# Comparison of Interval C/C++ Libraries in Global Optimization

Rafał Dąbrowski* and Bartłomiej Jacek Kubica*[1]

* Warsaw University of Technology, Institute of Control and Computation Engineering, Warsaw, Poland,
e-mail: R.J.Dabrowski@stud.elka.pw.edu.pl

**Abstract.** This article contains results of examination and comparison of some most popular interval libraries. Comparative analysis was based on investigating the efficiency of computations of simple functions and efficiency of the *Interval-Branch-And-Bound* global optimization method.

## 1 Introduction

Interval methods (see e. g. [2], [3], [6]) are a robust tool of global optimization. Thanks to the use of outward rounding when computing interval enclosures on the objective, the interval branch-and-bound method is guaranteed to find all global minima (and distinguish them from local ones unless they are epsilon-optimal). This power comes at a price – interval methods are in general very computationally intensive, which results in long computation times and high memory demands.

Because of that the efficiency is crucial for successful use of interval algorithms. And efficiency is affected not only by the implementation of the branch-and-bound method, but also low-level underlying procedures. As there are several publicly available interval libraries nowadays (e. g. [4], [5], [7], [8], [9]), it is useful to investigate and compare their efficiency to ba able to choose a proper one for a specific problem.

The paper is organized as follows. First we describe four publicly available interval libraries – PROFIL/BIAS, FILIB++, C-XSC and MPFI presenting their basic features. Then we compare the results of a branch-and-bound algorithm implemented in all of them for a few typical nonconvex optimization problems. Obviously, examined functions have many variables and contain in some cases many local minimizers.

## 2 Interval libraries

Interval libraries provide basic interval operations and – optionally – some more advanced procedures. In this paper four of such libraries are considered:

1. PROFIL/BIAS
2. FILIB++

[1] The research has been supported by the Polish Ministry of Science and Higher Education under grant N N514 416934.

3. C-XSC
4. MPFI

### 2.1 PROFIL/BIAS

PROFIL (Programmer's Runtime Optimized Fast Interval Library ) is a C++ library based on BIAS (Basic Interval Arithmetic Subroutines). The author is Olaf Knüppel from Technische Universität in Hamburg. BIAS provides here an interface for interval vector and matrix operations. Profil is a very fast library and still under development. The author put emphasis on very efficient use of the underlying hardware, portability and independency of a specific interval representation[1]. PROFIL/BIAS is a very comfortable in use library. It includes subroutines for local and global optimization and automatic differentiation. We can find here a package with a set of test matrices. Besides main data types as intervals, interval vectors and interval matrices it contains such data structure as linear singly linked list with arbitrary data types.

### 2.2 FILIB++

FILIB++ (Fast Interval Library) is an extension of the interval library FILIB. It is written in C++ and is developing at Wuppertal University in Germany. The authors are M. Lerch, G. Tischler, J. Wolf von Gudenberg, W. Hofschuster and W. Krämer. Implementation of this library is based on templates. Executed computations may be exception-free thanks using special mode which operates also on values representing infinities and NaNs [2]. FILIB++ makes it possible to compute fast a comprehensive set of elementary interval functions with guaranteed bounds. It benefits from implemented *Table Lookup* algorithms. Unfortunately, as for now there are no implementations of own data types like interval vectors or interval matrices. Only basic subroutines have been implemented in this library.

### 2.3 MPFI

MPFI (Multiple Precision Floating-Point Interval Library) is a library created by Nathalie Revol and Fabrice Rouillier at the University of Lyon in France. It is based on MPFR and GMP. This library is designed for computation where multiple precision interval arithmetic is needed. The main aim is to obtain guaranteed and accurate results. MPFI is now a C library and because of that we have to use appropriate functions instead of e.g. overloaded operators. Luckily, in the future it is going to obtain a C++ interface and some new functionalities [3],[4].

### 2.4 C-XSC

C-XSC (A C++ Library for Extended Scientific Computing) has been created at the University of Wuppertal (as FILIB++), but it does not aim efficiency as the main goal. Being less efficient, C-XSC is very developed and contains in particular:

- operations on vectors and matrices,
- operations on complex intervals, their vectors and matrices,
- a few variants of automatic differentiation arithmetic,
- computation of inverse matrices,
- so-called staggered arithmetic (since version 2.2.4),
- ...

Additional packages are also available - for slope arithmetic, linear systems solving, Taylor arithmetic and several other tools.

