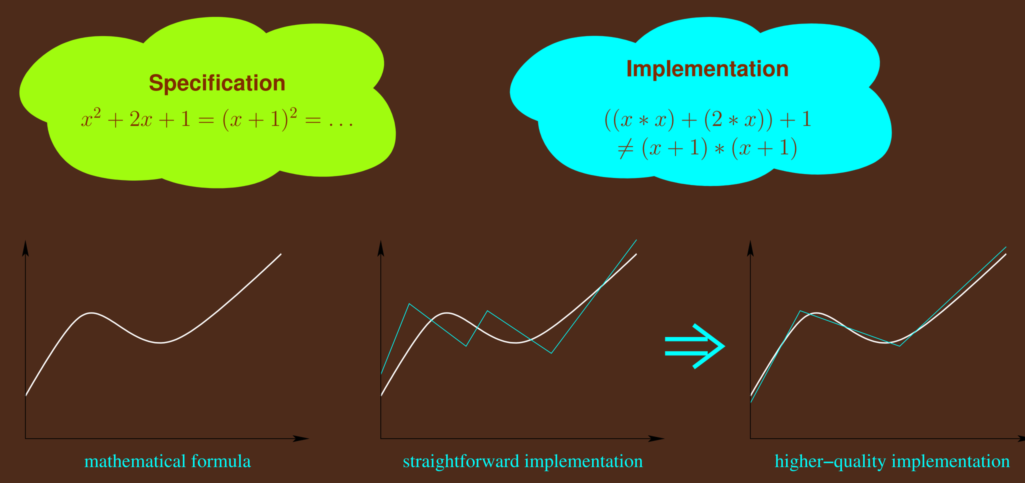
Projet ANR-Blanc STIC :

EVA-Flo : Évaluation et Validation Automatique pour le calcul Flottant
New Automatic Tools for Validated Floating-point Computations



The mathematical model, as illustrated here by the function on the left, can correspond to an implementation (center) that poorly approximates it. The goal is to obtain a better implementation, such as the one on the right, and as automatically as possible.

Demanding quality for numerical computations using floating-point arithmetic

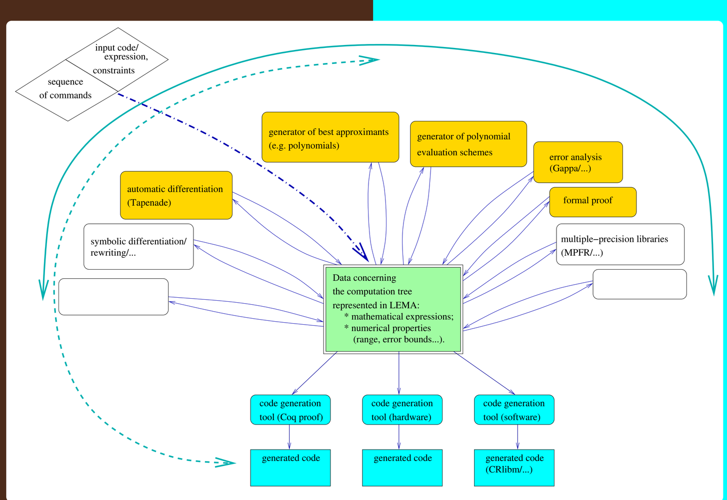
When a mathematical formula is translated into a numerical computation, it is hoped that computed results are close to the corresponding exact values. However, computers usually employ floating-point arithmetic: the representation of numbers has a finite fixed size. Consequently, rounding errors are made. The first goal of the EVA-Flo project is to evaluate numerically a formula in a fast and accurate way. The quality of the result can be specified, such as a relative or absolute error between the exact value and the computed result, or the guarantee that no overflow occurs (numbers too large to be represented and converted into ∞)... The second goal of the EVA-Flo project is that this quality can be quantified (for instance "the relative error is $< 10^{-14}$") and certified. The last goal is that this process of evaluation and validation is automated.

The target mathematical formulas of the EVA-Flo project are expressed using arithmetic or algebraic operations and mathematical functions ($\exp$, $\sinh$, $\arctan$...), and they can contain few conditional branches and loops. Typically, the focus is on small critical portions of large numerical codes.

Floating-point representation

Real numbers are represented on a computer, at the hardware level, as *floating-point numbers*. Such a representation comprises a significand, with a finite and fixed number of digits, multiplied by a power of the computing radix and by a sign (± 1). For instance, in radix 10 and with 3 digits, the number $125\,000$ can be represented as 125×10^3, or 12.5×10^4, or also 1.25×10^5, hence the terminology of *floating point*. The main advantage of this representation is to be able to represent values that have very different orders of magnitude and still to use a very limited amount of memory. For the double precision, the representable numbers vary between 2.25×10^{-308} and 1.798×10^{308}. The main drawback of this representation, as of any fixed-size representation, is that most real numbers, including the results of operations on floating-point numbers, cannot be represented and must be rounded.

