# MITACS Project CV and Business Plan

November 1, 2006

**Title of Project:**

Mathematics of Computer Algebra and Analysis (MOCAA)

**Project website:**

`www.cecm.sfu.ca/~pborwein/MITACS/index.htm`

# 1. Project Investigators

## Project leaders:

George Labahn, School of Computer Science, University of Waterloo.
Michael Monagan, Department of Mathematics, Simon Fraser University.

## Project investigators:

Contact information for network investigators may be found on the project website.

**Simon Fraser University:**
Peter Borwein, Mathematics
Petr Lisonek, Mathematics
Marnie Mishna, Mathematics (new)
Michael Monagan, Mathematics

**University of Waterloo:**
Keith Geddes, Computing Science
Mark Giesbrecht, Computing Science
George Labahn, Computing Science
Arne Storjohann, Computing Science

**University of Western Ontario:**
Robert Corless, Applied Mathematics
David Jeffrey, Applied Mathematics
Marc Moreno-Maza, Computer Science
Greg Reid, Applied Mathematics
Eric Schost, Computer Science (new)
Stephen Watt, Computer Science

**University of Calgary:**
Wayne Eberly, Computer Science

**McMaster University**
Jacques Carette, Computer and Software

**Dalhousie University**
Jon Borwein, Computer Science

**University of Lethbridge:**
Howard Cheng, Mathematics and Computer Science (new)

## Project colaborators:

Francois Bergeron, University of Quebec at Montreal
Robert Israel, Mathematics, University of British Columbia
Ilias Kotsirias, Wilfred Laurier University
Eugene Zima, Wilfred Laurier University.

## 2. Non-Academic Partners

a) Industrial partners.

**Organization Name:** Maplesoft Inc.
**Name of Primary Contacts:** Laurent Bernardin, Jürgen Gerhard.
**Mailing Address:** Maplesoft, 615 Kumpf Drive, Waterloo, Ontario, N2V 1K8
**Phone Number:** (519) 747-2373
**Fax Number:** (519) 747-5284
**E-mail Addresss:** lbernardin@maplesoft.com, jgerhard@maplesoft.com
**Web Page:** `http://www.maplesoft.com`

# 3. Science

## 3 (a) Summary of the state of the knowledge of the field

Symbolic algebra systems such as Maple and Mathematica have achieved a remarkable degree of sophistication over the last twenty years. Difficult problems, such as exact integration of elementary functions and polynomial factorization, have been attacked with considerable success. These systems have incorporated many of the most important algorithms of the twentieth century, including the Fast Fourier Transform, lattice reduction with the LLL algorithm, Groebner bases, and the Risch decision procedure for elementary function integration. As a result, they can now effectively deal with large parts of the standard mathematics curriculum (and can outperform most of our undergraduates).

Mathematical software systems have become a central research tool in many subareas of mathematics both from an exploratory and from a formal point of view. Initially, the objective of symbolic algebra systems was to capture as much symbolic and exact mathematics as possible. As a basic algorithmic understanding was achieved, a second objective arose, namely, that of efficiency: for example, how to avoid "intermediate expression swell", how to ensure small (or even minimal) sized expressions, and so on. The numerical computing environment for these systems have arbitrary precision arithmetic and recent trends have been to extend algebraic and parameterized problems for approximate mathematical objects, and then dealing in such an environment with the more usual questions of analysis. Basically one would like to be able to incorporate the usual methods of numerical analysis into an environment that includes parameters and symbolic quantities.

While computer algebra systems have had significant impact in education and research, the same cannot yet be said for their impact in industrial settings. Indeed one of the biggest challenges for symbolic computation is to increase its ability to play a central role in solving industrial problems. At the same time, the desire to create tools for use in industrial settings that allow for manipulation and computation with symbolic objects (for example, parameters) is clearly present today. In addition, symbolic computation and exact arithmetic allows for the ability to model processes in an exact rather than approximate manner. One example of this is a project between Maplesoft and Toyota Japan which seeks to build a modelling environment which makes use of symbolic rather than numeric computation. One can think of this as a symbolic version of simulink where the boxes represent exact rather than approximate models and where no information is lost during the modelling.

Currently, the biggest stumbling block to the use of symbolic computation in an industrial setting is one of efficiency of exact algorithms. In addition, one often finds that symbolic computations do their arithmetic operations in approximate domains, for example, computing with multivariate polynomials having approximate (fixed precision) coefficients. The main focus of our project is to strengthen and extend symbolic computation to industrial settings. We plan to do this by first increasing the size of problems that can be efficiently handled by core operations in computer algebra systems. These core operations focus on both symbolic linear algebra and symbolic and numeric polynomial algebra. High performance computer algebra focuses on techniques such as parallel computation, the use compiled rather than interpreted libraries for linear algebra and the construction of optimal algorithms using bit-complexity models (which have signicant impact for very large problems). In the case of linear algebra, the optimal algorithms are built using theoretic advances (for example, give procedures in terms of matrix multiplication or polynomial multiplication where fast methods exist) but which ultimately also provide practical advances (for example, making use of highly tuned numerical libraries such as BLAS and ATLAS).

