MPFI: a library for arbitrary precision interval arithmetic

Nathalie Revol  Arénaire (CNRS/ENSL/INRIA), LIP, ENS-Lyon and Lab. ANO, Univ. Lille
Nathalie.Revol@ens-lyon.fr

Fabrice Rouillier  Spaces, CNRS, LIP6, Univ. Paris 6, LORIA-INRIA
Fabrice.Rouillier@loria.fr

SCAN 2002, Paris, France, 24-27 September
Reliable computing: Interval Arithmetic


Numbers are replaced by intervals.
Ex: \( \pi \) replaced by \([3.14159, 3.14160]\)

Computations: can be done with interval operands.

Advantages:

- Every result is guaranteed (including rounding errors).
- Data known up to measurement errors are representable.
- Global information can be reached (“computing in the large”).

Drawbacks: overestimation of the results.
Hansen’s algorithm for global optimization

Hansen 1992

\[ \mathcal{L} = \text{list of boxes to process} := \{ X_0 \} \quad F: \text{function to minimize} \]

while \( \mathcal{L} \neq \emptyset \) loop

suppress \( X \) from \( \mathcal{L} \)

reject \( X \) ?

yes if \( F(X) > \bar{f} \)

yes if Grad\( F(X) \) \( \not= 0 \)

yes if \( HF(X) \) has its diag. non \( > 0 \)

reduce \( X \)

Newton applied with the gradient

solve \( Y \subset X \) such that \( F(Y) \leq \bar{f} \)

bisect \( Y \) into \( Y_1 \) and \( Y_2 \) if \( Y \) is not a result

insert \( Y_1 \) and \( Y_2 \) in \( \mathcal{L} \)
Solving linear systems
preconditioned Gauss-Seidel algorithm

Hansen & Sengupta 1981

Linear system: $Ax = b$ with $A$ and $b$ given.

Problem: compute an enclosure of
Hull ($\Sigma_{\exists\exists}(A, b)) = $ Hull ($\{x : \exists A \in A, \exists b \in b, Ax = b\}$).

Hansen & Sengupta’s algorithm
compute $C$ an approximation of $\text{mid}(A)^{-1}$
apply Gauss-Seidel to $CAx = Cb$ until convergence.

Idea (hope):
$CA$ contains the identity matrix, is diagonally dominant, thus the iteration matrix has a spectral radius $< 1$ and this iteration is a contraction.
Agenda

• Introduction: when is more precision wanted?
  – global optimization (Hansen’s algorithm)
  – linear system solving (Hansen & Sengupta’s algorithm)

• MPFI: a library for arbitrary precision interval arithmetic
  – general view of MPFI
  – MPFI in details
  – other libraries

• Applications
  – robot kinematics
  – isolation of real roots of a polynomial
  – approximation of zeros of a function
MPFI  Multiple Precision Floating-point Interval arithmetic library

general view

- **what:** C library for arbitrary precision interval arithmetic

- **why:** to compute reliably and accurately

- **where:** freely available, including source code and documentation (last release: April 2002)

- **who:** N. Revol and F. Rouillier

- **how:** based on MPFR for arbitrary precision floating-point arithmetic (based on GMP: efficient and portable)
• **Requirements:** directed roundings are required, exact is better. Even for elementary functions! (provided by MPFR)

• **Interval:** connected closed subset of \( IR \)

• **Operation:** \( op(X_1, \ldots, X_n) \supset \{ op(x_1, \ldots, x_n) : x_i \in X_i \} \)

• **Undefined operation:** NaN if \( op(x_1, \ldots, x_n) \) is undefined (no \( a \ priori \) intersection with the domain)

• **Precision:** intervals are represented by their endpoints, each endpoint carries its own precision.
MPFI: functionalities

- arbitrary precision interval data type
- arithmetic operations: $+, -, \times, \div, \sqrt{\cdot}$
- constants: $\pi$, log 2, Euler constant
- elementary functions: exp, log, atan, cosh, asinh...
- IEEE-754 special values: $\infty$, signed zeros, NaN
- conversions to and from integer, double, exact naturals, exact integers, rational, “reals” (MPFR numbers)
- Input/Output (to be improved)
Performances

Development version

Gaussian elimination on a random M-matrix (300 × 300)

<table>
<thead>
<tr>
<th>ARITHM PREC</th>
<th>double 53 bits</th>
<th>fi.lib 53 bits</th>
<th>MPFI 53 bits</th>
<th>MPFI 63 bits</th>
<th>MPFI 127 bits</th>
<th>MPFI 255 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME (s)</td>
<td>0.42</td>
<td>4.39</td>
<td>21.75</td>
<td>22.35</td>
<td>24.11</td>
<td>34.09</td>
</tr>
<tr>
<td>RATIO</td>
<td>0.10</td>
<td>1.00</td>
<td>4.95</td>
<td>5.09</td>
<td>5.49</td>
<td>7.76</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ARITHM PREC</th>
<th>MPFI 511 bits</th>
<th>MPFI 1023 bits</th>
<th>MPFI 2047 bits</th>
<th>MPFI 4095 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIME (s)</td>
<td>51.61</td>
<td>112.30</td>
<td>285.57</td>
<td>817.05</td>
</tr>
<tr>
<td>RATIO</td>
<td>11.75</td>
<td>25.58</td>
<td>65.05</td>
<td>186.11</td>
</tr>
</tbody>
</table>

