

2025 Activity Report

RESEARCH CENTRE: Inria Lyon Centre

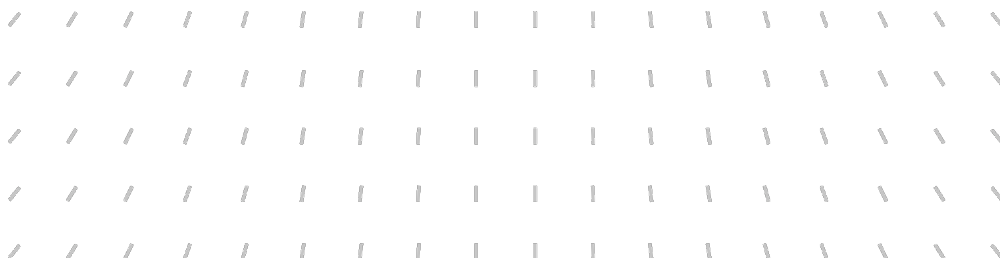
IN PARTNERSHIP WITH: Ecole normale supérieure de Lyon, Université Claude Bernard (Lyon 1), CNRS

Project-Team

PASCALINE

Computer Arithmetic, Computer Algebra and Formal Verification

In collaboration with Laboratoire de l'Informatique du Parallélisme (LIP)



Project-Team PASCALINE

Creation of the Project-Team: 2025 February 01

Each year, Inria research teams publish an Activity Report presenting their work and results over the reporting period. These reports follow a common structure, with some optional sections depending on the specific team. They typically begin by outlining the overall objectives and research programme, including the main research themes, goals, and methodological approaches. They also describe the application domains targeted by the team, highlighting the scientific or societal contexts in which their work is situated. The reports then present the highlights of the year, covering major scientific achievements, software developments, or teaching contributions. When relevant, they include sections on software, platforms, and open data, detailing the tools developed and how they are shared. A substantial part is dedicated to new results, where scientific contributions are described in detail, often with subsections specifying participants and associated keywords. Finally, the Activity Report addresses funding, contracts, partnerships, and collaborations at various levels, from industrial agreements to international cooperations. It also covers dissemination and teaching activities, such as participation in scientific events, outreach, and supervision. The document concludes with a presentation of scientific production, including major publications and those produced during the year.

Keywords

Computer sciences and digital sciences

- A4.5. – Formal method for verification, reliability, certification
- A6.2. – Scientific computing, Numerical Analysis & Optimization
- A7.1. – Algorithms
- A8.4. – Computer Algebra
- A8.10. – Computer arithmetic

Other research topics and application domains

- B6.1.1. – Software engineering
- B9.5.1. – Computer science

Contents

Project-Team PASCALINE	1
1 Team members, visitors, external collaborators	5
2 Overall objectives	5
3 Research program	6
3.1 Integrated tools for the development of mathematical functions	6
3.1.1 Better tools for analyzing accuracy	6
3.1.2 The Table-Maker’s Dilemma	7
3.1.3 Robustness issues in floating-point arithmetic	7
3.2 Any dimension, any precision	8
3.2.1 Tight error bounds for fast transforms	8
3.2.2 Symbolic approaches to rounding error analysis	8
3.2.3 Extended precision	9
3.2.4 Multiprecision libraries and their applications	9
3.2.5 Sparsification and quantization of artificial neural networks	10
3.3 A comprehensive chain for rigorous approximation	10
3.3.1 Symbolic summation and integration algorithms	10
3.3.2 Better complexities for structured and polynomial matrices	11
3.3.3 Rigorous polynomial approximations: towards certified spectral methods	11
3.4 Transverse topics	12
3.4.1 Reliability of tools	12
3.4.2 Standardization and dissemination	13
4 Application domains	13
5 Highlights of the year	13
5.1 Awards	13
6 Latest software developments, platforms, open data	13
6.1 Latest software developments	13
6.1.1 CoqInterval	13
6.1.2 Flocq	14
6.1.3 Gappa	14
6.1.4 Gfun	14
6.1.5 GNU MPFR	15
6.1.6 MPFI	15
6.1.7 AlgebraicSolving.jl	15
7 New results	15
7.1 Integrated tools for the development of mathematical functions	15
7.1.1 Emulation of the FMA and the correctly-rounded sum of three numbers in rounding-to-nearest floating-point arithmetic	15
7.1.2 Correctly rounded evaluation of a function: why, how, and at what cost?	16
7.1.3 FastTwoSum revisited	16
7.2 Any dimension, any precision	16
7.2.1 A rescaling-invariant Lipschitz bound based on path-metrics for modern ReLU network parameterizations	16
7.3 A comprehensive chain for rigorous approximation	16
7.3.1 A unified approach for degree bound estimates of linear differential operators	16
7.3.2 On deciding transcendence of power series	17
7.3.3 Effective asymptotics of combinatorial systems	17
7.3.4 Positivity proofs for linear recurrences through contracted cones	17

7.3.5	Optimal experimental design for partially observable pure birth processes	17
7.3.6	Computing roadmaps in unbounded smooth real algebraic sets	17
7.3.7	Algebraic and algorithmic methods for computing polynomial loop invariants	18
7.3.8	Beyond affine loops: a geometric approach to program synthesis	18
7.3.9	Efficient algorithms for maximal matroid degenerations and irreducible decompositions of circuit varieties	18
7.3.10	An exchange algorithm for optimizing both approximation and finite-precision evaluation errors in polynomial approximations	19
7.3.11	Bivariate polynomial reduction and elimination ideal over finite fields	19
7.3.12	Deterministic inversion of some matrices with displacement structure	19
7.3.13	Rational polynomial interpolation in FLINT	19
7.4	Transverse topics	20
7.4.1	Certifying the decidability of the word problem in monoids at large	20
7.4.2	Vulnerability found in perl	20
8	Bilateral contracts and grants with industry	20
8.1	Bilateral contracts with industry	20
8.1.1	ProofInUse-MERCE Collaboration	20
9	Partnerships and cooperations	20
9.1	European initiatives	20
9.1.1	H2020 projects	20
9.2	National initiatives	21
9.2.1	ANR NuSCAP project	21
9.2.2	ANR CNACS project	21
9.3	Regional initiatives	22
9.3.1	FIL inter-lab CONFIANTE project	22
10	Dissemination	22
10.1	Promoting scientific activities	22
10.1.1	Scientific events: organisation	22
10.1.2	Scientific events: selection	22
10.1.3	Journal	22
10.1.4	Invited talks	23
10.1.5	Leadership within the scientific community	23
10.1.6	Scientific expertise	23
10.1.7	Research administration	23
10.2	Teaching - Supervision - Juries	24
10.2.1	Teaching	24
10.2.2	Supervision	24
10.2.3	Juries	25
10.3	Popularization	25
10.3.1	Specific official responsibilities in science outreach structures	25
10.3.2	Participation in Live events	25
10.3.3	Others science outreach relevant activities	25
11	Scientific production	26
11.1	Publications of the year	26
11.2	Cited publications	27

1 Team members, visitors, external collaborators

Research Scientists

- Nicolas Brisebarre [Team leader, CNRS, Senior Researcher, from Feb 2025, HDR]
- Claude-Pierre Jeannerod [INRIA, Researcher, from Feb 2025]
- Vincent Lefèvre [INRIA, Researcher, from Feb 2025]
- Guillaume Melquiond [INRIA, Senior Researcher, from Oct 2025, Co-team leader, HDR]
- Jean-Michel Muller [CNRS, Senior Researcher, from Feb 2025, Emeritus since 1/12/2025, HDR]
- Rémi Prébet [INRIA, Researcher, from Feb 2025]
- Nathalie Revol [INRIA, Researcher]
- Bruno Salvy [INRIA, Senior Researcher, from Feb 2025]
- Gilles Villard [CNRS, Senior Researcher, from Feb 2025, HDR]

Faculty Member

- Nicolas Louvet [UNIV LYON I, Associate Professor, from Feb 2025]

Post-Doctoral Fellow

- Peio Borthelle [INRIA, Post-Doctoral Fellow, from Jun 2025]

PhD Students

- Giuseppe Carrino [ENS DE LYON, from Nov 2025]
- Louis Gaillard [ENS DE LYON, from Feb 2025]
- Tom Hubrecht [ENS DE LYON, from Feb 2025]
- Alaa Ibrahim [INRIA, from Feb 2025 until Nov 2025]

Technical Staff

- Joris Picot [ENS DE LYON, Engineer, from Feb 2025]

Administrative Assistant

- Chiraz Benamor [ENS DE LYON]

2 Overall objectives

The goal of the Pascaline team is to advance the fields of computer arithmetic and computer algebra, their interaction with formal verification, and their applications, in order to achieve unprecedented performance, accuracy, and reliability of numerical calculations.

To address the related challenges, we organize our work according to four directions:

1. New integrated tools for the development and the verification of functions from mathematical libraries: it is time to help the developer also!

2. Any dimension, any precision: from small formats for machine learning to guaranteed results for fast transforms or evaluation of neural networks. The main challenges are the dimensionality of the inputs and the quality of numerical evaluation.
3. A comprehensive chain for rigorous approximation: from a multiple integral to its polynomial approximation with a formally certified error.
4. Increasing the confidence in the tools through formal verification; disseminating our expertise through the writing of surveys and books. This transverse axis tackles the challenge of increasing the trust across all the first three research axes.

3 Research program

3.1 Integrated tools for the development of mathematical functions

While the IEEE 754 standard originally focused only on arithmetic operators that were meant to be implemented in hardware,¹ it now offers recommendation that mathematical functions such as \exp or $\sqrt{\cdot}$ should also be provided in a floating-point software environment. Indeed, developers take these functions for granted, due to their availability in the standard C library, and thus in most languages.

When designing the floating-point approximation of a mathematical function, two components are usually needed: a reduction of the input argument to an union of small intervals (and the corresponding reconstruction of the final result) and a polynomial (or rational) evaluation on these small intervals [55]. The design of these two components depends on the function, the input domain, and the accuracy and performance targets. In case one wants correct rounding, one might also need to build polynomial approximations at several working precisions. Each of those components needs not only to be written, but also specified, validated, and verified. Moreover, they cannot be designed in isolation, since the analyzed accuracy of a component impacts the design choices of the other ones.

The **MetaLibm project** aimed at hiding this complexity by making the generation of mathematical libraries as automatic as possible [51]. While this is a great approach for letting non-expert users design their own mathematical library (*e.g.*, for a custom precision or for an unusual function), it does not help expert users who are in the process of designing a state-of-the-art library à la **CRLibm** or **CORE-MATH** [65, 46]. Indeed, these users might have a clear idea of what the code should look like and thus might not be interested in what an automatic tool would produce. They would instead benefit more from a unified framework where they can design their functions, analyze them, and verify them.

3.1.1 Better tools for analyzing accuracy

A critical aspect in developing a floating-point approximation is the ability to analyze rounding errors and verify error bounds. For low-dimension and low-precision inputs (*e.g.*, the set of 32-bit floating-point numbers), exhaustive testing is computationally intensive but sufficient to fully analyze the accuracy of an algorithm, assuming one has a trusted reference computation of the ideal result. When considering a larger input space, the exhaustive approach is no longer a possibility. One instead has to mathematically bound all the inaccuracies that might occur during the execution of the algorithm: modeling error (*e.g.*, truncation of series, quantization of coefficients) and rounding errors.

This is usually a tedious exercise, leading to very long pen-and-paper proofs that challenge the sanity of both their authors and their reviewers. As a consequence, these proofs offer little confidence in the actual correctness of an algorithm. It is thus important to devise tools that are able to automatically generate such bounds. And since the correctness of these bounds can hardly be humanly reviewed, either these tools should be formally verified or they should produce a certificate that can be mechanically checked by a formal system. An example is the Gappa tool [26, §4.3], which is designed to analyze floating- and fixed-point algorithms and to generate proofs for the Rocq proof assistant.

