Exact real arithmetic

Keith Briggs

Keith.Briggs@bt.com

http://more.btexact.com/people/briggsk2/xr-kent-talk-pp.pdf

University of Kent Computing Laboratory 2004 Feb 10 1400

typeset 2004 February 9 12:18 in pdfLaTeX on a linux system

Contents

- exp
- *Computable bounds*
- *My python implementation*
- *Lambdas*
- *Comparison*
- *Python implementation in action*
- *C++ implementation in action*
- *Features of my implementations*
- *Computational geometry application*
- *Simultaneous Diophantine approximation application*
- *Limitations*
- *Maths* $\Longleftrightarrow$ *software*
- *Conclusion*
- *Bibliography*

Outline

- what's the problem?
- representation and processing of reals
- my implementation of the Boehm scheme
- applications
- notation
  - *integers:* $\mathbb{Z} = \{ \ldots, -2, -1, 0, 1, 2, \ldots \}$
  - *rationals:* $\mathbb{Q} = \{ p/q;\ p, q \in \mathbb{Z} \}$
  - $\mathbb{R}\backslash\mathbb{Q} = \{$ *algebraics* $\cup$ *transcendentals* $\}$
  - $\lfloor x \rceil$ *is the nearest integer to* x

IEEE floating point

☞ finite set of rationals and approximate operations:

$$\begin{aligned} \mathbb{F} &= \{\, m\,2^e : |m| < 2^{53}, |e| < 1024,\ \text{NaN},\ \text{Inf}\,\} \\ \mathbb{O} &= \{\, +, -, *, /\,\}, \text{round to even or nearest} \end{aligned}$$

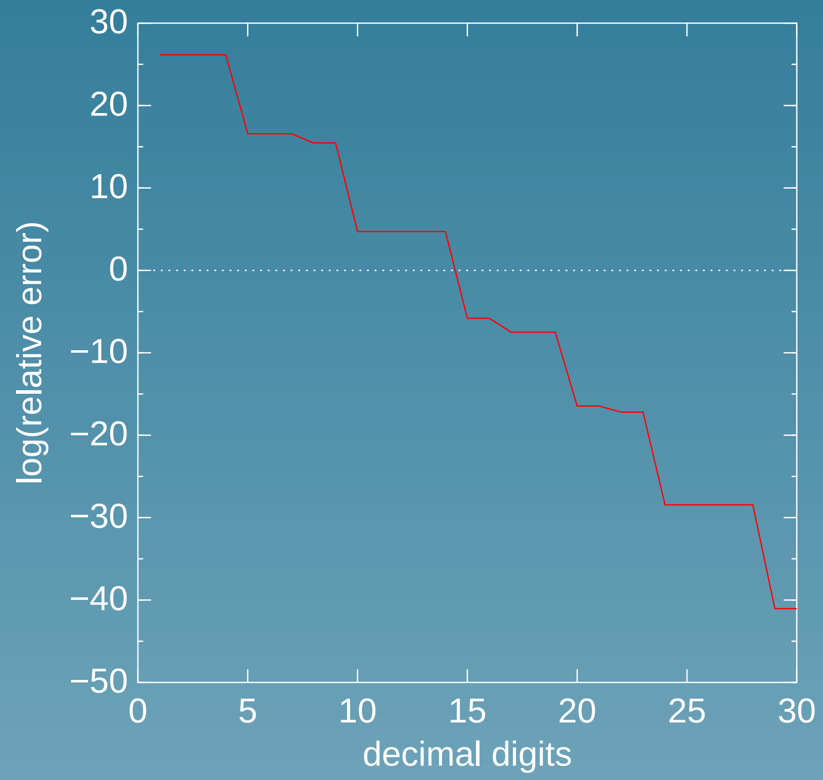
☞ problems!

$$\begin{aligned} x_0 &= \mathbb{F}(0.9) \\ x_{k+1} &= \mathbb{F}(3.999) * x_k * (1 - x_k) \qquad k = 0, 1, 2, \ldots \end{aligned}$$

☞ in $\mathbb{F}$, $x_{53} > 0.5$, but the exact result is
$x_{53} = 0.13023587481177373303964373008057 0\ldots$

Another floating point disaster

$$
\begin{aligned}
u_0 &= e - 1 \\
u_k &= k\, u_{k-1} - 1 \qquad k = 1, \ldots, 25
\end{aligned}
$$



