THE MPFR LIBRARY (WWW.MPFR.ORG)

PAUL ZIMMERMANN
LORIA/INRIA LORRAINE
NANCY, FRANCE

1. The library

What: a C library for multiple-precision floating-point arithmetic with exact rounding [3].
Why: compute wrt the IEEE754 philosophy in multiple-precision and with mathematical functions.
Where: freely available at `www.mpfr.org` under the GNU Library General Public License.
Who: the Spaces project (`www-spaces.lip6.fr`) and a few external developers (any help still welcome!).
How: based on GNU MP [4] for portability and efficiency.

Differences with IEEE754
- much much slower
- any variable has its own precision (from 1 to any number of bits)
- no gradual underflow (yet?)
- all functions are exactly rounded (including $\exp, \log, \ldots$)

Features of the current version (mpfr-2001)
- input/output (any base up to 36)
- conversions (integer, double)
- basic operations $(+, -, \times, /, \sqrt{\ })$
- comparison
- $\exp, \log, \sin, \cos, \mathrm{agm}$
- constants π and $\log 2$

2. A sample MPFR program

```
#include "gmp.h"
#include "mpfr.h"

main() { mpfr_t x, y;
   mpfr_init2 (x, 13);
   mpfr_set_d (x, 3.1416, GMP_RNDN);
   mpfr_init2 (y, 105);
   mpfr_exp (y, x, GMP_RNDZ);
   mpfr_out_str (stdout, 10, 15, y, GMP_RNDD);
   printf ("\n");
}
```

```
% gcc sample.c -lmpfr -lgmp
% ./a.out
2.31408987920503e1
```

The program on the left first initializes a MPFR variable x with a mantissa of 13 bits, then sets its value to the double-precision number 3.1416 rounded to nearest, which gives 3217/1024. Then a second MPFR variable y is created with a mantissa of 105 bits, and y is set to the exponential of x, rounded towards zero. Then the value of y is printed in base 10, with 15 digits and rounded towards $-\infty$. The program compilation and output is shown below left: exactly 15 digits of the variable y are printed. The rounding controls the input from a double-precision number (`mpfr_set_d`), the exponential computation (`mpfr_exp`), and the output (`mpfr_out_str`). Please note that different variables may have different precisions (here 13 and 105).

3. MPFR inside

$M(n)$ denotes the multiplication time of two n-bit integers (naïve, Karatsuba's, or Schönhage's algorithm):

- $+$ and $-$ hand-written: $O(n)$
- $\times$ based on `mpn_mul`: $M(n)$
- / based on `mpn_divrem`: $O(M(n))$
- $\sqrt{\ }$ based on `mpn_sqrtrem`: $O(M(n))$ [11]
- agm and log in $O(M(n) \log n)$
- exp has complexity $O(M(n) \log^2 n)$ [5]
- $\sin, \cos$ in $O(M(n) \log^2 n)$ [5]

Some timings (1000 bits, PII-366)

`x*y`	took 0.022583 ms
`x/y`	took 0.033876 ms
`sqrt(x)`	took 0.036012 ms
`exp(x)`	took 0.781632 ms
`log(x)`	took 1.358749 ms
`sin_cos(x)`	took 9.763780 ms

4. Connected work

Computer algebra systems (Maple, Mathematica, Macsyma, Axiom, Derive, MuPAD, Axiom) can do arbitrary precision floating-point computations, but with unclear semantics (in particular, no rounding mode) or incorrect implementations.

Brent designed in 1978 a Fortran package called MP [1], very similar to the floating-point class (MPF) of GNU MP. The main difference with MPFR is that MP provides only one rounding mode (R^*-rounding with four guard digits), whereas MPFR provides the four IEEE754 rounding modes (to nearest, towards zero, towards plus and minus infinity). MP did provide elementary functions (log, exp, sin, tan, arcsin, arctan, sinh, cosh, tanh), special functions (Bernoulli numbers, Bessel functions of the first kind, error and complementary error functions, exponential and logarithmic integrals, Dawson's integral, gamma function, Riemann zeta function), and constants like π and γ.

GNU MP [4] provides a class MPF of multiple-precision floating-point numbers. Based on GNU MP too, this package is as portable and efficient as MPFR. However it implements only the basic operations, and no elementary nor special function.

Arithmos [2] performs multiple-precision floating-point computations with correct rounding, in binary or decimal (MPFR computes only in binary), but does not provide (yet) mathematical functions.

The iRRAM software from Norbert Müller, also based on GNU MP, is an implementation of the RealRAM concept [7]. This is a higher-level package, which can provide any number of correct digits of any floating-point value, possibly by increasing the internal working precision. iRRAM works in fixed-point arithmetic, which avoids the problem of determining the sign of an expression which is in fact zero.

5. Companion packages

Nathalie Revol (Arénaire project, INRIA Rhône-Alpes) has written a C++ interface for MPFR [8], which – among other nice features — enables one to write `a = b + c` instead of `mpfr_add(a, b, c, rnd)`.

Nathalie Revol and Fabrice Rouillier have written on top of MPFR a multiple-precision interval arithmetic package called MPFI [9]. Two applications have already been built on top of MPFI: a program to isolate the real roots of a polynomial using Descartes' rule of signs [10], and a similar program using Newton's method together with interval arithmetic [6].

6. Future plans

The next version of MPFR should include all the mathematical functions available in the C9X standard. This includes the hyperbolic and inverse hyperbolic functions, the error function, the gamma function, the Bessel functions. It is also planned to include the Riemann zeta function.

Underflows and overflows will be properly detected, and it will be possible to specify the exponent range $[E_{\min}, E_{\max}]$ (bounds are 32-bit signed integers).

References

[1] Brent, R. P. A Fortran multiple-precision arithmetic package. *ACM Trans. Math. Softw. 4*, 1 (1978), 57–70.

[2] Cuyt, A., Kuterna, P., Verdonk, B., and Vervloet, J. Arithmos: a reliable integrated computational environment. `http://win-www.uia.ac.be/u/cant/arithmos/index.html`, 2001.

[3] Daney, D., Hanrot, G., Lefèvre, V., Rouillier, F., and Zimmermann, P. The MPFR library. `http://www.mpfr.org/`.

[4] Granlund, T. *GNU MP: The GNU Multiple Precision Arithmetic Library*, 3.1 ed., Aug. 2000. Available from `http://www.swox.se/gmp/`.

[5] Jeandel, E. Évaluation rapide de fonctions hypergéométriques. Rapport technique 242, Institut National de Recherche en Informatique et en Automatique, 2000. 17 pages.

[6] Langlois, P., and Revol, N. Validating polynomial numerical computations with complementary automatic methods. Research Report 2001-18, École Normale Supérieure de Lyon, May 2001. 12 pages.

[7] Müller, N. iRRAM - exact arithmetic in C++. http://www.informatik.uni-trier.de/iRRAM/, 2000.

[8] Revol, N. MPFR++, a C++ interface to MPFR. http://www.ens-lyon.fr/~nrevol/nr_software.html, 2001.

[9] Revol, N., and Rouillier, F. MPFI, a multiple precision interval arithmetic library based on MPFR. http://www.ens-lyon.fr/~nrevol/nr_software.html, 2001.

[10] Rouillier, F., and Zimmermann, P. Efficient isolation of a polynomial real roots. Rapport de recherche 4113, Institut National de Recherche en Informatique et en Automatique, Feb. 2001. 16 pages.

[11] Zimmermann, P. Karatsuba square root. Rapport de recherche 3805, Institut National de Recherche en Informatique et en Automatique, 1999.