We intend to keep on developing and improving tools that combine fine knowledge of floating-point arithmetic with mechanized reasoning like Gappa and CoqInterval [27, 38]. Ideally, these tools should progressively converge toward a single unified tool. Our aim is to design a methodology that would ease

¹Purely software implementations of the standard, however, have always existed, *e.g.*, Hauser's *SoftFloat*.

the development of a mathematical library such as CORE-MATH. This application is especially interesting because it relies on numerous low-level properties of floating-point arithmetic (*e.g.*, compensated arithmetic, Fast2Sum algorithm) that are tedious and error-prone to verify by hand.

3.1.2 The Table-Maker’s Dilemma

Ideally, floating-point approximations should offer *correct rounding*, that is, their result should be as if it was first computed with an infinite precision and range and then rounded to the nearest floating-point number, as is already the case for the basic arithmetic operators.²

Ziv’s approach works by running more and more precise (and thus costly) algorithms until one of them eventually succeeds in computing the correctly rounded result [70]. For such an approach to be usable in the context of critical systems, we need to be able to give beforehand a bound on the execution time and the memory footprint, which is directly related to the accuracy needed for correct rounding.

This is classically referred to as the *table-maker’s dilemma*. It can be phrased as the determination of the minimal distance between the graph of a mathematical function and the lattice \mathbb{Z}^2 . For univariate 32-bit (or less) floating-point functions, an exhaustive enumeration of all the inputs is sufficient. For univariate 64-bit functions that are regular enough, the enumeration can be made clever enough for the process to be fruitful [53, 67], even if the amount of computing power needed is still significant. Thanks to these results, the ARÉNAIRE team designed CRLibm, the first ever mathematical library with correctly rounded evaluation and bounded (and reasonable) delay. It is no longer maintained and the CORE-MATH library is now the reference library for evaluating functions with correct rounding.

For 64-bit functions that are numerically irregular (*e.g.*, for very large arguments of sin, cos, tan) as well as for 128-bit or multivariate functions, an exhaustive enumeration, how clever it is, is out of reach, due to the sheer size of the input domain. It would thus be quite interesting to develop alternative approaches to the ones developed earlier in order to reinforce the results for the 64-bit format and get additional results for new functions or larger domains. We have recently made a step forward with a new approach in the case of univariate transcendental functions [30]. An original combination of approximation theory, Euclidean lattice reduction and a trick of pre-reducing the lattice under consideration leads to a significant speed-up of the practical determination of bad cases for rounding. It yields, in particular, an improvement for the 128-bit case but does not give all the answers that we need. Therefore, we focus on the enhancement of this new approach, its generalization to the algebraic function case, and its extension to the multivariate case. Besides the algorithmic work, we will launch extensive computations to determine practical intermediate precisions that guarantee correctly rounded evaluations. Given the sheer amount of computing resources needed, it is also important to formally verify that the algorithms are correct *a priori*.

3.1.3 Robustness issues in floating-point arithmetic

Due to the increasingly wide availability of short floating-point formats (*i.e.*, low precision), underflows and overflows can now occur much more often than with traditional formats such as binary64. This undermines the adequacy of all the properties that have been proved assuming unbounded exponent ranges, which is a common assumption to simplify the verification of algorithms. As a consequence, we have to understand to which extent allowing for subnormal numbers in input or output may impact the error bounds obtained so far. We also need to design new algorithms, for which we can prove that spurious underflows and overflows never occur during intermediate computations. A main challenge here is to understand the intrinsic cost of achieving that kind of robustness for various numerical kernels, and some first results have already been obtained for Euclidean norm computations [52].

Another widespread assumption (though much safer than supposing that the exponent range is unbounded) is the use of the default rounding direction of the IEEE-754 standard, *i.e.*, round to nearest, when proving properties of low-level floating-point algorithms. Just like low-precision formats, unusual or underspecified rounding directions are becoming widely available in ML-driven hardware processing units. It is thus high time to develop a broader understanding of the behavior of numerical algorithms for such relaxed arithmetics, in the continuation of previous work on summation [24].

²Correct rounding not only guarantees the accuracy of these functions, but also the portability and reproducibility of their results. It also automatically guarantees useful properties such as the preservation of monotonicity, or the preservation of symmetries (*e.g.*, $\sin(-x) = -\sin(x)$).

3.2 Any dimension, any precision

Our first research axis has mostly focused on the design of scalar algorithms targeting a given accuracy (possibly correct rounding) under a set of constraints, *e.g.*, a given working floating-point format. Indeed, the more constraints, the better the tools we can provide to achieve a given result. This, however, leaves aside large families of algorithms.

Our second axis, that we consider here, is about algorithms working on data whose dimension is not fixed beforehand, or for which the restrictions on the working floating-point formats are relaxed. On the topic of arbitrary dimension, we are interested in the numerical behavior of fast arithmetic transforms such as the fast Fourier transform. We are also interested in the arithmetic issues caused by the implementation of artificial neural networks. As for providing an arithmetic that supports a larger precision, there are two different situations. Sometimes, one might need a bit more than the working precision (*e.g.*, 200 bits instead of 53), in which case it is worth having dedicated algorithms for this extended precision. But it might also happen that the precision depends on the inputs and is thus chosen at runtime, hence the need for arbitrary-precision algorithms. When dealing with an arbitrary precision, the usual methods for computing error bounds on an algorithm fall short, thus warranting new symbolic approaches.

3.2.1 Tight error bounds for fast transforms

The fast Fourier transform (FFT) and related algorithms play a central role in digital signal processing, but also in large-precision arithmetic. Indeed, these algorithms are used in fast polynomial and integer multiplication algorithms. When performing such an FFT-based multiplication, the final, exact, results (*e.g.*, the coefficients of the polynomial product) are integers, so they can be obtained by rounding to the nearest integers the intermediate values computed with floating-point arithmetic. Thus, the correctness of the algorithms depends on the proof that the absolute error on these computed values is less than $1/2$, which requires a careful error analysis.

Following some earlier work [32], we wish to provide tight error analyses for the most common variants of the fast Fourier transform and discrete cosine transforms. Having formal proofs of these bounds is essential if we wish to build reliable and efficient large-precision arithmetic on top of these transforms. We also consider other transforms such as fast rotations, Hadamard transforms, butterfly decompositions, and, more generally, the structured linear maps that have been used recently for designing faster neural network architectures [50, 62, 69]. While the efficiency gains made possible by fast transforms seem to be clear, a thorough understanding of how such floating-point algorithms behave is still missing. We thus focus on the derivation of tight error bounds for these key building blocks. In some cases, this also requires to make further progress on the analysis of lower-level routines such as those implementing complex arithmetic.

3.2.2 Symbolic approaches to rounding error analysis

When providing error bounds for a floating-point algorithm, their correctness (*i.e.*, the fact that they are mathematically true upper bounds) is critical for safety. But if these bounds were to largely overestimate the actual errors, the developer might unwillingly choose an algorithm that is too accurate, and thus slower than needed. To guarantee the tightness of such a bound when working at a fixed precision p (*e.g.*, $p = 53$ in binary64), one can build an input example for which the error committed by the algorithm comes close to that bound, or even attains it.

When analyzing the accuracy of an algorithm that works with formats of arbitrary precision p , the analysis now results in an error bound parameterized by p . To show the tightness of this error bound, one should symbolically describe a family of inputs whose numerical error converges to the bound as the precision grows larger. Such inputs are given as expressions parameterized by p , which can be viewed as symbolic floating-point numbers. This, however, requires the ability to run the algorithm on those inputs and, in particular, to compute the correctly-rounded sum, product, or ratio of two symbolic floating-point numbers. We have proposed a data type as well as some algorithms and their Maple implementation to perform correctly-rounded rational operations (+, −, ×, and ÷) on symbolic floating-point numbers, assuming an unbounded exponent range [47]. We plan to extend this work in order to characterize more cases in which non-rational functions can be handled, focusing first on the square root function. We also aim at incorporating bounds on the exponent in the data representation in order to be able to detect overflows and underflows and to handle subnormal numbers.

Symbolic computation can also be helpful in another way. Indeed, at least as a first stage in the analysis, the derivation of an error bound often looks like a tedious, technical, and lengthy propagation of inequalities, whose paper-and-pencil analysis is prone to errors. Allowing for the manipulation of expressions of large size, symbolic computation not only helps automate such analyses, but also makes it possible to keep tighter control on the correlations between rounding errors from one step to the next ones. Then, finding an upper bound on the error committed by a sequence of assignments with rounding errors can be expressed as a polynomial optimization problem, for which computer algebra algorithms are available. Research is needed to exploit the structure of the underlying polynomial system, as out-of-the box methods cannot cope with even moderate numbers of variables in the analysis. The system that we tackle is both triangular and very sparse and geometrically corresponds to a small tube around an algebraic set. This makes it susceptible to techniques based on regular chains, complemented by smt-solvers to prune the optimization tree.

3.2.3 Extended precision

For applications that require around 100 to 200 bits in critical parts,³ a library like GNU MPFR might not be the best choice. Indeed, its algorithms are designed to efficiently compute floating-point results with a very high precision. For intermediate precisions, it might be better to represent numbers as an unevaluated sum of two or more floating-point numbers and to design algorithms that manipulate such *double-word* [48] or *triple-word* [36] numbers. As the verification of these algorithms is long and tedious, the help of proof assistants is absolutely necessary to increase confidence [57].

This classical setting is worth revisiting with the advent of processors with large numbers of low-precision floating-point units. Indeed, while multi-word arithmetic is traditionally based on 64-bit numbers, the same principles could apply to 16-bit or even 8-bit floating-point numbers [43]. The original algorithms, however, were not designed with 4-bit significands in mind and might thus need to be re-engineered. The reduced exponent range of these new formats is also a concern, as it makes multi-word arithmetic more sensible to overflow and underflow. Even for the traditional 64-bit format, the situation is not set in stone. Indeed, the 2019 revision of the IEEE 754 standard mandates some new operations, which, if they were to be actually provided by hardware, would make multi-word arithmetic more efficient. To experiment with all these changes in the computer arithmetic landscape and to make the resulting algorithms widely available, we intend to resume the development of the Campary C++ library, which provides a multi-word arithmetic (+, −, ×, ÷, and $\sqrt{\cdot}$) that can be parameterized both by the length of the unevaluated sums and the underlying floating-point format [48].

While multi-word arithmetic based on floating-point words provides good performances when implementing mid-precision (ranging roughly from 100 to 200 bits as already mentioned), implementations based on hardware integers (such as in MPFR) are much more efficient in large precision. Finding the trade-offs between these two possible implementations of multi-precision floating-point arithmetic on different architectures is therefore an interesting question. There might also be a sweet spot at which a mix of floating-point and integer arithmetics is worth considering. This would be especially useful for mathematical libraries that aim at correct rounding such as CORE-MATH, as they juggle with increasingly large precisions.

3.2.4 Multiprecision libraries and their applications

The GNU MPFR library offers well-defined semantics for its floating-point algorithms, in particular correct rounding. A long term goal would be to rewrite MPFR functions in a domain specific language that fits the MPFR model and the expression of its (parameterized) algorithms, integrated with formal proofs. In particular, the fact that each MPFR variable has its own precision can give a large degree of freedom in the choice of parameters.

MPFI is a library for arbitrary precision interval arithmetic. It relies on MPFR for the floating-point arithmetic in arbitrary precision, with correct (outward) rounding. We are developing unit tests, for each operation or function provided by MPFI. This set of test cases needs to be expanded, so as to cover more difficult cases, such as bounds of the domain, neighborhoods of extrema or hard-to-round values that can be used as endpoints, for each function or operation. Furthermore, making MPFI compliant with the IEEE 1788-2015 standard for interval arithmetic implies a large and complete reworking of the implementation. Working on a standard-compliant version of MPFI also enriches our work on the revision of the standard.

³An example is the evaluation of the polynomial approximation for a function to be correctly rounded in binary64 arithmetic.