Irrationals

☞ construction conventionally involves a *limit* of a sequence of rationals

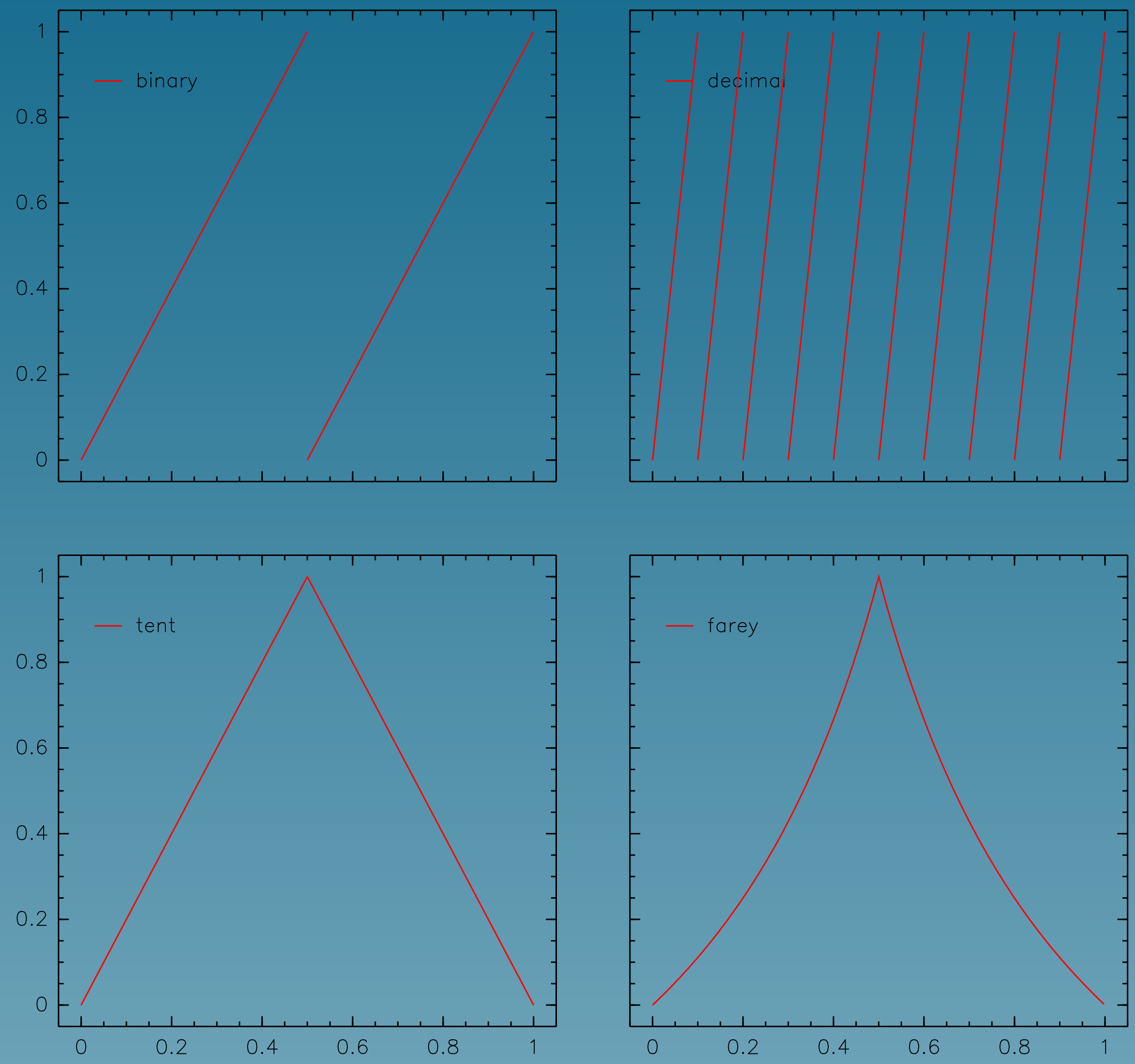
- ▷ *rate of convergence is not specified*
- ▷ *traditional digit expansions converge from below*
- ▷ *what about negative digits?*

☞ other possibilities:

- ▷ *symbolic dynamics of an expanding map*
- ▷ *continued fractions, Möbius maps, LFTs*
- ▷ *nested intervals with rational endpoints*
- ▷ *non-integer base - e.g. golden ratio*
- ▷ *does something even better exist?*

We want a representation that allows the computation of more significant digits first

