# INTERFLOP

## I. Pre-proposal's context, positioning and objective(s)

### Objectives

27 years ago, D. Golberg[1] stated that "Floating-Point (FP) arithmetic is considered an esoteric subject by many people". We have come a long way since then, thanks to the rise of many disasters (patriot missile failure, Ariane 5 explosion, sinking of the Sleipner platform...). Those disasters were mainly due to the use of FP numbers by negligent developers. This has contributed to the development of methodology, numerical algorithms and tools to make FP computations more robust. However, the tremendous increases in computational horsepower needed to address **new classes of problems** (ex: AI, drug simulation...), combined with the emergence of **new multicore processors** (ex: GPU, Tensor Core) and **new application-specific FP formats** (ex: unum, flexpoint, bfloat) demonstrate that we urgently need new solutions that scale with the dimensions of those applications, in order to avoid new disasters.

The InterFLOP project aims at providing a modular and scalable platform to both analyze and control the costs of FP behavior of today's real programs facing those new paradigms (bigger problems, new architectures, new representation formats). The results of existing tools often generate new questions such as: Is this small error guaranteed? May this numerical error occur and how to reduce it? Some more costly analyses bring pieces of answers to these questions, but also require more expertise. In InterFLOP, we propose new analyses and combinations of existing ones to address the challenge of providing a quick and precise numerical diagnosis requiring little user expertise. For that, InterFLOP will collect and combine information on numerical instabilities, catastrophic cancellations, unstable tests, build various statistical analyses of program executions at minimal overhead. InterFLOP capitalizes on the partners' expertise of UPVD (efficient propagation of accuracy during execution with FP-ANR), of LIP6 (stochastic Cadna analysis with C, C++, Fortran front-ends), of EDF, UVSQ and Intel (Verificarlo, Verrou analysis with executable and llvm front-ends), of CEA LIST (guaranteed analysis by abstract interpretation with Fluctuat affine forms and with C, C++ front-ends) and of ANEO (analysis of executable running on Windows or Linux servers). InterFLOP will 1) enlarge the class of possible applications by considering new front-ends and therefore new analyses 2) provide finer numerical analyses based on formalized composite analyses 3) verify the accuracy in the context of precision auto-tuning to make applications more efficient and robust 4) build statistical analyses tools and help the developer interpret the numerical behavior of program through graphical interpretation.

### Scientific hypothesis

Increasing capabilities and computational needs in many scientific fields lead to an increase in the size of the considered problems, data sets and consequently numerical errors. However, many existing codes have not considered their adaptation to accuracy requirements, which in turn makes their execution on future exascale architectures questionable. The raised issues are diverse: uncertainty and inaccuracy in the produced results, non-reproducibility, numerical instability, or loss of performance related to an inappropriate representation format. Academic and industrial users need tools to analyze, verify and optimize codes related to the intensive use of FP arithmetic. The French community has massively contributed in these fields by proposing various methods and tools. However, in order to address the new challenge previously described, we propose to combine their individual strength into a unique framework.

### Position of the project as it relates to the state of the art

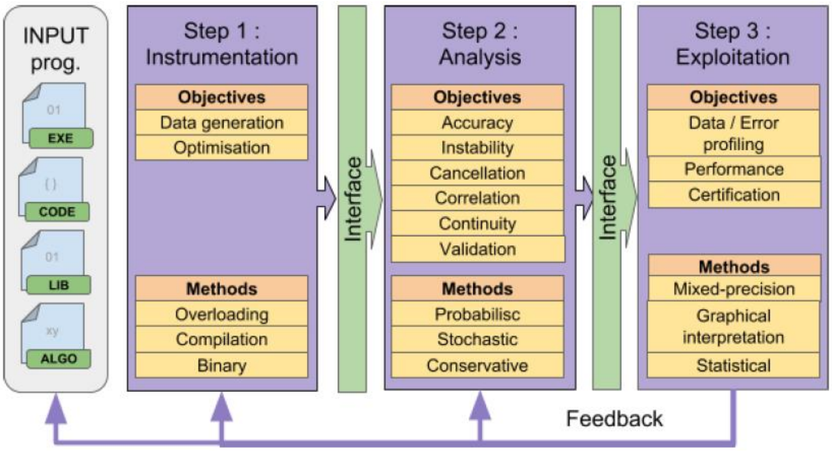
There exist several tools designed to test (CADNA, VERIFICARLO, VERROU), validate (FP-ANR, Fluctuat/FldLib) and optimize (PROMISE) numerical programs (other tools are listed at http://fpbench.org/). Those tools differ

[1] D. Golberg, "What every computer scientist should know about floating-point arithmetic".

by the offered compromises and level of information, but follow similar behavior: they take as input programs or descriptions, apply various numerical analyses and produce a numerical report. However, when analyzing real cases, an isolated view cannot reliably detect and resolve all potential numerical issues. InterFLOP will propose interoperable tools used for the analysis and optimization of FP calculus. By making those tools interoperable, it will be possible to take advantage of the properties and information specific to each of them in order to build composite and new analyses, inaccessible otherwise.