The development of libraries such as MPFR or MPFI must be accompanied by applications, to showcase their applicability. While MPFR is already widely used by external projects and its importance does not need to be demonstrated, work remains to be done for the use of interval arithmetic, the IEEE 1788-2015 interval standard, and the choice of the computing precision for interval computations. We illustrate and compare the use of MPFI with other possible approaches on various significant examples. These examples are both sufficiently specific so that we can completely and thoroughly handle them, and sufficiently general in the sense that they are well-studied and widely used. Such examples include the enclosures of the iterates of affine iterations, of the solutions of linear systems with interval coefficients, and other basic bricks in linear algebra. In all cases, the difficulty is to avoid large overestimations of these enclosures. In addition to determining adequate computing precisions, the straightforward use of MPFI will be compared with the use of other representations (such as affine forms) and of other bases (such as QR or SVD).

3.2.5 Sparsification and quantization of artificial neural networks

Artificial neural networks are used with success in many applications to approximate functions. In line with the works [34, 35, 42], we are interested in understanding their approximation power in practice and in theory. Two questions are of key importance: first, the comparison of the approximation properties of unquantized versus quantized neural networks, *i.e.*, networks with arbitrary real weights versus networks whose weights are constrained to a prescribed finite set (*e.g.*, binary64 numbers); second, the impact of the sparsification (that is canceling some chosen weights and biases) of a given neural network on its approximation properties. This work is done in collaboration with researchers from the OCKHAM team (R. Gribonval and E. Riccietti in particular).

3.3 A comprehensive chain for rigorous approximation

To handle most of the mathematical functions that we have to deal with, it is extremely convenient to define them as a solution of a differential equation, together with sufficient initial data. Hence, a first key step is to compute a differential equation that the function satisfies. So far, we have been major actors in producing such tools [64], and we plan to extend the scope of these methods to functions defined by multiple integrals or sums of elementary or special functions. To do so, we shall develop new symbolic summation and integration techniques. We shall also establish new complexity bounds and design more efficient algorithms in linear algebra, the importance of which goes far beyond our function implementation issues. Once we have these differential equations, a natural objective is to search for a closed form of the function solution. Although tempting, this approach is often of little relevance in practice. It is indeed much more fruitful to try to get an approximation (a polynomial, most of the time) of the solution, together with a certified approximation error, directly from the differential equation.

3.3.1 Symbolic summation and integration algorithms

Several of the algorithms for certified approximation that we develop in the team work at a level of generality that can accommodate any function specified by a linear differential equation with polynomial coefficients, together with sufficiently many initial conditions. Whatever the order of the equation and the degree of its coefficients, using computer algebra, we can generate certified approximations at arbitrary precision. The scope of these methods can be further extended by developing algorithms that produce such differential equations for functions originally defined in other ways. This is in particular possible for functions defined by multiple integrals or sums of elementary or special functions. A family of algorithms originating in Zeilberger's pioneering work in the 90's and progressively generalized and made more efficient leads to modern so-called 4th-generation algorithms for creative telescoping, to which we have contributed. One of our goals in the coming years is to produce more effort toward a reference implementation in this area. We do have implementations of parts of the algorithms, but there remain important issues with understanding the appropriate data-structures to reach a good efficiency, analyzing examples where the speed-up compared to previous algorithms is not as high as predicted by the theory, and possibly designing more specific algorithms that can take into account the extra structure some inputs possess, such as symmetry with respect to the parameters. All the algorithms, including the most recent ones, require in a crucial way efficient algorithms

for handling matrices with polynomial coefficients. This is a fundamental question with many applications beyond symbolic summation and integration, addressed below.

3.3.2 Better complexities for structured and polynomial matrices

Following the early results of Strassen in the 1960s, most problems of dense linear algebra over a field are essentially equivalent to matrix multiplication, in the algebraic complexity model. The last two decades have seen spectacular improvements — for which we have been important actors — with similar complexity results on matrices over principal domains. Central reductions (essentially equivalences) of problems to the product of polynomial or integer matrices have been established. In the dense case, one of the remaining hard questions is the computation of the characteristic polynomial of a univariate polynomial matrix [49].

Far beyond that, a broad observation today is that computational and experimental progress is seriously hampered by the fact that algorithms and software insufficiently exploit various structural aspects inherent to the problems at hand. Our motivation is to help remedy this situation concerning fast exact arithmetic for structured matrices whose coefficients lie in various domains such as the rational numbers, finite fields, and univariate polynomials.

Specifically, we design asymptotically fast algorithms for displacement structured matrices over an arbitrary commutative field. For efficiency, such matrices are encoded and manipulated via their displacement (that is, their image through a suitable linear operator) and cover classical structures such as those of Toeplitz or Vandermonde type. In some earlier work [28], we have obtained very satisfactory complexity results for a general definition of displacement that covers these classical structures as well as many intermediate situations appearing in important applications related to multivariate interpolation and approximation. The corresponding cost bounds are the best-known ones, but some of them require randomization. A first question is thus to understand the (im)possibility of derandomizing our fast algorithms. As a longer-term goal, we want to consider even larger classes of structured matrices [63], and to target better unified approaches for the displacement and quasi-separable structures, in relation with polynomial and sparse matrices.

We are also particularly focused on structured matrices whose entries are univariate polynomials. We have recently developed the fastest algorithms to date for the bivariate resultant and modular composition over an arbitrary commutative field [59, 68]. These results, which bring the first progress since the end of the 1970s, have highlighted links with structured polynomial matrices and fundamental problems on bivariate ideals — such as Gröbner bases and polynomial reduction, with bivariate interpolation and relation reconstruction. To improve complexity bounds and practical solutions, it is important to better understand the role of the baby steps/giant steps paradigm, which appears in many places, and to investigate new divide-and-conquer approaches [45, 58]. We especially plan to adopt a fine-grained complexity point of view to identify the difficulties more globally and establish new reductions between problems.

3.3.3 Rigorous polynomial approximations: towards certified spectral methods

In general, the functions that we are interested in are given by an expression tree whose nodes are solutions of differential equations. The differential equations range from linear to non-linear ones with uncertain parameters. We focus on the fast generation of *rigorous polynomial approximations* (RPAs) to these solutions, *i.e.*, a polynomial approximation together with a guaranteed approximation error bound. In particular, we need a fast and reliable arithmetic to combine these RPAs to the nodes of the tree representing the function. Once RPAs are computed, we can obtain certified quadratures, zero-finding or global optimization results for the functions under consideration. In particular, we use them to rigorously compute Abelian integrals in connection to Hilbert’s 16th problem [33], and to compute outer and inner approximations of reachable sets of states for control systems [41].

Key building blocks in our study of differential equations are the so-called *D-finite* functions, *i.e.*, solutions of linear differential equations with polynomial coefficients [66]. This class of functions includes many elementary (*e.g.*, \exp , \sin , \cos) and special functions (*e.g.*, Airy, Bessel, erf) commonly used in mathematical physics. D-finite functions have an efficient symbolic-numeric algorithmic treatment [64], which allows for the complexity study of validated-enclosure methods from a computer algebra point of view.

We design and implement algorithms to compute RPAs for D-finite functions, based on series expansions in certain orthogonal bases of functions (such as classical orthogonal polynomials, trigonometric or Bessel functions or the monomial basis) that are called *generalized Fourier series* [54]. When they satisfy linear

differential equations and under certain conditions, the coefficients of these series satisfy a linear recurrence with polynomial coefficients. In the case of Chebyshev expansions, we have shown that recurrences satisfied by the coefficients can be generated formally by a morphism of fractions of linear operators [23]. We will generalize this approach to a wide variety of generalized Fourier series, with a choice of basis left to the user, depending on the intended application.

When the method mentioned above succeeds, the computed linear recurrence can be used to efficiently generate the coefficients. A direct use of the recurrence to evaluate the coefficients may lead to huge numerical instabilities. Methods to overcome this problem in the Chebyshev setting were developed [61, 22]. We plan to extend this work to a much larger set of generalized Fourier series. To do so, we will set the problem in the framework of Functional Analysis (and more precisely, Operator Theory) and take advantage of the powerful results attached to the projection method [39]. This will provide a more robust foundation to the stability statements necessary to solve the recurrence and recover very good approximations to the actual value of the coefficients.

The two previous steps can be used as an oracle that presumably provides us with a very good approximation to the function we are looking for. We will extend our work in the Chebyshev basis setting [22, 29] to the generalized Fourier series case, via Picard's iteration or quasi-Newton contracting map, in order to yield a sharp bound on the approximation error automatically. This will be done in strong interaction with the formal proof part of the group.

All of this work paves the way for certified spectral methods [40, 31, 22].

3.4 Transverse topics

For the most part, the previous research axes involve software, whether they are mathematical libraries, algorithms for symbolic computations, tools to analyze numerical errors, and so on. Incorrect computations (*e.g.*, a poor polynomial approximation) might have adverse effects down the line. It is thus important to rise the level of confidence of the tools and libraries we propose. While we are committed to developing such high-quality software, we are well aware that this might not be the case for most developers of numerical software, be it by lack of time or expertise. So, we intend to keep disseminating our knowledge through the writing of surveys and books and the participation to standard committees.

3.4.1 Reliability of tools

The design of a mathematical function comprises several intricate steps, be it for dealing with rounding errors or finding a good enough polynomial approximation or finding the worst cases of the table-maker's dilemma. This often involves the use of external tools that might not have been designed for that specific purpose, and thus developers might unknowingly break some implicit assumption. For example, a tool that internally performs binary64 computations works usually well enough to design a binary32 approximation but would give bogus results if used to design a binary64 approximation.

This puts the developers in a precarious situation. One would usually turn to validation, *i.e.*, testing, to increase the confidence, but for the topics in which we are interested, except in the few cases where exhaustive testing is available, validation often falls short. For example, knowing the range of rounding errors on a subset of inputs, however well these inputs are chosen, is no indication of their range on the whole domain. There might always be some rogue inputs for which most elementary rounding errors contribute to the final rounding error in the same direction, thus causing a large difference with the average behavior.

Thus, validation needs to be complemented with verification, that is, the proof of a mathematical theorem that guarantees the property on the whole domain. This verification can take several forms. Ideally, one would prove that the tools are always correct (or at least prove that they properly detect when they would return incorrect results), and to reach the highest level confidence, this proof would be written in such a way that it can be mechanically checked by a computer, to make sure no corner cases were missed. The Rocq proof assistant is often used for that purpose [26].

Instead of proving the whole correctness of a tool, another approach is to instrument it in such a way that its results can be checked for correctness *a posteriori*. In other words, the tool would generate its result along with a certificate that can be mechanically checked. In that case, to reach the highest level of confidence, it is the checker itself that would be verified to be correct, which might be a more reasonable goal. Indeed, one no longer has to consider the correctness of the large amount of heuristic behaviors that might be implemented

in a tool. To this end, it is worth investigating whether some approaches found in computer algebra systems can make it easier to produce certificates that are amenable to be mechanically checked.

3.4.2 Standardization and dissemination

The rapid expansion of machine learning, especially deep neural networks, and its heavy use of floating-point arithmetic have brought a sudden influx of fresh and sometimes inexperienced users. This requires us to better educate them on these topics and to devise methods they can put into practice to raise the confidence in their numerical computations. Floating-point arithmetic itself has evolved a lot in the past few years, with the introduction of small-precision formats (such as Google's bf16, NVIDIA's tf32, and even 8-bit formats [60]) and with the advent of mixed-precision hardware and algorithms [44]. It is therefore important to prepare comprehensive treatments that reflect this evolution. This can be achieved through our participation to the standardization effort, *e.g.*, the third revision of the IEEE 754 standard on Floating-Point Arithmetic due in 2029, the second revision of the IEEE 1788-2015 standard for Interval Arithmetic, and the elaboration of the IEEE 3109 standard for Arithmetic Formats for Machine Learning (from 3 to 15 bits).

But our expertise on these topics can also be shared through the preparation and dissemination of reference texts about floating-point arithmetic such as what we recently did in [25], *e.g.*, a third edition of the *Handbook of Floating-Point Arithmetic* [56] and a fourth edition of *Elementary Functions: Algorithms and Implementation* [55]. New books are also on the way.

4 Application domains

Our expertise on validated numerics is useful to analyze and improve, and guarantee the quality of numerical results in a wide range of applications including:

- scientific simulation;
- global optimization;
- control theory.