Digit expansions

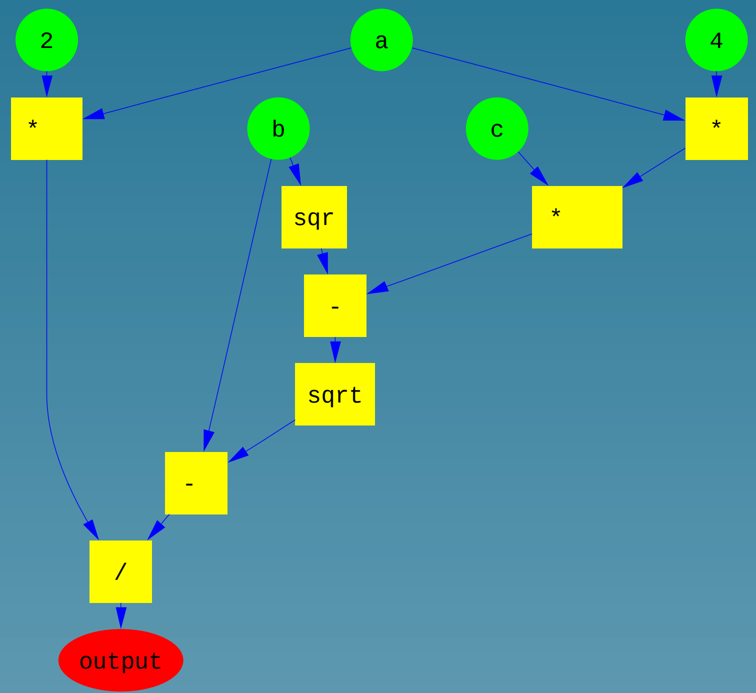


$x \mapsto f(x)$

digit=branch number

What's the problem?

☞ example: find sign of one root $x = \frac{-b+\sqrt{b^2-4ac}}{2a}$ of quadratic $ax^2+bx+c=0$



▷ *input nodes don't know how much precision to send*

▷ *all input nodes send data, even if it eventually may not be needed*

▷ *to recalculate requires the whole tree to be re-evaluated*

Continued fraction arithmetic

$$x = [x_0; x_1, x_2, x_3, \ldots] = x_0 + \cfrac{1}{x_1 + \cfrac{1}{x_2 + \cdots}}$$

- ☞ studied by Gosper, Vuillemin, Liardet, Stambul, Ménissier-Morain, Potts
- ☞ only easy operation: $\frac{ax+b}{cx+d}$
- ☞ for $+, *$ etc., inefficient
- ☞ for $\exp, \sin$ etc., *very* inefficient!
- ☞ problem:

> How do we convert inputs to cf form?

Scaled-integer representation (Boehm)

☞ for $\widehat{x} \in \mathbb{R}$, consider a function $x : \mathbb{Z}^+ \to \mathbb{Z}$ such that

$$\boxed{\ |\ \boldsymbol{B^n\ \widehat{x} - x(n)}\ |\ < \mathbf{1}\ }$$

for fixed $B = 2, 3, 4, \ldots$

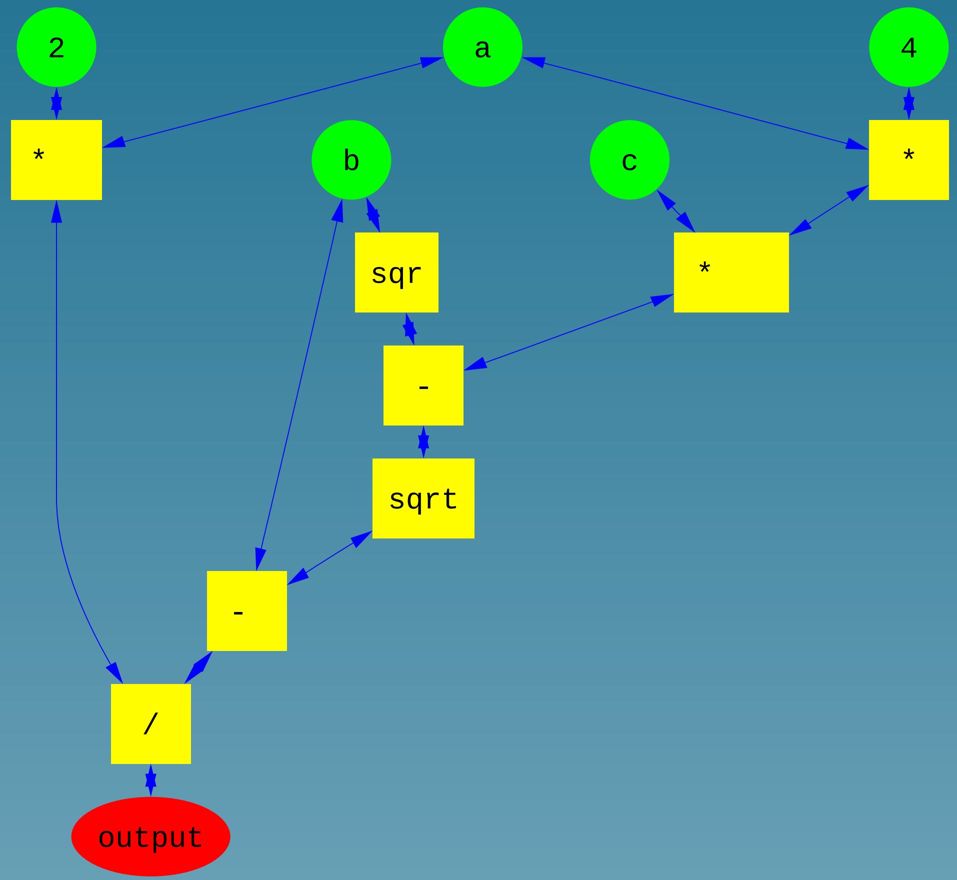
☞ $x : \mathbb{Z}^+ \to \mathbb{Z} \Leftrightarrow x \in \mathbb{XR}$