# Methodology to reach the objectives

InterFLOP relies on a modular environment enabling software and methodology sharing. In addition, it will optimize existing analyses by making them more precise, more pertinent for the user, faster while enriching them through new methods and visualization solutions. This figure summarizes the proposed environment.




To achieve this goal, the InterFLOP project is divided into 7 tasks which are described next. The task related to Project Coordination is not detailed in this pre-proposal.

## Task 1: Specification of the platform (Task Leader: EDF)

(i) One of the first sub-objectives is to rapidly propose an operational workflow. It is therefore necessary to define the type and the format of the data exchanged between each module in order to minimize bandwidth and memory usage while maximizing the amount of useful exchanged information. (ii) It will promote a modular, sustainable and open platform with a common exchange specification between the modules while minimizing the impact on performance. (iii) This would enable us to provide users with a software platform, an operational test environment that will be used after the project to promote the InterFLOP approach.

## Task 2: New front-end and mechanism to collect information (TL: ANEO)

This task aims at providing tools to enable mixed analysis between the different FP arithmetic of the project. The choice of the arithmetic will come from the availability of different back-ends: floating-point like FP-ANR, Monte-Carlo – MCA, stochastic – CESTAC and Taylor based or affine arithmetic. They will be made dynamically interoperable with the different front-ends dealing with executable (valgrind – linux x86, dynamorio – windows x86, unisim-vp – PowerPC, ARM), intermediate (llvm) and source code (C, C++).

## Task 3: New models for error estimation and composite analysis (TL: CEA)

The objective of this task is to define and implement composite analyses which will illustrate the added value of the software chain, taking as input existing analyses developed by InterFLOP's partners (MCA, CESTAC, FP-ANR, Taylor based). InterFLOP promotes an approach based on (i) an efficient search for instabilities (delta-debugging, origin of errors), (ii) guarantees of robustness/absence of instabilities on some parts of the code, and (iii) additions of generated code annotations expressing the impact of some algorithms on the accuracy (*e.g.* compensated algorithms) and enabling dynamic changes of the analysis mode.

## Task 4: Precision auto-tuning and verified computing (TL: LIP6)

This task aims at validating the accuracy of numerical results and automatically tuning the precision to achieve the desired accuracy for the result. (i) It will provide a systematic review of considered techniques/methods with respect to the advantages/drawbacks according to the scientific code studied. To automatically tune, the intermediate precision to reach a given final accuracy (ii) task 4 will extend the PROMISE tool so that it will be able to deal with a range of inputs, using automatic differentiation and interval arithmetic. However, this arithmetic is known to be costly, both in terms of memory and execution time. (iii) Therefore, task 4 will investigate how to reduce this overhead by proposing new compressed format based on FP-ANR concept.

### Task 5: Post-processing and statistical analysis of the results (TL: UVSQ)

This task tackles the problem of post-processing and analyzing the results of the InterFLOP chain through three axes: statistical analysis, instabilities tracking and visualization. (i) It will model the distribution of the error on a stochastic arithmetic, calculate the number of samples required to get a given level of confidence and analyze the sensitivity of these models. (ii) It will locate numerical errors, both spatially (function, variable, source line...) and temporally (iteration, call path...). (iii) It will provide graphical representations to interpret the InterFLOP results with different views (temporal, spatial, at function or variable level).

### Task 6: Application Cases (TL: ANEO/INTEL)

This task aims both at providing a feedback on the results from the other tasks with regards to their use in industrial applications and at proposing new analysis methodologies. The new methodologies and tools will be applied to draw conclusions about real codes. Considered applications (Yales2, AVBP, Abinit, Slang, EPX, Code_Aster, Telemac, quantitative analysis) have been chosen after prior attempts to run analysis using existing tools.

## Innovative nature of the project

This project takes the problem of numerical bug detection, software verification and validation to a new level, which is necessary to address issues that will be encountered with larger problems, new architectures, and new representation formats. Thanks to this project, numerical bug detection will be aided and guided through a unique interface at every step of the lifecycle of a software starting from its prototyping, testing, installation and operation. Industrial and academic users could then evaluate some basic compositions and develop customized ones for their own needs. Such composite analyses mixing execution (for speed on large codes), analysis (for precise diagnosis) and auto-tuning (to propose automatic enhancement) will be pioneer in the field and will be enriched with statistical and visual analysis.