Much of our work, in particular the development of correctly rounded elementary functions, is critical to the reproducibility of floating-point computations.

5 Highlights of the year

5.1 Awards

- Distinguished student author award at ISSAC 2025 for Alaa Ibrahim [14].
- Grand prix Inria-Académie des Sciences 2025 for Jean-Michel Muller.
- French open science award for free research software for the GNU MPFR software.

6 Latest software developments, platforms, open data

6.1 Latest software developments

6.1.1 CoqInterval

Keyword: Interval arithmetic

Functional Description: CoqInterval is a library for the Rocq proof assistant.

It provides several tactics for proving theorems on enclosures of real-valued expressions. The proofs are performed by an interval kernel which relies on a computable formalization of floating-point arithmetic in Rocq.

The Marelle team developed a formalization of rigorous polynomial approximation using Taylor models in Coq. In 2014, this library has been included in CoqInterval.

URL: <https://coqinterval.gitlabpages.inria.fr/>

Publications: [hal-04859533](#), [hal-04702129](#), [hal-04515714](#), [hal-04114233](#), [hal-03168208](#), [tel-02194683](#), [hal-01630143](#), [hal-01289616](#), [hal-00180138](#), [hal-01086460](#), [hal-00797913](#)

Contact: Guillaume Melquiond

Participant: 12 anonymous participants

6.1.2 Flocq

Keyword: Floating-point

Functional Description: The Flocq library for the Rocq proof assistant is a comprehensive formalization of floating-point arithmetic: core definitions, axiomatic and computational rounding operations, high-level properties. It provides a framework for developers to formally verify numerical applications. Flocq is currently used by the CompCert verified compiler to support floating-point computations.

URL: <https://flocq.gitlabpages.inria.fr/>

Publications: [hal-04114233](#), [hal-03233227](#), [hal-01632617](#), [hal-00862689](#), [hal-01091189](#), [hal-01091186](#), [hal-00743090](#), [inria-00534854](#)

Contact: Guillaume Melquiond

Participant: 3 anonymous participants

6.1.3 Gappa

Name: The Gappa tool for automated proofs of arithmetic properties

Keywords: Floating-point, Software Verification, Interval arithmetic

Functional Description: Gappa is a tool intended to help formally verifying numerical programs dealing with floating-point or fixed-point arithmetic. While Gappa is intended to be used directly, it can also act as a backend prover for the Why3 software verification platform or as an automatic tactic for the Rocq proof assistant.

URL: <https://gappa.gitlabpages.inria.fr/>

Publications: [hal-03233227](#), [tel-02194683](#), [inria-00070739](#), [inria-00344518](#), [inria-00070330](#), [tel-01094485](#), [inria-00071232](#), [inria-00432726](#), [ensl-00379167](#), [ensl-00200830](#), [hal-01110666](#), [hal-01110669](#), [hal-01632617](#)

Contact: Guillaume Melquiond

Participant: 2 anonymous participants

6.1.4 Gfun

Name: generating functions package

Keyword: Symbolic computation

Functional Description: Gfun is a Maple package for the manipulation of linear recurrence or differential equations. It provides tools for guessing a sequence or a series from its first terms, for manipulating rigorously solutions of linear differential or recurrence equations, using the equation as a data-structure.

URL: <http://perso.ens-lyon.fr/bruno.salvy/software/the-gfun-package/>

Contact: Bruno Salvy

6.1.5 GNU MPFR

Keywords: Multiple-Precision, Floating-point, Correct Rounding

Functional Description: GNU MPFR is an efficient arbitrary-precision floating-point library with well-defined semantics (copying the good ideas from the IEEE 754 standard), in particular correct rounding in 5 rounding modes. It provides about 100 mathematical functions, in addition to utility functions (assignments, conversions...). Special data (Not a Number, infinities, signed zeros) are handled like in the IEEE 754 standard. GNU MPFR is based on the mpn and mpz layers of the GMP library.

URL: <https://www.mpfr.org/>

Publications: [hal-01394289](#), [hal-01502326](#), [inria-00069930](#), [inria-00070174](#), [inria-00103655](#), [inria-00000026](#)

Contact: Vincent Lefèvre

Participants: Paul Zimmermann, Vincent Lefèvre, 2 anonymous participants

6.1.6 MPFI

Name: Multiple Precision Floating-point Interval

Keyword: Arithmetic

Functional Description: MPFI is a C library based on MPFR and GMP for arbitrary precision interval arithmetic.

URL: <https://gitlab.inria.fr/mpfi/mpfi>

Contact: Nathalie Revol

6.1.7 AlgebraicSolving.jl

Keywords: Computer algebra, Polynomial or analytical systems, Algebraic curve, Real solving

Functional Description: AlgebraicSolving.jl is a computer algebra package for the Julia programming language.

Its main features include algorithms for computing Gröbner bases over finite fields and for finding real solutions. The msolve library is the main workhorse.

URL: <https://algebraic-solving.github.io>

Contact: Remi Prebet

Participants: Christian Eder, Mohab Safey El Din, Rafael Mohr, Remi Prebet

Partners: Sorbonne Université, RPTU Kaiserslautern

7 New results

7.1 Integrated tools for the development of mathematical functions

7.1.1 Emulation of the FMA and the correctly-rounded sum of three numbers in rounding-to-nearest floating-point arithmetic

Stef Graillat (Sorbonne U.) and Jean-Michel Muller have introduced algorithms that make it possible to emulate the fused multiply-add (FMA) instruction and the correctly-rounded sum of three numbers (ADD3) in binary floating-point arithmetic, using only rounding-to-nearest floating-point additions, multiplications, and comparisons [5]. They also introduced variants of these algorithms that make it possible to compute the error of an ADD3 or FMA operation.

7.1.2 Correctly rounded evaluation of a function: why, how, and at what cost?

Nicolas Brisebarre, Guillaume Hanrot, Jean-Michel Muller, and Paul Zimmermann (Inria Nancy) have surveyed the various computational and mathematical issues and progress related to the problem of providing efficient correctly rounded elementary functions in floating-point arithmetic. This survey also aims at convincing the reader that a future standard for floating-point arithmetic should require the availability of a correctly rounded version of a well-chosen core set of elementary functions [3].

7.1.3 FastTwoSum revisited

The FastTwoSum algorithm is a classical way to evaluate the rounding error that occurs when adding two numbers in finite precision arithmetic. Starting with Dekker in the early 1970s, numerous floating-point analyses have been made of this algorithm, that are aimed at identifying sufficient conditions for the error to be computed exactly and, otherwise, at quantifying the quality of the error estimate thus produced. Claude-Pierre Jeannerod has revisited these two aspects of FastTwoSum [15]. This work has first provided new, less restrictive conditions for exactness, and showed that FastTwoSum performs an error-free transform in more general situations than those found so far in the literature. Second, when exactness cannot be guaranteed this work gives several error analyses of the output of FastTwoSum and showed that the bounds obtained are tight. In particular, this provides further insight into how the algorithm behaves when roundings other than ‘to nearest’ are used, or when the operands are reversed. This is joint work with Paul Zimmermann (Inria Centre at Université de Lorraine).

7.2 Any dimension, any precision

7.2.1 A rescaling-invariant Lipschitz bound based on path-metrics for modern ReLU network parameterizations

Robustness with respect to weight perturbations underpins guarantees for generalization, pruning and quantization. Existing guarantees rely on *Lipschitz bounds in parameter space*, cover only plain feed-forward MLPs, and break under the ubiquitous neuron-wise rescaling symmetry of ReLU networks. Antoine Gonon, Nicolas Brisebarre, Elisa Riccietti, and Rémi Gribonval have proved a new Lipschitz inequality expressed through the ℓ_1 -*path-metric* of the weights. The bound is (i) *rescaling-invariant* by construction and (ii) applies to any ReLU-DAG architecture with any combination of convolutions, skip connections, pooling, and frozen (inference-time) batch-normalization—thus encompassing ResNets, U-Nets, VGG-style CNNs, and more. By respecting the network’s natural symmetries, the new bound strictly sharpens prior parameter-space bounds and can be computed in two forward passes. To illustrate its utility, this work has derived a symmetry-aware pruning criterion and shown—through a proof-of-concept experiment on a ResNet-18 trained on ImageNet—that its pruning performance matches that of classical magnitude pruning, while becoming totally immune to arbitrary neuron-wise rescalings [13, 21].

7.3 A comprehensive chain for rigorous approximation

7.3.1 A unified approach for degree bound estimates of linear differential operators

Louis Gaillard has identified a common scheme in several existing algorithms addressing computational problems on linear differential equations with polynomial coefficients. These algorithms reduce to computing a linear relation between vectors obtained as iterates of a simple differential operator known as pseudo-linear map. This work focuses on establishing precise degree bounds on the output of this class of algorithms. It turns out that in all known instances (least common left multiple, symmetric product, etc), the bounds that are derived from the linear algebra step using Cramer’s rule are pessimistic. The gap with the behaviour observed in practice is often of one order of magnitude, and better bounds are sometimes known and derived from ad hoc methods and independent arguments. This work proposes a unified approach for proving output degree bounds for all instances of the class at once. The main technical tools come from the theory of realisations of matrices of rational functions and their determinantal denominators [12].

7.3.2 On deciding transcendence of power series

It is well known that algebraic power series are differentially finite (D-finite): they satisfy linear differential equations with polynomial coefficients. The converse problem, whether a given D-finite power series is algebraic or transcendental, is notoriously difficult. Alin Bostan, Bruno Salvy, and Michael F. Singer have proved that this problem is decidable by giving two theoretical algorithms and a transcendence test that is efficient in practice [18].

7.3.3 Effective asymptotics of combinatorial systems

Analytic combinatorics studies asymptotic properties of families of combinatorial objects using complex analysis on their generating functions. In their reference book on the subject, Flajolet and Sedgewick describe a general approach that allows one to derive precise asymptotic expansions starting from systems of combinatorial equations. In the situation where the combinatorial system involves only cartesian products and disjoint unions, the generating functions satisfy polynomial systems with positivity constraints for which many results and algorithms are known. Carine Pivoteau and Bruno Salvy have extended these results to the general situation. This produces an almost complete algorithmic chain going from combinatorial systems to asymptotic expansions. Thus, it is possible to compute asymptotic expansions of all generating functions produced by the symbolic method of Flajolet and Sedgewick when they have algebraic-logarithmic singularities (which can be decided), under the assumption that Schanuel's conjecture from number theory holds. That conjecture is not needed for systems that do not involve the constructions of sets and cycles [20].

7.3.4 Positivity proofs for linear recurrences through contracted cones

Behind many inequalities in combinatorics or involving special functions lies an elementary question: deciding the positivity of the terms of a sequence of real numbers defined by a linear recurrence with polynomial coefficients and initial conditions, known as a P-finite sequence. The question of positivity also arises in various fields such as code verification, numerical error analysis or modeling in biology. The work of Alaa Ibrahim focuses on finding efficient algorithms to automatically prove the positivity of P-finite sequences. This problem is difficult in general: even for recurrences with constant coefficients, decidability is linked to open problems in number theory. In the more general case of polynomial coefficients, Kauers and Pillwein (2010) established decidability for recurrences up to order 3, under certain assumptions about the roots of the characteristic polynomial and the initial conditions. The aim of this work is to extend these results in the polynomial framework to arbitrary orders. The contribution is based on a geometric approach: positivity is established by showing that the terms of the sequence remain in contracted convex cones, constructed using a generalization of the Perron-Frobenius theory to cones. This method leads to the decidability of positivity for a large class of recurrences, and is applicable to many examples from the literature [16, 6, 14].

7.3.5 Optimal experimental design for partially observable pure birth processes

Ali Eshragh, Matthew Skerritt, Bruno Salvy, and Thomas Mccallum have developed an efficient algorithm to find optimal observation times by maximizing the Fisher information for the birth rate of a partially observable pure birth process involving n observations. Partially observable implies that at each of the observation time points for counting the number of individuals present in the pure birth process, each individual is observed independently with a fixed probability p , modeling detection difficulties or constraints on resources. This work applies concepts and techniques from generating functions, using a combination of symbolic and numeric computation, to establish a recursion for evaluating and optimizing the Fisher information [4]. The recursion, while still computationally intensive, greatly improves on previously known computational methods which quickly became intractable even in the $n = 2$ case.