☞ in other words: $$\frac{x(n)-1}{B^n} < \widehat{x} < \frac{x(n)+1}{B^n}$$

☞ example: $\widehat{x} \in \mathbb{Q},\ \ x(n) \equiv \lfloor B^n\ \widehat{x} \rfloor$

> All computation with $\mathbb{XR}$s is reduced to large integer operations

Data flow in $\mathbb{XR}$ system



Some easy operations

- $|x|(n) \equiv |x(n)|$
- $[-x](n) \equiv -x(n)$
- $[\mathsf{sqrt}(x)](n) \equiv \sqrt{x(2n)}$
- for $m \in \mathbb{Z}$, $[m\,x](n) \equiv \lfloor\, m\,x(n+p)/2^p \,\rceil$, $p = 1+\log_2|m|$
- for $m \in \mathbb{Z}^*$, $[x/m](n) \equiv \lfloor\, x(n)/m \,\rceil$
- comparison: if for some n, $x(n)$ and $y(n)$ differ by more than 1, they are unequal
- caching: $x(n)$ can be evaluated as $\lfloor\, x(m)/2^{m-n} \,\rceil$ if $x(m)$ for $m > n$ is available

Addition and multiplication (for $B=2$)

$$\boxed{[x{+}y](n) \equiv \lfloor\ (x(n{+}2){+}y(n{+}2){+}2)\ /4\ \rfloor}$$

$$\begin{aligned}
& |\ [x{+}y](n) - 2^n(\widehat{x}{+}\widehat{y})\ | \\
= \quad & |\ \lfloor\ (x(n{+}2){+}y(n{+}2){+}2\ )\ /4 \rfloor - 2^n(\widehat{x}{+}\widehat{y})\ | \\
\leqslant \quad & 1/2 + |\ (x(n{+}2){+}y(n{+}2))/4 - 2^n(\widehat{x}{+}\widehat{y})\ | \\
= \quad & 1/2 + |\ x(n{+}2){+}y(n{+}2) - 2^{n+2}(\widehat{x}{+}\widehat{y})\ |\ /\ 4 \\
\leqslant \quad & 1/2 + |\ x(n{+}2) - 2^{n+2}\,\widehat{x}\ |\ /\ 4 + |\ y(n{+}2) - 2^{n+2}\,\widehat{y}\ |\ /\ 4 \\
< \quad & 1
\end{aligned}$$

$$\boxed{[x{*}y](n) \equiv \lfloor\ \left(x(q{+}r{+}1)y(p{+}s{+}1){+}2^{p+q+3}\right)\ /2^{p+q+4}\ \rfloor}$$

where

$$r = \lfloor\ (n{+}2)/2\ \rfloor,\ \ s = n{+}2{-}r,\ \ p = \lfloor\ \log_2|x(r)|\ \rfloor,\ \ q = \lfloor\ \log_2|x(s)|\ \rfloor$$