(Athlon 1GHz)
Other libraries

Most of the interval arithmetic libraries are based on machine floating-point precision (Profil/BIAS, Intlib, Sun Forte. . . )

Some libraries offer multiple precision:

- **Maple, Mathematica**: floating-point arithmetic and roundings bugged
- **range**: “the interval is only crudely represented” (Aberth)
- **XSC**: limited use (in long accumulators)
- **IntLab**: limited set of operations and functions
Agenda

• Introduction: when is more precision wanted?
  – global optimization (Hansen’s algorithm)
  – linear system solving (Hansen & Sengupta’s algorithm)

• MPFI: a library for arbitrary precision interval arithmetic
  – general view of MPFI
  – MPFI in details
  – other libraries

• Applications
  – robot kinematics
  – isolation of real roots of a polynomial
  – approximation of zeros of a function
Parallel robot kinematics

Formulation: algebraic system depending on 7 to 12 variables.

Goal: computing all the real solutions (up to 40).

Usual numerical computational strategies failed on this example using hardware doubles.

Solution obtained by

- a few exact methods
- MPFI using 128 bits of precision

Remark: some preliminary computations were performed using exact computations (Gröbner bases).
Isolation of polynomial real roots

Rouillier and Zimmermann 2002

Descartes’ rule of sign
Let $P = \sum_{i=0}^{d} p_i x^i$ be a polynomial,
$V(P) =$ number of sign changes in $(p_0, \ldots, p_d)$
and $\text{pos}(P) =$ number of $> 0$ real roots of $P$, counted with mult.
Then $\text{pos}(P) \leq V(P)$ and $V(P) - \text{pos}(P)$ is even.
Isolation of polynomial real roots algorithm

“Uspensky’s” algo.
Assumption: \( P \) is a square-free polynomial with roots in \([0, 1]\).
Transform \( P \) into \( Q = \sum_i q_i x^i \) such that \( V(Q) = 0 \) or 1
i.e. \( P \) has 0 or 1 root on a given interval:
a root of \( P \) is isolated.

Modified algorithm using MPFI
replace exact computations in \( \text{Transform } P \text{ into } Q \)
by MPFI computations:
\( q_i \) is now an interval.
If \( q_i \ni 0 \) then stop interval computation, increase MPFI precision or go
on with exact computation.
Approximation of zeros: interval Newton algorithm
principle of one iteration


The result will be a list of intervals.
Interval Newton algorithm

Input: $F, F', X_0$  // $X_0$ initial search interval
Initialization: $\mathcal{L} = \{X_0\}$,
$\alpha = 0.75$  // any value in $]0.5, 1[$ is suitable

Loop: while $\mathcal{L} \neq \emptyset$

Suppress $(X, \mathcal{L})$

Increase the working precision if needed

$x := \text{mid}(X)$

$(X_1, X_2) := (x - \frac{F(\{x\})}{F'(X)}) \cap X$  // $X_1$ and $X_2$ can be empty

if $w(X_1) > \alpha w(X)$ or $w(X_2) > \alpha w(X)$ then $(X_1, X_2) := \text{bisect}(X)$
if $X_1 \neq \emptyset$ and $F(X_1) \ni 0$ then

if $w(X_1)/|\text{mid}(X_1)| \leq \varepsilon_X$ and $w(F(X_1)) \leq \varepsilon_Y$ then Insert $X_1$ in $\text{Res}$
else Insert $X_1$ in $\mathcal{L}$

same handling of $X_2$

Output: $\text{Res}$, a list of intervals that may contain the roots.
Interval Newton algorithm: experiments

Chebychev polynomials: $C_n(\cos(\theta)) = \cos(n\theta)$.

Results: very precise roots for degrees up to 40, with proof of existence and uniqueness.

Wilkinson polynomial: $\prod_{i=1}^{20} (X - i)$.

With enough precision to be able to store exactly the coefficients: roots found with a precision $5.10^{-2}$ and a proof of existence (but not uniqueness). A lot of intervals are not eliminated: $[0.96, 1.02]$ et $[1.62, 20.984]$.

With enough precision and a perturbation $\pm 2^{-19}$ on the coefficient of $X^{19}$: roots (with proof of existence but not uniqueness): $1 \pm 4.10^{-2}$, $2 \pm 5.10^{-2}$, $3 \pm 4.10^{-2}$, $4 \pm 4.10^{-2}$, $5 \pm 4.10^{-2}$, $6 \pm 5.10^{-2}$, $7 \pm 6.10^{-2}$ and $[7.91, 22.11]$. A lot of intervals are not eliminated: $[0.96, 22.64]$. 
Conclusion

MPFI:
• library for arbitrary precision interval computation
• at a reasonable cost.

Applications
• guaranteed results with arbitrary accuracy
• algorithms: increase precision when needed and continue

Coming soon
• MPFI++
• linear algebra
• more algorithms.