7.3.6 Computing roadmaps in unbounded smooth real algebraic sets

A roadmap for an algebraic set V defined by polynomials with coefficients in the field \mathbb{Q} of rational numbers is an algebraic curve contained in V whose intersection with all connected components of $V \cap \mathbb{R}^n$ is connected. These objects, introduced by Canny, can be used to answer connectivity queries over $V \cap \mathbb{R}^n$ provided that they are required to contain the finite set of query points $\mathcal{P} \subset V$; in this case, we say that the roadmap is

associated to (V, \mathcal{P}) . Rémi Prébet, Mohab Safey El Din, and Éric Schost have made effective a connectivity result they previously proved, to design a Monte Carlo algorithm which, on input (i) a finite sequence of polynomials defining V (and satisfying some regularity assumptions) and (ii) an algebraic representation of finitely many query points \mathcal{P} in V , computes a roadmap for (V, \mathcal{P}) [8]. This algorithm generalizes the nearly optimal one introduced by the last two authors by dropping a boundedness assumption on the real trace of V . The output size and running times of our algorithm are both polynomial in $(nD)^{n \log d}$, where D is the maximal degree of the input equations and d is the dimension of V . As far as we know, the best previously known algorithm dealing with such sets has an output size and running time respectively polynomial in $(n^{\log n} D)^{n \log n}$ and $(n^{\log n} D)^{n \log^2 n}$.

7.3.7 Algebraic and algorithmic methods for computing polynomial loop invariants

Loop invariants are properties of a program loop that hold both before and after each iteration of the loop. They are often used to verify programs and ensure that algorithms consistently produce correct results during execution. Consequently, generating invariants becomes a crucial task for loops. This work specifically focuses on polynomial loops, where both the loop conditions and the assignments within the loop are expressed as polynomials. Although computing polynomial invariants for general loops is undecidable, efficient algorithms have been developed for certain classes of loops. For instance, when all assignments within a while loop involve linear polynomials, the loop becomes solvable. Erdenebayar Bayarmagnai, Fatemeh Mohammadi, and Rémi Prébet have studied the more general case, where the polynomials can have arbitrary degrees. Using tools from algebraic geometry, they have presented two algorithms designed to generate all polynomial invariants within a given vector subspace, for a branching loop with nondeterministic conditional statements. These algorithms combine linear algebraic subroutines with computations on polynomial ideals. They differ depending on whether the initial values of the loop variables are specified or treated as parameters. Additionally, this work has presented a much more efficient algorithm for generating polynomial invariants of a specific form, applicable to all initial values. This algorithm avoids expensive ideal computations [2].

7.3.8 Beyond affine loops: a geometric approach to program synthesis

Ensuring software correctness remains a fundamental challenge in formal program verification. One promising approach relies on finding polynomial invariants for loops. Polynomial invariants are properties of a program loop that hold before and after each iteration. Generating such invariants is a crucial task in loop analysis, but it is undecidable in the general case. Recently, an alternative approach to this problem has emerged, focusing on synthesizing loops from invariants. However, existing methods only synthesize affine loops without guard conditions from polynomial invariants.

Erdenebayar Bayarmagnai, Fatemeh Mohammadi, and Rémi Prébet have addressed a more general problem, allowing loops to have polynomial update maps with a given structure, inequations in the guard condition, and polynomial invariants of arbitrary form. This work uses algebraic geometry tools to design and implement an algorithm that computes a finite set of polynomial equations whose solutions correspond to all nondeterministic branching loops satisfying the given invariants. Furthermore, it introduces a new class of invariants with a significantly more efficient algorithm. In other words, it reduces the problem of synthesizing loops to find solutions of multivariate polynomial systems with rational entries. This final step is handled using an SMT solver [10, 17].

7.3.9 Efficient algorithms for maximal matroid degenerations and irreducible decompositions of circuit varieties

Matroid theory provides a unifying framework for studying dependence across combinatorics, geometry, and applications ranging from rigidity to statistics. In this work, Emiliano Liwski, Fatemeh Mohammadi, and Rémi Prébet have studied circuit varieties of matroids, defined by their minimal dependencies, which play a central role in modeling determinantal varieties, rigidity problems, and conditional independence relations. They have introduced an efficient computational strategy for decomposing the circuit variety of a given matroid M , based on an algorithm that identifies its maximal degenerations. These degenerations correspond to the largest matroids lying below M in the weak order. Their framework yields explicit and

computable decompositions of circuit varieties that were previously out of reach for symbolic or numerical algebra systems. They applied their strategy to several classical configurations, including the Vámos matroid, the unique Steiner quadruple system $S(3, 4, 8)$, projective and affine planes, the dual of the Fano matroid, and the dual of the graphic matroid of $K_{3,3}$. In each case, they successfully computed the minimal irreducible decomposition of their circuit varieties [7].

7.3.10 An exchange algorithm for optimizing both approximation and finite-precision evaluation errors in polynomial approximations

The finite precision implementation of mathematical functions frequently depends on polynomial approximations. A key characteristic of this approach is that rounding errors occur both when representing the coefficients of the polynomial on a finite number of bits, and when evaluating it in finite precision arithmetic. Hence, to find a best polynomial, for a given fixed degree, norm and interval, it is necessary to account for both the approximation error and the floating-point evaluation error. While efficient algorithms were already developed for taking into account the approximation error, the evaluation part is usually a posteriori handled, in an ad-hoc manner. Denis Arzelier, Florent Bréhard, Tom Hubrecht, and Mioara Joldes have formulated a semi-infinite linear optimization problem whose solution is a best polynomial with respect to the supremum norm of the sum of both errors. This problem is then solved with an iterative exchange algorithm, which can be seen as an extension of the well-known Remez exchange algorithm. An open-source C implementation using the Sollya library has been developed and tested on several examples, which have been analyzed and compared against state-of-the-art Sollya routines [1].

7.3.11 Bivariate polynomial reduction and elimination ideal over finite fields

Given two polynomials a and b in $\mathbb{F}_q[x, y]$ which have no non-trivial common divisors, Gilles Villard has proved that a generator of the elimination ideal $\langle a, b \rangle \cap \mathbb{F}_q[x]$ can be computed in quasi-linear time. To achieve this, he has proposed a randomized algorithm of the Monte Carlo type which requires $(de \log q)^{1+o(1)}$ bit operations, where d and e bound the input degrees in x and in y respectively. The same complexity estimate applies to the computation of the largest degree invariant factor of the Sylvester matrix associated with a and b (with respect to either x or y), and of the resultant of a and b if they are sufficiently generic, in particular such that the Sylvester matrix has a unique non-trivial invariant factor. This work has exploited reductions to problems of minimal polynomials in quotient algebras of the form $\mathbb{F}_q[x, y]/\langle a, b \rangle$. By proposing a new method based on structured polynomial matrix division for computing with the elements of the quotient, this work has improved the best-known complexity bounds [9].

7.3.12 Deterministic inversion of some matrices with displacement structure

Claude-Pierre Jeannerod has considered the problem of inverting an $n \times n$ matrix with displacement rank α , and describe in particular a very simple way to reduce Toeplitz-like/Hankel-like inversion to block-Toeplitz/block-Hankel inversion in time $O(\alpha^{\omega-1}n)$, where ω is an exponent for dense, unstructured matrix multiplication [19]. This makes it possible to perform deterministically and in the same amount of time the inversion operation for a wide class of displacement-structured matrices. This improves upon previous inversion algorithms for such matrices, which were either randomized or slower.

7.3.13 Rational polynomial interpolation in FLINT

Rémi Prébet developed new algorithms in the FLINT library [37] for interpolating polynomials with rational coefficients. The methods rely on integer arithmetic to avoid costly intermediate fractions and include both fast and multimodular approaches. A unified interface selects the most efficient strategy using heuristics based on the input data. This work improves the performance and scope of exact rational polynomial computations in FLINT.

7.4 Transverse topics

7.4.1 Certifying the decidability of the word problem in monoids at large

While the word problem for monoids is undecidable in general, having a decision procedure for some finitely presented monoid of interest has numerous applications. As part of the ERC Fresco project, Assia Mahboubi, Florent Hivert, and Guillaume Melquiond have helped to design a toolbox for the Rocq proof assistant that can be used to verify the decidability of the word problem for a given monoid and, in some cases, to produce the corresponding decision procedure. As this verification can be computationally intensive, the toolbox heavily relies on proofs by reflection guided by an external oracle. This approach has been successfully used on several large presentations from the literature, as well as on a database of one million 1-relation monoids compiled by Reinis Cirpons, James Mitchell, and Finn Smith (University of St Andrews). The huge size of this database forced some unusual considerations onto the Rocq formalization, so that the formal proofs could be checked in a reasonable amount of time [11].

7.4.2 Vulnerability found in perl

In the search for the hardest-to-round values to solve the table-maker's dilemma for 64-bit functions, Vincent Lefèvre found a vulnerability in the perl threading code ([bug 23010](#), [report on oss-security](#), [CVE-2025-40909](#)). The bug was fixed by a perl developer in [perl 5.42](#). In addition to harmless errors caused by this bug, the script was potentially vulnerable, but this bug was not exploitable in the restricted use of the script, so that the results were not affected.

8 Bilateral contracts and grants with industry

8.1 Bilateral contracts with industry

8.1.1 ProofInUse-MERCE Collaboration

Participant: Guillaume Melquiond.

This bilateral contract is part of the ProofInUse effort. It is a collaboration between Inria and Mitsubishi Electric R&D Centre Europe (MERCE) in Rennes. It is funded by a bilateral contract of 6 years and 6 months from Nov 2019 to April 2026. This collaboration is mostly investigated by the Inria team Toccata (Saclay).

MERCE has strong and recognised skills in the field of formal methods. In the industrial context of the Mitsubishi Electric Group, MERCE has acquired knowledge of the specific needs of the development processes and meets the needs of the group in different areas of application by providing automatic verification and demonstration tools adapted to the problems encountered.

The objective of ProofInUse-MERCE is to significantly improve on-going MERCE tools regarding the verification of Programmable Logic Controllers and the verification of numerical C codes. The latter topic is the one investigated by the team Pascaline.

9 Partnerships and cooperations

9.1 European initiatives

9.1.1 H2020 projects

FRESCO, ERC Consolidator project

Participants: Peio Borthelle, Guillaume Melquiond.

Title: Fast and Reliable Symbolic Computation

Duration: November 2021 – October 2027

Coordinator: Assia Mahboubi (Gallinette)

Website

Using computers to formulate conjectures and consolidate proof steps pervades all mathematics fields, even the most abstract. Most computer proofs are produced by symbolic computations, using computer algebra systems. However, these systems suffer from severe, intrinsic flaws, rendering computational correction and verification challenging. The FRESCO project aims to shed light on whether computer algebra could be both reliable and fast. Researchers will disrupt the architecture of proof assistants, which serve as the best tools for representing mathematics in silico, enriching their programming features while preserving their compatibility with their logical foundations. They will also design novel mathematical software that should feature a high-level, performance-oriented programming environment for writing efficient code to boost computational mathematics.

9.2 National initiatives

9.2.1 ANR NuSCAP project

Participants: Nicolas Brisebarre, Guillaume Melquiond, Jean-Michel Muller, Joris Pi-cot, Bruno Salvy.

NuSCAP (Numerical Safety for Computer-Aided Proofs) is a five-year project started in February 2021. It is headed by Nicolas Brisebarre and, besides AriC, involves people from LIP lab, Galinette, Lfant, Stamp and Toccata INRIA teams, LAAS (Toulouse), LIP6 (Sorbonne Université), LIPN (Univ. Sorbonne Paris Nord) and LIX (École Polytechnique). Its goal is to develop theorems, algorithms and software, that will allow one to study a computational problem on all (or any) of the desired levels of numerical rigor, from fast and stable computations to formal proofs of the computations.