Algebraic number construction

- given a polynomial p with integer coefficients, and integers $a, k > 0$, we can compute the sign of p at a/B^k with only integer operations

```
def signat(p,a,k): # return the sign of polynomial p at a/B^k
 n,w=len(p),p[0]
 for j in 1..n:
  w*=a
  w+=p[j]*B^(k*j)
 return w>0
```

- thus, given a bracketed root:

$$\operatorname{sign} p(a/B^k) < 0, \quad \operatorname{sign} p(b/B^k) > 0$$

we may refine it by bisection to any desired accuracy

Transcendental functions

☞ $f = \exp, \arctan, \sin$ etc. can be computed if we can implement an approximating function $\tilde{f} : \mathbb{Q} \times \mathbb{Z}^+ \to \mathbb{Z}$ such that

$$\left| B^n f(q) - \tilde{f}(q, n) \right| < 1$$

☞ we need approximating functions which give us *a priori* error bounds

☞ or, at least, error bounds must be computable in rational arithmetic

☞ for π, use

$$\pi = \sum_{n=0}^{\infty} \left[\frac{4}{8n+1} - \frac{2}{8n+4} - \frac{1}{8n+5} - \frac{1}{8n+6} \right] \frac{1}{16^n}$$

exp

☞ easy to get $\exp(x)$ with bounded absolute error if we know $|x|<1$: use Padé approximants:

$$\max_{|x|<1} |\, e^x - R_{n/n}(x) \,| \leqslant \frac{8(n!)^2}{(2n)!(2n+1)!}$$

- *we can't test* $|x|<1$*!. . .*
- *but we can safely find* k *such that* $|\widehat{x}/2^k|<1$

The big software challenge

Produce efficient software which hides the bottom-up data flow and can be used by a programmer as if it were conventional top-down code

- ☞ needed features:
  - ▷ *λ anonymous function constructor*
  - ▷ *operator overloading*
- ☞ not easily possible in Fortran, C, java, . . .
- ☞ easy in python, Haskell, clean, ML, ocaml
- ☞ possible in C++! (with tricks)

My python implementation

- www.python.org
- ‘perl done right’
- much cleaner syntax than java
- portable (Linux, Unix, Mac, . . . , Windows)
- functional features - lambda, map, filter, reduce
- operator overloading
- free ☺

Lambdas

```
def +(x,y):
 # compute the sum of x and y for B=2
 return lambda n: (x(n+2)+y(n+2)+2)/4

def sqrt(x):
 # compute the square root of x for B=2
 return lambda n: sqrt(x(2*n))
```

☞ possibility of distributed computation: a lambda may be executed remotely

Comparison

This is a system for proving inequalities

. . . but, equality is *undecidable*

```
def <(x,y):
 # return true if x<y, false if x>y, else don't return
 n=1
 while 1:
  if x(n)<y(n)-1: return true
  if x(n)>y(n)+1: return false
  n+=1

def max(x,y):
 # compute the maximum of x and y without comparison
 return (x+y+abs(x-y))/2
```

Python implementation in action

```
> import XR
> e=exp(XR(1))
> print contfrac(e)
[2, 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1, 10, 1, 1, 12, 1, 1, 14]
> print cos(2*pi()/7).dec(50)
0.62348980185873353052500488400423981063227473
> print polyroot([8,4,-4,-1],0,1,0).dec(50)
0.62348980185873353052500488400423981063227473
> x=XR(Q(1,3))
> for i in range(10):
  print 2*x>1
  x=4*x*(1-x)
0 1 0 1 0 1 1 0 0 1
```

C++ implementation in action

```
#include "XR.h"

int main() {
 XR e=exp(XR(1));
 cout<<contfrac(e);
 cout<<cos(2*pi()/7).dec(50);
 ZZ coeffs[]={8,4,-4,-1};
 cout<<polyroot(coeffs,0,1,0).dec(50);
 x=XR(Q(1,3));
 for (int i=0; i<10; i++) {
  cout<<2*x>1;
  x=4*x*(1-x);
 }
}
```