## 3 The branch-and-bound method

Branch-and-bound is the basic meta-algorithm for solving global optimization problems ([2], [3]). In case of unconstrained optimization it can be expressed by the following pseudocode:

```
IBB (x0, f)
//x0 is the initial box
//f is the interval extension of minimized function
[yl, yu] = f(x0);
L = {(x0, yl)};     // list of boxes to consider
Lsol = {};          // list of solutions
while (L not empty) do
        get from L a box x with minimal yl;
        perform rejection/reduction tests on x;
        if (x was verified to contain a minimum) put (Lsol, x);
        else if (x was verified not to contain a minimum) delete x;
        else if (x is sufficiently small) put (Lsol, x);   // might be a separate list
        else subdivide x and put resulting boxes to L;
end while;
end IBB;
```

The essence of power of interval methods are the rejection/reduction tests.
Main of them are:

- the monotonicity test – compute enclosure of the objective's gradient and check if all components contain zero; if one of them does not, the function is monotonous in the box and cannot contain a minimum – unless lying on the constraints (or bound constraints),
- many variants of the interval Newton operator – a powerful analog of pointwise Newton methods; can verify existence of a solution in an interval and – under specific circumstances – even uniqueness of the solution.

Several textbooks ([2], [3], [7]) describe these tests and their properties. Also other, more elaborate tests can be developed (concavity test, constraint propagation, pruning of boxes, Bauman tents, etc.), but they are not going to be discussed here. Interested reader is referred to the literature.
The implementation considered in our experiments was using only the monotonicity test and the Neton method based on interval Gauss-Seidel iteration ([2], [3], [7]).

## 4 Results

The interval branch-and-bound algorithm was tested on following functions: Booth, Branin, Rastrigin, Rosenbrock, Levy, Camel and Hansen. These functions are often used as benchmarks fo global optimization algorithms. They are multivariate and may contain several or many local

minimizers not only one global minimum on a given area. For each discussed interval library the interval branch-and-bound algorithm was implemented.
Experiments were done on a Linux openSUSE 10.3 system with Intel Core 2 Duo (2GHz) and 3GB RAM. Results are presented in Figure 1.

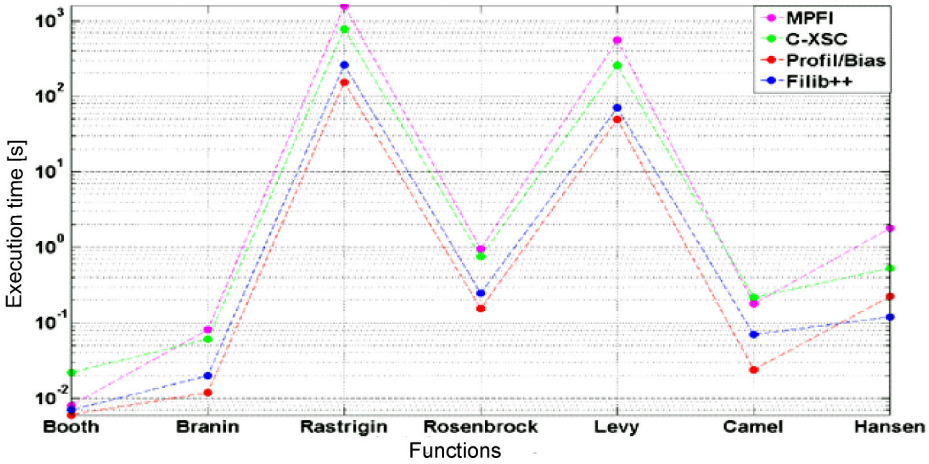


**Figure 1.** IBB algorithm tested on interval libraries.

The experiments show that PROFIL/BIAS was most efficient in the majority of cases. Only for Hansen's function the IBB algorithm needed more time than FILIB++ library. FILIB++ is just behind PROFIL/BIAS in this test. C-XSC and MPFI are the slowest ones. They are not so efficient as PROFIL/BIAS and FILIB++ because they were mainly created for elementary interval arithmetic operations and implementations of basic functions like sine or cosine are inferior with respect to PROFIL/BIAS.
Obtained results can be explained when we compare the efficiency of standard operations and functions implemented in the libraries in Figure 2.