9.2.2 ANR CNACS project

Participant: Rémi Prébet.

Title: Certified Numerics for Algebraic Curves with Singularities

Duration: December 2024 – December 2028

Coordinator: Florent Brehard (CRISAL)

Website

The CNACS project aims to design symbolic-numerical algorithms for singular algebraic curves that are both efficient and reliable thanks to validated numerical computation techniques. The cornerstone will be a symbolic-numerical Newton-Puiseux algorithm for “unfolding” singularities and parameterizing branches using Puiseux series with interval coefficients (real or complex). We will then use it as a building block to improve the complexity of algorithms in effective algebraic geometry, such as connectivity queries and the topology of real curves, or the calculation of the monodromy of complex algebraic curves viewed as Riemann surfaces.

9.3 Regional initiatives

9.3.1 FIL inter-lab CONFIANTE project

Participant: Nathalie Revol.

CONFIANTE (CONstraint programming using Floating-point Interval Arithmetic: New heuristics, Tests, Experiments), 2025-2027, is funded by FIL (Fédération Informatique de Lyon). Algorithms solving CSPs (Constraint Satisfaction Problems), be they discrete or continuous, are *Branch-and-Propagate* or *Branch-and-Prune* ones. This project focuses on the *Branch* part, to identify heuristics that are successfully used to solve discrete CSPs, in order to adapt them to continuous CSPs. Preliminary work has been presented at the [SCAN 2025 conference](#). External participants: Christine Solnon (Emeraude team, INSA and CITI), and Christophe Jermann (U. Nantes, LS2N).

10 Dissemination

Participants: Nicolas Brisebarre, Claude-Pierre Jeannerod, Vincent Lefèvre, Nicolas Louvet, Guillaume Melquiond, Jean-Michel Muller, Joris Picot, Rémi Prébet, Nathalie Revol, Bruno Salvy, Gilles Villard.

10.1 Promoting scientific activities

10.1.1 Scientific events: organisation

Member of the organizing committees

- Chiraz Benamor, Nicolas Brisebarre, Claude-Pierre Jeannerod, Mioara Joldes (CNRS-LAAS), Vincent Lefèvre, Nicolas Louvet, Guillaume Melquiond and Rémi Prébet co-organized RAIM 2025 (16th Rencontres de l'Arithmétique en Informatique Mathématique) - A Tribute to Jean-Michel Muller.

10.1.2 Scientific events: selection

Chair of conference program committees

- Guillaume Melquiond was the program chair of the IEEE International Symposium in Computer Arithmetic in 2025.

Member of the conference program committees

- Claude-Pierre Jeannerod was a member of the program committee of the IEEE International Symposium in Computer Arithmetic, ARITH 2025.
- Guillaume Melquiond was a member of the program committee of the ACM SIGPLAN International Conference on Functional Programming, ICFP 2025.
- Guillaume Melquiond was a member of the program committee of Journées Francophones des Langages Applicatifs, JFLA 2026.

10.1.3 Journal

Member of editorial boards

- Nathalie Revol is associate editor of *IEEE Transactions on Computers*.
- Jean-Michel Muller is associate editor in chief of *IEEE Transactions on Emerging Topics in Computing*.

- Bruno Salvy is a member of the editorial boards of the *Journal of Symbolic Computation* and of *Annals of Combinatorics*.

10.1.4 Invited talks

- Jean-Michel Muller gave an invited talk at the Second French-Japanese Workshop on Numerical Computations, Paris (March 2025).
- Jean-Michel Muller gave an invited talk (online) at FPTalks2025 (July 2025).

10.1.5 Leadership within the scientific community

- Guillaume Melquiond, Jean-Michel Muller and Nathalie Revol are members of the steering committee of the IEEE International Symposium on Computer Arithmetic.

10.1.6 Scientific expertise

- Nicolas Brisebarre is a member of the scientific council of "Journées Nationales de Calcul Formel".
- Tom Hubrecht, Claude-Pierre Jeannerod, Vincent Lefèvre, Nicolas Louvet, Jean-Michel Muller, and Nathalie Revol participate to the biweekly video meetings of the 754-2029 Working Group (for the third revision of the IEEE Standard for Floating-Point Arithmetic, due in 2029).
- Claude-Pierre Jeannerod is a member of "Comité des Moyens Incitatifs" of the Lyon Inria research center.
- Vincent Lefèvre is an editor for the next revision of the IEEE 754 standard.
- Vincent Lefèvre participates in the revision of the ISO C standard via the C Floating Point Study Group.
- Guillaume Melquiond was a member of the hiring committee for a professor position at École Normale Supérieure Paris-Saclay.
- Nathalie Revol was a member of the hiring committee for Inria junior researcher positions at Centre Inria of the University of Rennes, and an expert for MSCA postoral fellowships of the European Union. As a member of the Inria Commission d'Évaluation, she was part of the CRHC and CRCH-8 promotion and the C3 bonuses committees and she took part in the assessment or creation procedures of Inria teams.
- Nathalie Revol participates to the biweekly video meetings of the 3109-2029 Working Group (for the elaboration of the IEEE Standard for Binary Floating-point Formats for Machine Learning, due in 2026).
- Bruno Salvy was a member of the hiring committee for a Maître de Conférences at Sorbonne Université.

10.1.7 Research administration

- Nicolas Brisebarre is co-head of GT ARITH (GDR IFM).
- Bruno Salvy was a member of the scientific council of the GDR IFM.
- Jean-Michel Muller is a member of the steering committee of the GDR-IFM.

10.2 Teaching - Supervision - Juries

10.2.1 Teaching

- Claude-Pierre Jeannerod, “Floating-point Arithmetic and Beyond”, 8h, M2, École Normale Supérieure de Lyon.
- Claude-Pierre Jeannerod, “Computer Algebra”, 20.5h, M2, ISFA, Lyon.
- Vincent Lefèvre, “Computer Arithmetic”, 10.5h, M2, ISFA, Lyon.
- Nicolas Louvet, “Algorithms and data structures”, 12h, L2, U. Lyon 1.
- Nicolas Louvet, “Application design and development”, 39h, L2, U. Lyon 1.
- Nicolas Louvet, “Operating systems”, 22h, L2, U. Lyon 1.
- Nicolas Louvet, “Program compilation and translation”, 22h, M1, U. Lyon 1.
- Nicolas Louvet, “Operating systems”, 20h, M2, U. Lyon 1.
- Guillaume Melquiond, “Floating-point Arithmetic and Beyond”, 12h, M2, École Normale Supérieure de Lyon.
- Jean-Michel Muller, “Floating-point Arithmetic and Beyond”, 10h, M2, École Normale Supérieure de Lyon.
- Nathalie Revol, “Scientific communication towards a large audience”, 39h, 4th year students, École Normale Supérieure de Lyon.
- Nathalie Revol, with Daria Pchelina, Damien Pous and Michael Rao, “Computer-assisted proof”, 4h, M2, École Normale Supérieure de Lyon.
- Bruno Salvy and Gilles Villard, “Computer Algebra”, 20h, M1, École Normale Supérieure de Lyon.
- Bruno Salvy and Gilles Villard, with Alin Bostan, “Modern Algorithms in Symbolic Summation and Integration”, 32h, M2, École Normale Supérieure de Lyon.

10.2.2 Supervision

- PhD: Alaa Ibrahim, “Automatic positivity proofs for P-finite sequences”, supervised by Alin Bostan, Mohab Safey El Din and Bruno Salvy, defended in Dec. 2025.
- PhD in progress: Louis Gaillard, “Bornes fines et algorithmes efficaces pour les équations différentielles linéaires”, supervised by Alin Bostan, Bruno Salvy and Gilles Villard, since Sep. 2024.
- PhD in progress: Tom Hubrecht, “Autour d’une bibliothèque de calcul numérique à plusieurs niveaux de confiance”, supervised by Nicolas Brisebarre, since Sep. 2024.
- PhD in progress: Giuseppe Carrino, “Numerical frugality in optimization: Newton’s methods and variable precision strategies”, supervised by Nicolas Brisebarre, Elisa Riccietti (E.P.I. Ockham) and Théo Mary (CNRS, LIP6, Sorbonne U.), since Nov. 2025.
- PhD in progress: Erdenebayar Bayarmagnai, “Formal Analysis and Synthesis of Polynomial Programs”, supervised by Fatemeh Mohammadi (KU Leuven, Belgium) and Rémi Prébet, since Sep. 2023.

10.2.3 Juries

- Claude-Pierre Jeannerod and Nathalie Revol were members of the M2 jury of ENS de Lyon.
- Jean-Michel Muller chaired the committee for the PhD defense of Orégane Desrentes (INSA Lyon), September 17, 2024.
- Jean-Michel Muller was a member of the committee for the Habilitation defense of Théo Mary (Sorbonne U.), March 12, 2025.
- Rémi Prébet was a member of the committee for the MSc thesis of Lisa Vandebrouck, supervised by Emiliano Liwski and Fatemeh Mohammadi, KU Leuven.
- Nathalie Revol was a member of the committee for the MSc thesis of Johnatan Garcia, supervised by Martine Ceberio and Christoph Lauter, University of Texas at El Paso.

10.3 Popularization

10.3.1 Specific official responsibilities in science outreach structures

- Nathalie Revol is the scientific editor of the Web magazine [Interstices](#) and she co-heads the “commission médiation” of the LIP laboratory since October 2025.

10.3.2 Participation in Live events

- Nicolas Brisebarre has been a scientific consultant for “Les maths et moi”, a one-man show by Bruno Martins since 2020. He also takes part to Q & A sessions with the audience after some shows.
- Nathalie Revol has received a class of high-school pupils from Lycée Le Valentin (Bourg-les-Valence), she has introduced research careers to Lycée Descartes (Saint-Genis-Laval), she has been part of a “MixTeen coding goûter” at MMI for pupils between 7 and 12 years old, she has taken part in a speed-meeting at Lycée La Martinière-Duchère (Lyon), she has given a talk for “Tournoi Français des Jeunes Mathématiciennes et Mathématiciens (Lyon), and she has been part of an “enquête scientifique” for primary school pupils (MMI): around 300 pupils.
- In relation with the “Binôme” project initiated in 2024, Nathalie Revol accompanied the 5 performances of the play at La Reine Blanche theater (Paris), Lycée Paul-Émile Victor (Osny) and L'Assemblée fabrique artistique (Lyon): around 330 audience members.

10.3.3 Others science outreach relevant activities

Actions related to parity and intended for a large audience:

- As an incentive for high-school girls to choose scientific careers, Nathalie Revol co-organized half a day for 45 high-school girls for the "Sciences : un métier de femmes" event, and a day “Filles et maths-info” for about 100 pupils.
- Nathalie Revol was invited to a table ronde about “Femmes et sciences” at médiathèque de Brignais, and a table ronde about “Femmes et numérique” for the master “genre” of Université Lyon 2; she has been portrayed for Collège des Sociétés Savantes Académiques de France and for the Inria video series “Archétypes”, both about female researchers.

