t. $f(x_0, y_0) = 0 \bmod e$ with $|x_0| \leqslant e^{\delta}$, $|y_0| < e^{0.5}$.

Bivariate version of Coppersmith' method.

Let $h(x, y) = \sum_{i,j} a_{i,j} x^i y^j$, we put $||h(x, y)||^2 = \sum_{i,j} |a_{i,j}|^2$.

Lemma

Let $h(x, y) \in \mathbb{Z}[x, y]$, sum of at most w monomials, s.t.

1. $h(x_0, y_0) = 0 \bmod e^m$ *for an integer* m *with* $|x_0| < X$, $|y_0| < Y$;
2. $||h(xX, yY)|| < e^m/\sqrt{w}$;

then $h(x_0, y_0) = 0$.

Cryptanalysis of RSA with a small decryption exponent

We set $f(x, y) = x(A + y) - 1$. We search for (x_0, y_0) s.t. $f(x_0, y_0) = 0 \bmod e$ with $|x_0| \leqslant e^{\delta}$, $|y_0| < e^{0.5}$.

LLL yields P_1 et $P_2 \in \mathbb{Z}[x, y]$ s. t. $P_1(x_0, y_0) = 0$ et $P_2(x_0, y_0) = 0$ in $\mathbb{Z}$.

Soit $R(y) = \text{Rés}_x(P_1, P_2)$, **heuristic** : $R \neq 0$.

$y_0 \in \mathbb{Z}$ root de R : easy to compute. As $y_0 = \frac{p+q}{2}$, we obtain p and q.

Remark: we could (should) also use Hensel lifting to determine the roots.

Cryptanalysis of RSA with a small decryption exponent

We set $f(x, y) = x(A + y) - 1$. We search for (x_0, y_0) s.t. $f(x_0, y_0) = 0 \bmod e$ avec $|x_0| \leqslant e^{\delta}$, $|y_0| < e^{0.5}$.

Idea: construct, from f, a lattice L "generated" by the polynomials. To each polynomial corresponds one vector and the coordinates of the vector are the coefficients of the polynomials.

LLL yields two "short" vectors P_1 and P_2 of this lattice: the lemma will ensure that $P_1(x_0, y_0) = 0$ and $P_2(x_0, y_0) = 0$ in $\mathbb{Z}$.

Reminder: Let $(b_1, \ldots, b_d)$ be an LLL-reduced basis of L then

$$||b_1|| \leqslant 2^{d/2}(\det L)^{1/d} \quad \text{et} \quad ||b_2|| \leqslant 2^{d/2}(\det L)^{\frac{1}{d-1}}.$$

Cryptanalysis of RSA with a small decryption exponent

Let $g_{i,k} = x^i f^k(x,y) e^{m-k}$ et $h_{j,k} = y^j f^k(x,y) e^{m-k}$ with $m \in \mathbb{N}$.

(x_0, y_0) is a root of the $g_{i,k}$ and $h_{j,k}$ modulo e^m for $k = 0, \ldots, m$.

We consider the lattice generated by the polynomials $g_{i,k}(xX, yY)$ et $h_{j,k}(xX, yY)$.

Cryptanalysis of RSA with a small decryption exponent

$f(x,y) = x(A+y) - 1$.
Example: lattice generated by $g_{i,k} = x^i f^k(x,y) e^{m-k}$ and $h_{j,k} = y^j f^k(x,y) e^{m-k}$ for $m = 3$, $k = 0, \ldots, 3$, $i = 0, \ldots, 3-k$ and $j = 0, 1$.

$$
\begin{array}{c|c|cc|ccc|cccc|cccc|}
 & e^3 & xe^3 & fe^2 & x^2e^3 & xfe^2 & f^2e & x^3e^3 & x^2fe^2 & xf^2e & f^3 & ye^3 & yfe^2 & yf^2e & yf^3 \\
\hline
1 & e^3 & & - & & & - & & & & - & & & & \\
\hline
x & 0 & e^3X & - & & - & - & & - & - & & & & & \\
xy & \vdots & \ddots & e^2XY & & & - & & & & - & & - & - & - \\
\hline
x^2 & \vdots & & \ddots & e^3X^2 & - & - & & - & - & - & & & & \\
x^2y & \vdots & & & \ddots & e^2X^2Y & - & & & - & - & & & - & \\
x^2y^2 & \vdots & & & & \ddots & eX^2Y^2 & & & & - & & & - & - \\
\hline
x^3 & \vdots & & & & & \ddots & e^3X^3 & - & - & - & & & & \\
x^3y & \vdots & & & & & & \ddots & e^2X^3Y & - & - & & & & - \\
x^3y^2 & \vdots & & & & & & & \ddots & eX^3Y^2 & - & & & & - \\
x^3y^3 & \vdots & & & & & & & & \ddots & X^3Y^3 & & & & - \\
\hline
y & \vdots & & & & & & & & & \ddots & e^3Y & - & - & - \\
xy^2 & \vdots & & & & & & & & & & \ddots & e^2XY^2 & - & - \\
x^2y^3 & \vdots & & & & & & & & & & & \ddots & eX^2Y^3 & - \\
x^3y^4 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & X^3Y^4 \\
\hline
\end{array}
$$