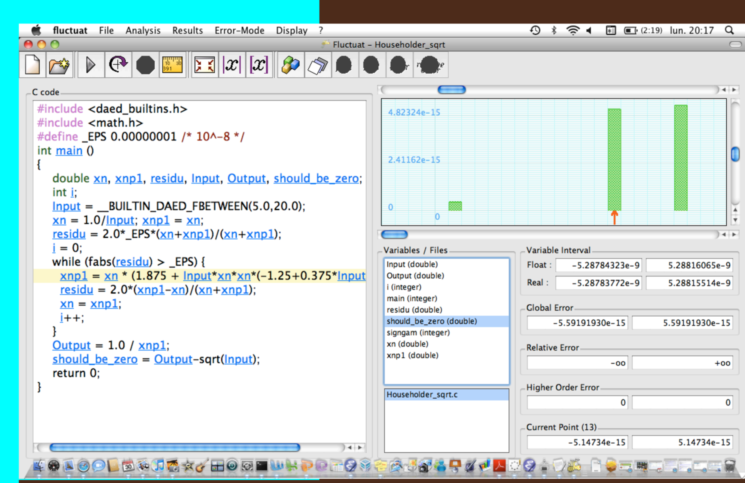
Taming roundoff errors...
as well as other numerical errors made by your computer



LEMA (un Langage pour les Expressions Mathématiques Annotées) is a representation language that encodes not only a program, but also extra knowledge about the link between exact and floating-point values, about properties of some operations... It will serve both as a common representation format between various software tools and as a track of the various steps performed during the automated generation of code (either more accurate, or better suited to a specific architecture such as an embedded processor or a GPU, or ...).

Automate, automate, automate the accumulated expertise

Numerous problems have been handled in a pen and paper manner in the past. The current step consists in automating, at each level, the expertise gained through the handling of these problems. The first level is to specify precisely the desired mathematical result and to determine good approximants (e.g., with a small relative error) that are well-suited to an implementation on a computer. Typically, these approximants are polynomials with floating-point coefficients. The second level is to determine evaluation schemes that are both fast and accurate, using exhaustive search. The architecture of the target processor plays a key role here. Another level is the choice of the technique employed to reach the required accuracy: double-double arithmetic, compensated evaluation schemes.... Both method error and implementation error are then bounded and certified. Usually, the method error is estimated but not bounded, whereas the implementation error is rarely handled. Such proofs on the quality of the computed result use a fine knowledge of the properties of the floating-point arithmetic, they are also based on computations using interval arithmetic and extended-precision arithmetic. Then, a proof is reworked and written so that a proof checker can check it; we use the Coq proof checker. Indeed, a typical proof includes many peculiar cases and is thus error-prone when it is performed by a human. This explains why it is essential to check it automatically.



Fluctuat is a static analyzer, intended to cope with real industrial problems.
Its designers contribute to EVA-Flo through their expertise in automated validation of floating-point codes.

Most significant outcomes

Software production

Reaching full automation becomes closer with the development of software pieces:

- **Sollya:** determination of a good polynomial approximation, including a guaranteed approximation error;
- **Gappa (mainly developed prior to EVA-Flo):** bounds on evaluation errors, that can be checked by the proof checker Coq;
- **CRlibm (mainly developed prior to EVA-Flo):** correctly rounded mathematical functions; this proof of concept library resulted in the recommendation, in the IEEE 754-2008 standard, that elementary functions should be correctly rounded; large parts of its current code are automatically generated by an experimental code generation tool, **metalibm**.
- **FloPoCo:** a VHDL code generator for Floating-Point Cores on FPGAs, with application-specific optimizations for non-standard operators;
- **FLIP:** software emulation of IEEE-754 floating-point arithmetic on some embedded media processors;
- **CGPE:** Code Generation for Polynomial Evaluation, taking into account architectural features;
- **Fluctuat (developed independently and prior to EVA-Flo):** analysis of the numerical quality of scientific codes;
- **Tapenade (developed independently and prior to EVA-Flo):** automatic differentiation of codes;
- **Sardanes project:** analysis and rewriting of mathematical expressions to achieve better accuracy.

Scientific production

Apart from the software developments already mentioned, 6 PhD theses, related to these topics, have been defended. Around 15 articles in scientific journals and 20 presentations at international conferences have been produced. Eventually, the expertise of the group on floating-point arithmetic has given rise to a collective book, *The Handbook of Floating-Point Arithmetic*, published by Birkhäuser in November 2009. Let us also mention the activity in standardization committees: participation to IEEE 754-2008 for floating-point arithmetic, chair of the IEEE 1788 for interval arithmetic.

The EVA-Flo project: *Évaluation et Validation Automatique pour le calcul Flottant - New Automatic Tools for Validated Floating-point Computations* is an ANR Blanc-STIC research project. It is headed by Arénaire (LIP, ENS Lyon). It associates Dali (Eliaus, U. Perpignan), MeASI (LIST, CEA Saclay) and Tropics (INRIA Sophia Antipolis - Méditerranée). The project started in November 2006 and its duration is 48 months. It is subsidized by ANR: 130 k€ and the global cost is 1.5 M€.

Contact: Nathalie REVOL Nathalie.Revol@ens-lyon.fr - http://www.ens-lyon.fr/LIP/Arenaire/EVA-Flo/

http://www-list.cea.fr/ http://www.cnrs.fr/ http://webdali.univ-perp.fr/ http://eliaus.univ-perp.fr/ http://www.ens-lyon.fr/ http://www.inria.fr/ http://www.ens-lyon.fr/LIP http://www.univ-lyon1.fr/ http://www.univ-perp.fr/ http://www.universite-lyon.fr/