11 Scientific production

11.1 Publications of the year

International journals

- [1] D. Arzelier, F. Bréhard, T. Hubrecht and M. Joldes. ‘An Exchange Algorithm for Optimizing both Approximation and Finite-Precision Evaluation Errors in Polynomial Approximations’. In: *ACM Transactions on Mathematical Software* (2025). DOI: [10.1145/3770066](https://doi.org/10.1145/3770066). URL: <https://hal.science/hal-04709615>. In press (cit. on p. 19).
- [2] E. Bayarmagnai, F. Mohammadi and R. Prébet. ‘Algebraic and algorithmic methods for computing polynomial loop invariants’. In: *Journal of Symbolic Computation* 135 (31st Dec. 2025), p. 102551. DOI: [10.1016/j.jsc.2025.102551](https://doi.org/10.1016/j.jsc.2025.102551). URL: <https://hal.science/hal-05446629> (cit. on p. 18).
- [3] N. Brisebarre, G. Hanrot, J.-M. Muller and P. Zimmermann. ‘Correctly rounded evaluation of a function: why, how, and at what cost?’ In: *ACM Computing Surveys* 58.1 (Jan. 2026). DOI: [10.1145/3747840](https://doi.org/10.1145/3747840). URL: <https://hal.science/hal-04474530> (cit. on p. 16).
- [4] A. Eshragh, M. Skerritt, B. Salvy and T. Mccallum. ‘Optimal experimental design for partially observable pure birth processes’. In: *PLoS ONE* 20.8 (29th Aug. 2025), e0328707. DOI: [10.1371/journal.pone.0328707](https://doi.org/10.1371/journal.pone.0328707). URL: <https://inria.hal.science/hal-05232003> (cit. on p. 17).
- [5] S. Graillat and J.-M. Muller. ‘Emulation of the FMA and the correctly-rounded sum of three numbers in rounding-to-nearest floating-point arithmetic’. In: *Numerische Mathematik* (Sept. 2025). DOI: [10.1007/s00211-025-01487-2](https://doi.org/10.1007/s00211-025-01487-2). URL: <https://hal.science/hal-04575249> (cit. on p. 15).
- [6] A. Ibrahim and B. Salvy. ‘Positivity proofs for linear recurrences through contracted cones’. In: *Journal of Symbolic Computation* 132 (Jan. 2026), p. 102463. DOI: [10.1016/j.jsc.2025.102463](https://doi.org/10.1016/j.jsc.2025.102463). URL: <https://inria.hal.science/hal-05232016> (cit. on p. 17).
- [7] E. Liwski, F. Mohammadi and R. Prébet. ‘Efficient Algorithms for Maximal Matroid Degenerations and Irreducible Decompositions of Circuit Varieties’. In: *Journal of Algebra* 691 (1st Apr. 2025), pp. 526–576. DOI: [10.1016/j.jalgebra.2025.12.003](https://doi.org/10.1016/j.jalgebra.2025.12.003). URL: <https://hal.science/hal-05043051> (cit. on p. 19).
- [8] R. Prébet, M. Safey El Din and É. Schost. ‘Computing roadmaps in unbounded smooth real algebraic sets II: algorithm and complexity’. In: *Journal of Symbolic Computation* 135 (July 2026), p. 102532. DOI: [10.1016/j.jsc.2025.102532](https://doi.org/10.1016/j.jsc.2025.102532). URL: <https://hal.science/hal-04439518>. In press (cit. on p. 18).
- [9] G. Villard. ‘Bivariate polynomial reduction and elimination ideal over finite fields’. In: *Journal of Symbolic Computation* 127 (2025), p. 102367. DOI: [10.1016/j.jsc.2024.102367](https://doi.org/10.1016/j.jsc.2024.102367). URL: <https://hal.science/hal-04542799> (cit. on p. 19).

International peer-reviewed conferences

- [10] E. Bayarmagnai, F. Mohammadi and R. Prébet. ‘Beyond Affine Loops: A Geometric Approach to Program Synthesis’. In: *SOAP ’25: Proceedings of the 14th ACM SIGPLAN International Workshop on the State Of the Art in Program Analysis*. SOAP ’25: 14th ACM SIGPLAN International Workshop on the State Of the Art in Program Analysis. Seoul, South Korea: Association for Computing Machinery, New York, NY, United States, 13th June 2025, pp. 1–7. DOI: [10.1145/3735544.3735581](https://doi.org/10.1145/3735544.3735581). URL: <https://hal.science/hal-05127957> (cit. on p. 18).
- [11] R. Cirpons, F. Hivert, A. Mahboubi, G. Melquiond, J. D. Mitchell and F. Smith. ‘Certifying the Decidability of the Word Problem in Monoids at Large’. In: *CPP ’26: Proceedings of the 15th ACM SIGPLAN International Conference on Certified Programs and Proofs*. Certified Programs and Proofs. Rennes, France: ACM, Jan. 2026, pp. 128–142. DOI: [10.1145/3779031.3779101](https://doi.org/10.1145/3779031.3779101). URL: <https://hal.science/hal-05448783> (cit. on p. 20).

- [12] L. Gaillard. ‘A unified approach for degree bound estimates of linear differential operators’. In: *ISSAC ’25: Proceedings of the 2025 International Symposium on Symbolic and Algebraic Computation*. ISSAC ’25: International Symposium on Symbolic and Algebraic Computation. Guanajuato Mexico, Mexico: ACM, 28th July 2025, pp. 16–24. DOI: [10.1145/3747199.3747542](https://doi.org/10.1145/3747199.3747542). URL: <https://hal.science/hal-04977634> (cit. on p. 16).
- [13] A. Gonon, N. Brisebarre, E. Riccietti and R. Gribonval. ‘A Rescaling-Invariant Lipschitz Bound Based on Path-Metrics for Modern ReLU Network Parameterizations’. In: *Proceedings of the 42nd International Conference on Machine Learning (ICML 2025)*. International Conference on Machine Learning. Vancouver (BC), Canada, 13th July 2025. URL: <https://hal.science/hal-04584311> (cit. on p. 16).
- [14] A. Ibrahim. ‘Positivity Proofs for Linear Recurrences with Several Dominant Eigenvalues’. In: *ISSAC ’25: International Symposium on Symbolic and Algebraic Computation*. Guanajuato Mexico, France: ACM, 28th July 2025, pp. 224–232. DOI: [10.1145/3747199.3747565](https://doi.org/10.1145/3747199.3747565). URL: <https://inria.hal.science/hal-05446330> (cit. on pp. 13, 17).
- [15] C.-P. Jeannerod and P. Zimmermann. ‘FastTwoSum revisited’. In: *Proceedings of 32nd IEEE Symposium on Computer Arithmetic*. 32nd IEEE Symposium on Computer Arithmetic (ARITH 2025). El Paso, TX, United States, 4th May 2025. URL: <https://inria.hal.science/hal-04875749> (cit. on p. 16).

Doctoral dissertations and habilitation theses

- [16] A. Ibrahim. ‘Automatic positivity proofs for P-finite sequences’. Ecole normale supérieure de lyon - ENS LYON, 21st Nov. 2025. URL: <https://theses.hal.science/tel-05443899> (cit. on p. 17).

Reports & preprints

- [17] E. Bayarmagnai, F. Mohammadi and R. Prébet. *From Affine to Polynomial: Synthesizing Loops with Branches via Algebraic Geometry*. 3rd Oct. 2025. URL: <https://hal.science/hal-05296092> (cit. on p. 18).
- [18] A. Bostan, B. Salvy and M. F. Singer. *On deciding transcendence of power series*. 23rd Apr. 2025. URL: <https://hal.science/hal-05172426> (cit. on p. 17).
- [19] C.-P. Jeannerod. *Deterministic inversion of some matrices with displacement structure*. 30th Dec. 2025. URL: <https://inria.hal.science/hal-05435741> (cit. on p. 19).
- [20] C. Pivoteau and B. Salvy. *Effective Asymptotics of Combinatorial Systems*. 27th Aug. 2025. URL: <https://hal.science/hal-05227398> (cit. on p. 17).

Software

- [21] [SW] A. Gonon, N. Brisebarre, E. Riccietti and R. Gribonval, *Code for reproducible research - A rescaling-invariant Lipschitz bound using path-metrics* version 0.0.1, 13th June 2025. LIC: BSD 3-Clause "New" or "Revised" License. HAL: [hal-05088847](https://hal.science/hal-05088847), URL: <https://hal.science/hal-05088847>, vcs: https://github.com/agonon/pathnorm_toolkit, SWHID: [sw:1:dir:3ef1659778414497b9fc2469b8afad02c6eb9e68;origin=https://hal.archives-ouvertes.fr/hal-05088847;visit=sw:1:snp:4315baa0c8e513fa42b84634266d757d5e44fcbd;anchor=sw:1:rel:147f1cfa0d54dace5a3c59dad43d04b055217f5b;path=/](https://sw.hal.archives-ouvertes.fr/hal-05088847;visit=sw:1:snp:4315baa0c8e513fa42b84634266d757d5e44fcbd;anchor=sw:1:rel:147f1cfa0d54dace5a3c59dad43d04b055217f5b;path=/) (cit. on p. 16).

11.2 Cited publications

- [22] A. Benoit, M. Joldes and M. Mezzarobba. ‘Rigorous uniform approximation of D-finite functions using Chebyshev expansions’. In: *Mathematics of Computation* 86.305 (2017), pp. 1303–1341. DOI: [10.1090/mcom/3135](https://doi.org/10.1090/mcom/3135) (cit. on p. 12).
- [23] A. Benoit and B. Salvy. ‘Chebyshev Expansions for Solutions of Linear Differential Equations’. In: *34th International Symposium on Symbolic and Algebraic Computation*. ACM Press, 2009, pp. 23–30. DOI: [10.1145/1576702.1576709](https://doi.org/10.1145/1576702.1576709) (cit. on p. 12).

- [24] S. Boldo, S. Graillat and J.-M. Muller. ‘On the robustness of the 2Sum and Fast2Sum algorithms’. In: *ACM Transactions on Mathematical Software* 44.1 (July 2017), 4:1–4:14. doi: [10.1145/3054947](https://doi.org/10.1145/3054947) (cit. on p. 7).
- [25] S. Boldo, C.-P. Jeannerod, G. Melquiond and J.-M. Muller. ‘Floating-point Arithmetic’. In: *Acta Numerica* 32 (May 2023), pp. 203–290. doi: [10.1017/S0962492922000101](https://doi.org/10.1017/S0962492922000101) (cit. on p. 13).
- [26] S. Boldo and G. Melquiond. *Computer Arithmetic and Formal Proofs. Verifying Floating-point Algorithms with the Coq System*. ISTE Press – Elsevier, 2017 (cit. on pp. 6, 12).
- [27] S. Boldo and G. Melquiond. ‘Some Formal Tools for Computer Arithmetic: Flocq and Gappa’. In: *28th IEEE International Symposium on Computer Arithmetic*. Ed. by M. Joldes and F. Lamberti. June 2021 (cit. on p. 6).
- [28] A. Bostan, C.-P. Jeannerod, C. Moulleron and É. Schost. ‘On Matrices With Displacement Structure: Generalized Operators and Faster Algorithms’. In: *SIAM Journal on Matrix Analysis and Applications* 38.3 (2017), pp. 733–775. doi: [10.1137/16M1062855](https://doi.org/10.1137/16M1062855) (cit. on p. 11).
- [29] F. Bréhard, N. Brisebarre and M. Joldes. ‘Validated and numerically efficient Chebyshev spectral methods for linear ordinary differential equations’. In: *ACM Transactions on Mathematical Software* 44.4 (July 2018), 44:1–44:42. doi: [10.1145/3208103](https://doi.org/10.1145/3208103) (cit. on p. 12).
- [30] N. Brisebarre and G. Hanrot. ‘Integer points close to a transcendental curve and correctly-rounded evaluation of a function’. 2023. URL: <https://hal.science/hal-03240179> (cit. on p. 7).
- [31] N. Brisebarre and M. Joldes. ‘Chebyshev interpolation polynomial-based tools for rigorous computing’. In: *35th International Symposium on Symbolic and Algebraic Computation*. ACM. ACM Press, 2010, pp. 147–154. doi: [10.1145/1837934.1837966](https://doi.org/10.1145/1837934.1837966) (cit. on p. 12).
- [32] N. Brisebarre, M. Joldes, J.-M. Muller, A.-M. Nanes and J. Picot. ‘Error Analysis of Some Operations Involved in the Cooley-Tukey Fast Fourier Transform’. In: *ACM Transactions on Mathematical Software* 46.2 (May 2020). doi: [10.1145/3368619](https://doi.org/10.1145/3368619) (cit. on p. 8).
- [33] C. Christopher and C. Li. *Limit Cycles of Differential Equations*. Advanced Courses in Mathematics. CRM Barcelona. Birkhäuser Verlag, Basel, 2007. doi: [10.1007/978-3-7643-8410-4](https://doi.org/10.1007/978-3-7643-8410-4) (cit. on p. 11).
- [34] R. DeVore, B. Hanin and G. Petrova. ‘Neural network approximation’. In: *Acta Numerica* 30 (2021), pp. 327–444. doi: [10.1017/S0962492921000052](https://doi.org/10.1017/S0962492921000052) (cit. on p. 10).
- [35] D. Elbrächter, D. Perekrestenko, P. Grohs and H. Bölcskei. ‘Deep Neural Network Approximation Theory’. In: *IEEE Transactions on Information Theory* 67.5 (2021), pp. 2581–2623. doi: [10.1109/TIT.2021.3062161](https://doi.org/10.1109/TIT.2021.3062161) (cit. on p. 10).
- [36] N. Fabiano, J.-M. Muller and J. Picot. ‘Algorithms for Triple-Word Arithmetic’. In: *IEEE Transactions on Computers* 68.11 (2019), pp. 1573–1583. doi: [10.1109/TC.2019.2918451](https://doi.org/10.1109/TC.2019.2918451) (cit. on p. 9).
- [37] The FLINT team. *FLINT: Fast Library for Number Theory*. Version 3.4.0, <https://flintlib.org>. 2025 (cit. on p. 19).
- [38] P. Geneau de Lamarlière, G. Melquiond and F. Faissolle. ‘Slimmer Formal Proofs for Mathematical Libraries’. In: *30th IEEE International Symposium on Computer Arithmetic*. Ed. by T. Drane and A. Volkova. Portland, OR, USA, Sept. 2023. doi: [10.1109/ARITH58626.2023.00026](https://doi.org/10.1109/ARITH58626.2023.00026) (cit. on p. 6).
- [39] I. Gohberg, S. Goldberg and M. A. Kaashoek. *Basic classes of linear operators*. Birkhäuser Verlag, Basel, 2003. doi: [10.1007/978-3-0348-7980-4](https://doi.org/10.1007/978-3-0348-7980-4) (cit. on p. 12).
- [40] D. Gottlieb and S. A. Orszag. *Numerical analysis of spectral methods: theory and applications*. CBMS-NSF Regional Conference Series in Applied Mathematics, No. 26. SIAM, Philadelphia, Pa., 1977 (cit. on p. 12).
- [41] É. Goubault and S. Putot. ‘Inner and Outer Reachability for the Verification of Control Systems’. In: *22nd ACM International Conference on Hybrid Systems: Computation and Control*. 2019. doi: [10.1145/3302504.3311794](https://doi.org/10.1145/3302504.3311794) (cit. on p. 11).
- [42] P. Grohs. ‘Optimally sparse data representations’. In: *Harmonic and applied analysis*. Applied and Numerical Harmonic Analysis. Birkhäuser/Springer, Cham, 2015, pp. 199–248. doi: [10.1007/978-3-319-18863-8_5](https://doi.org/10.1007/978-3-319-18863-8_5) (cit. on p. 10).