Cryptanalysis of RSA with a small decryption exponent

LLL gives two short vectors P_1 and P_2: we have $P_1(x_0, y_0) = 0$ in $P_2(x_0, y_0) = 0$ in $\mathbb{Z}$ (Lemma).

Reminder: Let $(b_1, \ldots, b_d)$ be an LLL-reduced basis L then

$$||b_1|| \leqslant 2^{d/2}(\det L)^{1/d} \quad \text{et} \quad ||b_2|| \leqslant 2^{d/2}(\det L)^{\frac{1}{d-1}}.$$

Triangular matrix: RSA broken as soon as $d \leqslant N^{7/6-\sqrt{7}/3}$ with $7/6 - \sqrt{7}/3 = 0.2847...$

A refinement of the choice of the family of polynomials yields: RSA is broken as soon as $d \leqslant N^{1-1/\sqrt{2}}$ with $1 - 1/\sqrt{2} = 0.29289....$

Complexity of the two algorithms

It is related to the size of the interval on which the approximation $|f(x) - \widetilde{f(x)}|$ is valid!

Case of degree 1 approximation (Cassels-Lefèvre-Ostrowski-... algorithm): on each interval, the complexity is negligible but to address the whole interval $[1, 2]$, you need to subdivide it into $2^{2n/3}$ intervals...

Complexity of SLZ

The complexity of the algorithm, for solving $|f(x) - \frac{2j+1}{2^n}| < 1/M$ is essentially

$$\mathrm{poly}(n + \log(M)) \cdot 2^{\frac{n^2}{n+\log(M)}},$$

- "usual version" of the problem: $M \approx 2^n$ (comes from the "probabilistic" reasoning given above). The complexity is essentially a polynomial function times $2^{n/2}$. Too costly to be useful for big values (e.g., $n = 113$);
- "degraded" version: use the fact that if $M \approx 2^{n^2}$ the complexity boils down to polynomial $\Rightarrow$ information of the form *"if the approximation to f is with error $< \varepsilon$ (with ε smaller than necessary yet reasonable for efficient implementation) then rounding the approximation $\Leftrightarrow$ rounding f"*

Results

Table: *Worst cases for exponentials of double precision FP numbers.*

Interval	worst case (binary)
$[-\infty, -2^{-30}]$	$\exp(-1.1110110100110001100011101111101101100010011111101010 \times 2^{-27})$ $= 1.111111111111111111111111111100 \cdots 0111000100 \quad 1 \quad 1^{59}0001\ldots \times 2^{-1}$
$[-2^{-30}, 0)$	$\exp(-1.0000000000000000000000000000000000000000000000000001 \times 2^{-51})$ $= 1.111111111111111 \cdots 11111111111111100 \quad 0 \quad 0^{100}1010\ldots \times 2^{-1}$
$(0, +2^{-30}]$	$\exp(1.1111111111111111111111111111111111111111111111111111 \times 2^{-53})$ $= 1.0000000000000000000000000000000000000000000000000000 \quad 1 \quad 1^{104}0101\ldots$
$[2^{-30}, +\infty]$	$\exp(1.0111111111111111001111111111111111011100000000000100100 \times 2^{-32})$ $= 1.0000000000000000000000000000000101111111111111101000 \quad 0 \quad 0^{57}1101\ldots$
	$\exp(1.1000000000000001011111111111111101101111111111111011100 \times 2^{-32})$ $= 1.00000000000000000000000000000001100000000000000010111 \quad 1 \quad 1^{57}0010\ldots$
	$\exp(1.1001111010011100101110111111101011000001000000001011 \times 2^{-31})$ $= 1.0000000000000000000000000000001100111101001110010111 \quad 1 \quad 0^{57}1010\ldots$
	$\exp(110.0000111101010010111100110111101011101100111111010 0)$ $= 110101100.01010000101101000000100111001000101011101110 \quad 0 \quad 0^{57}1000\ldots$