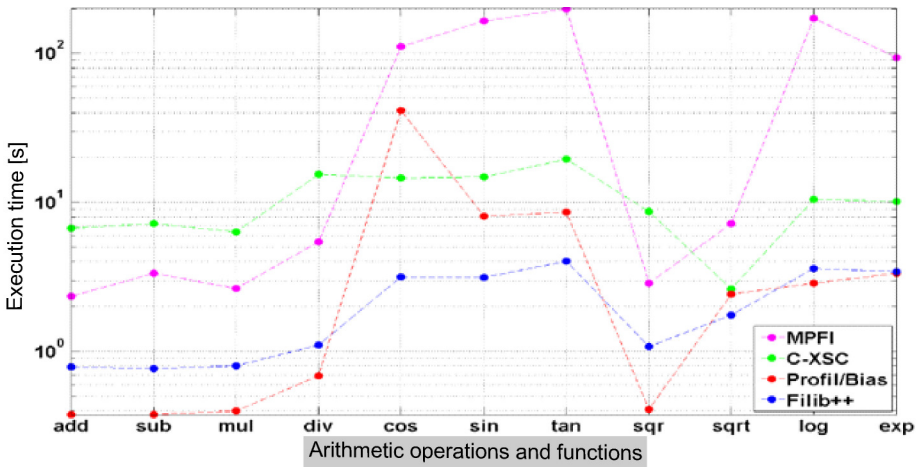


**Figure 2.** Efficiency of standard operations and functions.

Addition, subtraction, multiplication and division are executed the fastest by PROFIL/BIAS. Trigonometric functions are the best in FILIB++ thanks to *Table Lookup* algorithms. As standard operations are used the most frequent, we have the answer why in most cases *Interval-Branch-and-Bound* algorithm was so efficient for PROFIL/BIAS. Only global minimum for Hansen function was found quicker by FILIB++ library. It resulted from frequent use of cosine in this function which formula is presented below (1). Tests show that cosine is 13 times slower in PROFIL/BIAS than FILIB++.

$$f(x)=\sum_{i=1}^{5} i\cdot\cos((i-1)\cdot x_1+i)\cdot\sum_{j=1}^{5} j\cdot\cos((j+1)\cdot x_2+j) \qquad (1)$$

C-XSC and MPFI were the slowest. It should be marked that MPFI has very inefficient trigonometric, logarithm and exponential functions but standard operations are better than in C-XSC.
As far as the accuracy is concerned comparison of searched global minimum with *Interval-Branch-and-Bound* algorithm is not so obvious. Widths of computed intervals depend on the stopping criterion. The moment when global minimum is included in the final interval, which is thin enough to be a searched solution, can be sometimes accidental. However intervals found by PROFIL/BIAS are the widest in most cases.

## 5 Conclusions

Four popular interval libraries have been compared in our experiments. Efficiency of these libraries was compared in a few experiments. And basic features of the packages are presented in the following table:

Table 1. Comparison of basic features of the libraries

| | PROFIL/BIAS | FILIB++ | MPFI | C-XSC |
|---|---|---|---|---|
| Licence | GPL 2 | GPL 2 | LGPL | GPL 2 |
| Arithmetic operations | yes | yes | yes | yes |
| Complex numbers | basic operations | no | no | full |
| Basic functions | yes | yes | yes | yes |
| Vector&matrix operations | yes | no | no | yes |
| Automatic differentiation | yes | no | no | yes |

It occurred that PROFIL/BIAS is most efficient and – when considering the computation time – it should be recommended for use in the majority of cases. Only when computing the cosine is an important part of the algorithm, it would be outperformed by FILIB++. On the other hand, when

concerning the accuracy of computation, PROFIL/BIAS performs worse than other libraries, but the difference is usually very small or almost negligible.
It is also worth noting that C-XSC library – as it is inefficient – contains several useful tools. When we need to make computations on – say – intervals of complex numbers, implementing such operations on PROFIL or FILIB++ would be time consuming; while on C-XSC they are already available. Hence – at least for writing prototypes – C-XSC might be very useful.
Using MPFI on the other hand seems justified only when using its multi-precision features is necessary.
It is worth noting that there are also several other libraries, not considered in the paper – like the implementation of interval arithmetic in the BOOST library or the GAOL package, to name a few.

## Bibliography


[1] R. Dąbrowski, *Analiza porównawcza bibliotek przedziałowych*, BEng thesis under supervision of B. J. Kubica, WUT, 2008.
[2] E. Hansen, G. W. Walster, *Global Optimization Using Interval Analysis, Second Edition: Revised and Expanded*, Marcel Dekker, New York, 2004.
[3] R. B. Kearfott, *Rigorous Global Search: Continuous Problems*, Kluwer Academic Publishers, Dordrecht, 1996.
[4] O. Knüppel, *PROFIL/BIAS V 2.0*, Technische Universität Hamburg-Harburg, Luty 1999.
[5] M. Lerch, G. Tischler, J. Wolf von Gudenberg, *FILIB++ – Interval Library Specification and Reference Manual,* Lehrstuhl für Informatik II, Universität Würzburg, Sierpień 2001.
[6] A. Neumaier, *Interval methods for systems of equations*, Cambridge University Press, Cambridge, 1990.
[7] *MPFI 1.3.4 manual,* manual dołączony do biblioteki MPFI, 2002.
[8] N. Revol, F. Rouillier, *Motivations for an arbitrary precision interval arithmetic and the MPFI library*, Research Report 2002-27, Lyon, Lipiec 2002.
[9] C-XSC library http://www.xsc.de .