## Ability of the project to address the research issues covered by the chosen research theme

Exaflopic architectures carry many hopes regarding potential problems that could be addressed. However, it is known that the associated new architectures, new representation formats, and the longer sequence of FP operations potentially executed in parallel will be the source of many numerical bugs with various level of seriousness.

Thanks to the proposed platform built on top of acknowledged tools (CADNA, VERIFICARLO, ...) and methods proposed by the InterFLOP's consortium, exascale software will be more robust numerically (**Fault tolerance**). In addition, many leading companies in the field of **Artificial Intelligence (AI)** are currently developing and promoting new **FP representation formats** (unum, flexpoint, bfloat) which will revamp the execution of software and will require adequate solutions to quantify the **uncertainty** due to finite precision and their propagation up to the final results. These measurements will be done by setting **statistical analysis** on the data produced and collected over several guided runs.

# II. Partenariat

InterFLOP is carried by 7 complementary partners which bring their own expertise, tools and methodology in Software Analysis (CEA LIST, LIP6), Compilation (UVSQ, INTEL, CEA LIST), Numerical Simulation (ANEO, EDF, INTEL), Statistics (UVSQ, LIP6), Computer Arithmetic (UPVD, LIP6), Parallelism (ANEO, INTEL, EDF), Computer Architecture (UPVD, INTEL, ANEO, CEA LIST) to make this project successful.

1. **LAMPS, UPVD (David DEFOUR)** has been conducting research at the intersection of High-Performance Computing, Computer Architecture and FP Arithmetic. He is involved in the development of software that increases precision and performance of programs. He has coordinated 5 industrial projects and was a scientific task-leader for 2 ANR.
2. **Li-Parad, UVSQ (Pablo OLIVEIRA, Devan SOHIER)** explores the interactions between High Performance Computing, Compilers, FP Arithmetic, stochastic process, applied to distributed systems and stochastic arithmetic. They are involved in the Verificarlo compiler.
3. **PEQUAN/LIP6, Sorbonne Université (Fabienne JEZEQUEL, Stef GRAILLAT, Jean-Luc LAMOTTE)** research topics include computer arithmetic, validated numeric and high-performance computing. They are known for the CADNA library, have large experience in developing software and strong relationship with industry.
4. **LSL, CEA List (Franck VEDRINE, Julien SIGNOLES, Yves LHUILLIER)** research topics include accuracy analyses by affine arithmetic (Fluctuat, fldlib), efficient memory model (frama-c) and transfer of analyses between source code and binary level.
5. **ACT, ANEO (Wilfried KIRSCHENMANN, Vivien MILLE)** conducts performance and numerical quality analysis in industrial platforms of different fields (quantitative analysis, radioprotection, seismic wave propagation, etc.).
6. **ECR, INTEL (Eric PETIT)** research topics include computer arithmetic, validated numeric and stochastic arithmetic. He is involved in the Verificarlo compiler.
7. **Analysis and Numerical Modeling, EDF R&D (Bruno LATHUILIERE, François FEVOTTE)** have developed numerous numerical solvers in industrial platforms for the simulation of various physical phenomena ranging from ultra-sonic waves propagation to neutron transport. They developed the VERROU tool.

# III. References related to the project


[1]. **D. Defour,** «FP-ANR: A representation format to handle floating-point cancellation at run-time», 25th IEEE Symposium on Computer Arithmetic (ARITH'23), 2018.

[2]. **F. Févotte, B. Lathuilière**, «Studying the Numerical Quality of an Industrial Computing Code: A Case Study on code_aster», 10th Workshop on Numerical Software Verification (NSV), 2017

[3]. Y. Chatelain, **P. de Oliveira Castro**, **E. Petit**, **D. Defour**, J. Bieder and M. Torrent. «VeriTracer: Context-enriched tracer for floating-point arithmetic analysis». 25th IEEE Symposium on Computer Arithmetic (ARITH'23), 2018.

[4]. **D. Sohier, P. de Oliveira Castro, F. Févotte, B. Lathuilière, E. Petit**, O. Jamond. «Confidence Intervals for Stochastic Arithmetic», (Preprint : https://arxiv.org/abs/1807.09655 ), 2018.

[5]. M. Jacquemin, S. Putot, **F. Védrine**, «A Reduced Product of Absolute and Relative Error Bounds for Floating-Point Analysis». 25th Static Analysis Symposium (SAS 2018), 2018.

[6]. **S. Graillat, F. Jézéquel**, R. Picot**, F. Févotte, B. Lathuilière**, «PROMISE: floating-point precision tuning with stochastic arithmetic», 17th international symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN 2016), 2016.

[7]. **F. Jézéquel, J.-L. Lamotte**, «I. Said, Estimation of numerical reproducibility on CPU and GPU, Annals of Computer Science and Information Systems», 2015 Federated Conference on Computer Science and Information Systems (FedCSIS), 2015.