Results

Table: *Worst cases for logarithms of double precision FP numbers.*

Interval	**worst case (binary)**
$[2^{-1074}, 1)$	$\log(1.1110101001110001110110000101110011101110000000100000 \times 2^{-509})$ $= -101100000.0010100101101010011001101011010000101111111111 \quad 1 \quad 1^{60}0000\ldots$
	$\log(1.1001010001110110111000110000010011001101011111000111 \times 2^{-384})$ $= -100001001.10110110000011001010111101000111101100110101 \quad 1 \quad 0^{60}1010\ldots$
	$\log(1.0010011011101001110001001101001100100111100101100000 \times 2^{-232})$ $= -10100000.101010110010110000100101111001101000010000100 \quad 0 \quad 0^{60}1001\ldots$
	$\log(1.0110000100111001010101011101110010000000001011111000 \times 2^{-35})$ $= -10111.111100000010111110011011101011110110000000110101 \quad 0 \quad 1^{60}0011\ldots$
$(1, 2^{1024}]$	$\log(1.0110001010101000100001100001001101100010100110110110 \times 2^{678})$ $= 111010110.0100011110011110101110100111110010010111 0001 \quad 0 \quad 0^{64}1110\ldots$

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f .
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Evaluation of Elementary Functions

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

Goal: evaluation of φ to a given accuracy η.

- Step 0. Computation of hardest-to-round cases: V. Lefèvre and J.-M. Muller.
- Step 1. Argument reduction (Payne & Hanek, Ng, Daumas *et al*): evaluation of a function φ over $\mathbb{R}$ or a subset of $\mathbb{R}$ is reduced to the evaluation of a function f over $[a, b]$.
- Step 2. Computation of $p^\star$, a "machine-efficient" polynomial approximation of f .
- Step 3. Computation of a rigorous approximation error $||f - p^\star||$.
- Step 4. Computation of a certified evalutation error of $p^\star$: GAPPA (G. Melquiond).

Some references

- Approximation Theory
  - E. W. Cheney, *Introduction to Approximation Theory, 2nd edition*, AMS Chelsa Publishing. Providence, Rhode Island, 1982.
  - M. J. D. Powell, *Approximation theory and methods*, Cambridge University Press, Cambridge, 1981.
- Computer Algebra
  - J. Gerhard and J. von zur Gathen, *Modern Computer Algebra*, Cambridge University Press.

Some references

- Computer Arithmetic (Correct Rounding, Evaluation of Functions)
  - V. Lefèvre, *Moyens arithmétiques pour un calcul fiable*, PhD thesis, École Normale Supérieure de Lyon, January 2000.
  - J.-M. Muller, *Elementary Functions, Algorithms and Implementation*, Birkhaüser.
  - J.-M. Muller *et al.*, *Handbook of Floating-Point Arithmetic*, Birkhaüser.
  - D. Stehlé, *On the randomness of bits generated by sufficiently smooth functions*. In F. Hess, S. Pauli, and M. E. Pohst, editors, Proc. of the 7th Algorithmic Number Theory Symposium, ANTS VII, vol. 4078 of LNCS, pp 257–274. Springer-Verlag, Berlin, 2006.
- Continued Fractions, Diophantine Approximation
  - J.W.S. Cassels, *An Introduction to Diophantine Approximation*, Cambridge University Press.
  - R. Descombes, *Éléments de théorie des nombres*, PUF.
  - A.Y. Khinchin, *Continued Fractions*, Chicago University Press.
  - O. Perron, *Die Lehre von den Kettenbrüchen*, B.G. Teubner.

Some references

- Ostrowski's numeration system
  - V. Berthé, *Autour du système de numération d'Ostrowski*, Bull. Belgian Math. Soc. 8 (2001), 209–239.
  - V. Berthé, *Numeration and discrete dynamical systems*, Computing, 94 (2012) 369–387.
- Polynomial Approximation, Evaluation of Functions
  - S. Chevillard, *Évaluation efficace de fonctions numériques - Outils et exemples*, PhD thesis, École Normale Supérieure de Lyon, July 2009.
  - M. Joldeș, *Rigorous Polynomial Approximations and Applications*, PhD thesis, École Normale Supérieure de Lyon, September 2011.
  - C. Lauter, *Arrondi correct de fonctions mathématiques - Fonctions univariées et bivariées, certification et automatisation*, PhD thesis, École Normale Supérieure de Lyon, October 2008.