- [43] G. Henry, P. T. P. Tang and A. Heinecke. ‘Leveraging the bfloat16 artificial intelligence datatype for higher-precision computations’. In: *26th IEEE International Symposium on Computer Arithmetic* (Kyoto, Japan). IEEE, June 2019, pp. 69–76. doi: [10.1109/ARITH.2019.00019](https://doi.org/10.1109/ARITH.2019.00019) (cit. on p. 9).
- [44] N. J. Higham and T. Mary. ‘Mixed precision algorithms in numerical linear algebra’. In: *Acta Numerica* 31 (2022), pp. 347–414. doi: [10.1017/S0962492922000022](https://doi.org/10.1017/S0962492922000022) (cit. on p. 13).
- [45] J. van der Hoeven and R. Larrieu. ‘Fast Gröbner basis computation and polynomial reduction for generic bivariate ideals’. In: *Applicable Algebra in Engineering, Communication and Computing* 30.6 (2019), pp. 509–539. doi: [10.1007/s00200-019-00389-9](https://doi.org/10.1007/s00200-019-00389-9) (cit. on p. 11).
- [46] T. Hubrecht, C.-P. Jeannerod and P. Zimmermann. ‘Towards a correctly-rounded and fast power function in binary64 arithmetic’. In: *30th IEEE International Symposium on Computer Arithmetic*. July 2023. URL: <https://hal.science/hal-04159652v1> (cit. on p. 6).
- [47] C.-P. Jeannerod, N. Louvet, J.-M. Muller and A. Plet. *A library for symbolic floating-point arithmetic*. 2016. URL: <https://hal.science/hal-01232159> (cit. on p. 8).
- [48] M. Joldes, J.-M. Muller and V. Popescu. ‘Tight and rigorous error bounds for basic building blocks of double-word arithmetic’. In: *ACM Transactions on Mathematical Software* 44.2 (Oct. 2017), pp. 1–27. doi: [10.1145/3121432](https://doi.org/10.1145/3121432) (cit. on p. 9).
- [49] E. Kaltofen and G. Villard. ‘On the complexity of computing determinants’. In: *Comput. Complex.* 13.3 (2005), pp. 91–130. doi: [10.1007/s00037-004-0185-3](https://doi.org/10.1007/s00037-004-0185-3) (cit. on p. 11).
- [50] M. Kissel and K. Diepold. ‘Structured Matrices and Their Application in Neural Networks: A Survey’. In: *New Generation Computing* 41 (2023), pp. 697–722. doi: [10.1007/s00354-023-00226-1](https://doi.org/10.1007/s00354-023-00226-1) (cit. on p. 8).
- [51] O. Kupriianova and C. Lauter. ‘Metalibm: A Mathematical Functions Code Generator’. In: *4th International Congress on Mathematical Software* (Seoul, South Korea). Ed. by C. Y. Hoon Hong. Vol. 8592. Lecture Notes in Computer Science. Springer, Aug. 2014, pp. 713–717. doi: [10.1007/978-3-662-44199-2_106](https://doi.org/10.1007/978-3-662-44199-2_106) (cit. on p. 6).
- [52] V. Lefèvre, N. Louvet, J.-M. Muller, J. Picot and L. Rideau. ‘Accurate Calculation of Euclidean Norms Using Double-Word Arithmetic’. In: *ACM Trans. Math. Softw.* 49.1 (Mar. 2023). doi: [10.1145/3568672](https://doi.org/10.1145/3568672) (cit. on p. 7).
- [53] V. Lefèvre, J.-M. Muller and A. Tisserand. ‘Towards Correctly Rounded Transcendentals’. In: *13th IEEE Symposium on Computer Arithmetic*. Pacific Grove, CA: IEEE Computer Society Press, Los Alamitos, CA, 1997, pp. 132–137. doi: [10.1109/ARITH.1997.614888](https://doi.org/10.1109/ARITH.1997.614888) (cit. on p. 7).
- [54] Y. L. Luke. *The special functions and their approximations, Vol. I*. Mathematics in Science and Engineering, Vol. 53. New York: Academic Press, 1969 (cit. on p. 11).
- [55] J.-M. Muller. *Elementary Functions, Algorithms and Implementation*. 3rd. Birkhäuser Boston, MA, 2016. doi: [10.1007/978-1-4899-7983-4](https://doi.org/10.1007/978-1-4899-7983-4) (cit. on pp. 6, 13).
- [56] J.-M. Muller, N. Brunie, F. de Dinechin, C.-P. Jeannerod, M. Joldes, V. Lefèvre, G. Melquiond, N. Revol and S. Torres. *Handbook of Floating-Point Arithmetic*. 2nd ed. Birkhäuser Basel, 2018. doi: [10.1007/978-3-319-76526-6](https://doi.org/10.1007/978-3-319-76526-6) (cit. on p. 13).
- [57] J.-M. Muller and L. Rideau. ‘Formalization of double-word arithmetic, and comments on “Tight and rigorous error bounds for basic building blocks of double-word arithmetic”’. In: *ACM Transactions on Mathematical Software* 48.2 (2022), pp. 1–24. doi: [10.1145/3484514](https://doi.org/10.1145/3484514) (cit. on p. 9).
- [58] S. Naldi and V. Neiger. ‘A divide-and-conquer algorithm for computing Gröbner bases of syzygies in finite dimension’. In: *45th International Symposium on Symbolic and Algebraic Computation*. 2020, pp. 380–387. doi: [10.1145/3373207.3404059](https://doi.org/10.1145/3373207.3404059) (cit. on p. 11).
- [59] V. Neiger, B. Salvy, É. Schost and G. Villard. ‘Faster modular composition’. In: *Journal of the ACM* 71.2 (2024), pp. 1–79. doi: [10.1145/3638349](https://doi.org/10.1145/3638349). eprint: [2110.08354](https://arxiv.org/abs/2110.08354) (cit. on p. 11).
- [60] *OCF 8-bit Floating Point Specification (OFP8)*. Tech. rep. Open Compute Project, 2023. URL: <https://www.opencompute.org/documents/ocf-8-bit-floating-point-specification-ofp8-revision-1-0-2023-06-20-pdf> (cit. on p. 13).

- [61] S. Olver and A. Townsend. ‘A fast and well-conditioned spectral method’. In: *SIAM Review* 55.3 (2013), pp. 462–489. doi: [10.1137/120865458](https://doi.org/10.1137/120865458). eprint: [1202.1347](https://arxiv.org/abs/1202.1347) (cit. on p. 12).
- [62] A. Rudra. ‘Arithmetic Circuits, Structured Matrices and (not so) Deep Learning’. In: *Theory of Computing Systems* 67 (2023), pp. 592–626. doi: [10.1007/s00224-022-10112-w](https://doi.org/10.1007/s00224-022-10112-w) (cit. on p. 8).
- [63] C. D. Sa, A. Cu, R. Puttagunta, C. Ré and A. Rudra. ‘A Two-pronged Progress in Structured Dense Matrix Vector Multiplication’. In: *ACM-SIAM Symposium on Discrete Algorithms*. 2018, pp. 1060–1079. doi: [10.1137/1.9781611975031.69](https://doi.org/10.1137/1.9781611975031.69) (cit. on p. 11).
- [64] B. Salvy. ‘Linear Differential Equations as a Data-Structure’. In: *Foundations of Computational Mathematics* 19.5 (2019), pp. 1071–1112. doi: [10.1007/s10208-018-09411-x](https://doi.org/10.1007/s10208-018-09411-x). URL: <https://rdcu.be/bf1eN> (cit. on pp. 10, 11).
- [65] A. Sibidanov, P. Zimmermann and S. Glondou. ‘The CORE-MATH Project’. In: *29th IEEE International Symposium on Computer Arithmetic*. virtual, France, Sept. 2022 (cit. on p. 6).
- [66] R. P. Stanley. ‘Differentiably finite power series’. In: *European Journal of Combinatorics* 1.2 (1980), pp. 175–188. doi: [10.1016/S0195-6698\(80\)80051-5](https://doi.org/10.1016/S0195-6698(80)80051-5) (cit. on p. 11).
- [67] D. Stehlé. ‘On the Randomness of Bits Generated by Sufficiently Smooth Functions’. In: *7th Algorithmic Number Theory Symposium*. Ed. by F. Hess, S. Pauli and M. E. Pohst. Vol. 4076. Lecture Notes in Computer Science. 2006, pp. 257–274. doi: [10.1007/11792086_19](https://doi.org/10.1007/11792086_19) (cit. on p. 7).
- [68] G. Villard. ‘On computing the resultant of generic bivariate polynomials’. In: *ACM International Symposium on Symbolic and Algebraic Computation*. New York, NY, USA: ACM Press, 2018, pp. 391–398. doi: [10.1145/3208976.3209020](https://doi.org/10.1145/3208976.3209020) (cit. on p. 11).
- [69] L. Zheng, E. Riccietti and R. Gribonval. ‘Efficient identification of butterfly sparse matrix factorizations’. In: *SIAM Journal on Mathematics of Data Science* 5.1 (2023), pp. 22–49. doi: [10.1137/22M1488727](https://doi.org/10.1137/22M1488727) (cit. on p. 8).
- [70] A. Ziv. ‘Fast evaluation of elementary mathematical functions with correctly rounded last bit’. In: *ACM Transactions on Mathematical Software* 17.3 (Sept. 1991), pp. 410–423. doi: [10.1145/114697.116813](https://doi.org/10.1145/114697.116813) (cit. on p. 7).