My C implementation

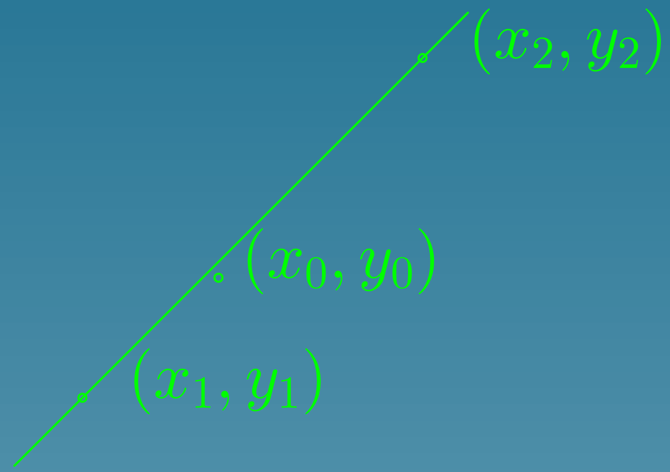
- `more.btexact.com/people/briggsk2/xrc.html`
- portable ISO C
- Uses pointers to construct DAG representing compositions of lambdas
- Function call ust be written for every arithmetic operation
- C++ wrapper

Features of my implementations

- python, C++, C
- all integer arithmetic, using `gmp`
- automatic caching of all intermediate results
- easily integrated with existing code
- C, C++ versions are fully compiled
- possible application areas:
  - *computational number theory*
  - *computational geometry*
  - *computer-assisted proofs in analysis*
  - *computer algebra*

Computational geometry application

- ☞ Simple planar example: line through $(x_1, y_1), (x_2, y_2)$; is the point (x_0, y_0) to the left or right of the line?



- ☞ This is determined by $\operatorname{sign}\left([(y_1 - y_0)(x_2 - x_1) - (x_1 - x_0)(y_2 - y_1)]\right)$

> With $\mathbb{XR}$ arithmetic, the sign is always determined correctly and with the minimal necessary computation

Simultaneous Diophantine approximation application

- typical subproblem: given
  - $x_1, x_2 \in \mathbb{R}$
  - $p_1, p_2, q \in \mathbb{Z}$
  - $e_1 = |qx_1 - p_1|,\ e_2 = |qx_2 - p_2|$
- determine whether $e_1 > e_2$
  - *very efficiently solved in* $\mathbb{XR}$ *with trivial coding by user*
- is $\exp(\pi\sqrt{163})$ an integer?

```
x=exp(pi*sqrt(XR(163)))
print x>floor(x) # prints 'True'
```

Limitations

☞ memory demands

☞ non-incrementality

☞ the floor function does not terminate on integer inputs

☞ transcendental functions still incomplete

☞ no verified decimal output

Mathematics $\overset{?}{\Longleftrightarrow}$ software

☞ mathematics $\Longrightarrow$ mathematical software

- *Numerical analysis*
- *Statistics*
- *Computer algebra*
- *Computational number theory*
- *Combinatorics and graph theory*
- *Finite groups*
- *Theorem proving*
- ...

☞ what about the other direction?

Conclusion

- ☞ it's worth rethinking how we represent numbers and do arithmetic
- ☞ lots of future work possible:
  - ▷ *granularity - automatic choice for* B*?*
  - ▷ *optimize caching strategy*
  - ▷ *log, sin, cos etc.*
  - ▷ *non-functional - graph method*
  - ▷ *parallel and multi-threaded techniques*
  - ▷ *characterize the set* $\mathbb{XR}$
  - ▷ *complexity analysis (not assuming each* $*$, $+$ *etc. has the same cost)*

Bibliography 1

- ☞ H-J Boehm, R Cartwright, M Riggle, and M J O'Donnell: *Exact real arithmetic: A case study in higher order programming* dev.acm.org/pubs/citations/proceedings/lfp/319838/p162-boehm/
- ☞ V Ménissier-Morain: *Arithmétique exacte, conception, algorithmique et performances d'une implémentation informatique en précision arbitraire* Paris thesis 1994 calfor.lip6.fr/∼vmm/
- ☞ J R Harrison: *Theorem proving with the real numbers* Cambridge thesis 1996 www.ftp.cl.cam.ac.uk/ftp/papers/reports/
- ☞ P Potts: *Exact Real Arithmetic using Möbius Transformations* www.purplefinder.com/∼potts/

Bibliography 2

- ☞ Computability and Complexity in Analysis Network `www.informatik.fernuni-hagen.de/import/cca/`
- ☞ K Weihrauch: *Computable Analysis* `www.informatik.fernuni-hagen.de/import/thi1/klaus.weihrauch/book.html`
- ☞ P Liardet and P Stambul: *Algebraic Computations with Continued Fractions* `www.idealibrary.com/links/doi/10.1006/jnth.1998.2274`
- ☞ K M Briggs: *XR homepage* `more.btexact.com/people/briggsk2/XR.html`; This talk: `more.btexact.com/people/briggsk2/xr-kent-talk-pp.pdf`