Computer algebra systems rely on (literally) hundreds of individual developments of two varieties: math-

ematical development of new and faster algorithms and software design improvements. Some of the areas to which the principals on this grant have contributed include:

- modular algorithms for polynomial GCD computation
- polynomial factorization and Groebner basis computation
- simplification of elementary and various special functions (with branch cuts)
- exact solution of ODEs
- matrix polynomial algebra
- algorithms for symbolic integration
- differential Groebner bases
- an algebraic solver for differential (ordinary and partial) equations
- high precision numerical quadrature
- structured total least squares methods for approximate polynomials
- integer relation algorithms and inverse symbolic computation (PSLQ)
- finding all zeros of arbitrary analytic functions in arbitrary regions
- solving systems of sparse linear diophantine equations
- methods for accurate arbitrary-precision linear algebra
- algorithms for linear algebra over finite fields
- normal forms of linear systems of differential and difference operators

## 3 (b) Summary of Main Achievements during last Project CV

In this section we summarize the project's main achievements since November 2004 and describe the project's progress towards objectives set at that time.

In 2004 we proposed research in one new area, symbolic linear algebra, along with four additional areas that followed from previous projects. Exciting progress can be reported in all the areas with some highlights mentioned below.

During the past two years there has been a strong focus for many of the researchers in the consortium on research in symbolic linear algebra. Symbolic linear algebra is viewed as a core component of many of the tools needed in computer algebra systems. The overall project had several important goals: (i) the continued development of the LinBox library routines for symbolic linear algebra (ii) linking the LinBox library with Maple and its existing linear algebra functionalities (iii) to develop and implement new fast (hopefully optimal) algorithms for (sparse) integer matrices, and matrices of polynomials. There was a strong linkage between the goals of (i) and (iii) in the sense that the new, fast algorithms could be implemented inside the LinBox library and hence give experimental backing to the hoped for complexity gains of the new procedures. We have made outstanding progress on all three goals.

(i) We have created a Maple interface which is now included with the LinBox library (www.linbox.org). This allows Maple users to link up with the compiled algoithms in the library, ensuring both wider usage of LinBox (since the library will be easy to use in the Maple environment and access to high performance linear algebra routines for power users of Maple.

(ii) The paper "Solving Sparse Rational Linear Systems" by Eberly, Giesbrecht, Giorgi, Storjohann, and Villard, proposes a new algorithm to find the rational solution of a sparse linear system over the integers. The method is conjectured to give it gives sub-cubic time complexity for solving such systems. A LinBox implementation comparing the new algorithm and others on very large linear systems is given.

A copy of this paper is included with the submission.

(iii) The paper "Normal Forms for General Polynomial Matrices" by Beckermann, Labahn and Villard, gives a new effective method which makes use of minimal approximant bases for computation of many different matrix polynomial normal forms and algebraic operations. The algorithms are the best so far for such a general computation is far from optimal. The results will form the starting point for our search for optimal algorithms for matrix polynomial arithmetic.

A copy of this paper is included with the submission.

(iv) In "Computing the rank and a small nullspace basis of a polynomial matrix", Storjohann and Villard reduce the problem of computing the rank and null-space of a univariate polynomial matrix to polynomial matrix multiplication. For an input $n$ x $n$ matrix of degree, $d$ over a field $K$ the methods give a rank and nullspace algorithm using about the same number of operations as for multiplying two matrices of dimension, $n$ and degree, $d$. The results are achieved through a combination of matrix Hensel high-order lifting and matrix minimal fraction reconstruction, and through the computation of minimal or small degree vectors in the nullspace seen as a $K[x]$-module.

(v) In "A Modular Algorithm for Computing the Characteristic Polynomial of an Integer Matrix in Maple", Lo, Monagan and Wittkopf implement a known modular algorithm using 25 bit floating point primes and delayed division for increased efficiency. Comparsion with the existing Berkowitz algorithm in Maple 10 on an 1916 by 1916 integer matrix from combinatorics, given to us by J. Quaintance, shows that there is no comparison : 2.3 million seconds vs 16.5 thousand seconds for our modular approach. Then the authors reduced the time to 47 seconds! The matrices have a non-obvious block structure that can be identified by identifying the strongly connected components of the directed graph associated with the matrix. This work is in Maple 11.

(vi) In "Generic Linear Algebra and Quotient Rings" Lo, Monagan and Pearce (2006) develop a new way of implementing generic algorithms in Maple. The paper demonstrates the idea by providing Maple with linear algebra over general quotient rings. A package of generic algorithms for linear algebra has been integrated into Maple 11.

(vii) In "Pivot-Free Block Matrix Inversion" Watt (2006) presents an algorithm to compute the inverse of a matrix in recursive block form using only block operations. This method is applicable to matrices over a wide class of coefficient rings, which may be non-commutative or of finite characteristic.

The other areas of research in the last project came under the headings : differential equations, simplification and special functions, number theory and finally symbolic and numeric integration and summation. As was the case with our linear algebra work, there are many results to report!

(i) In “Symbolic-numeric Sparse Interpolation of Multivariate Polynomials”, M. Giesbrecht, G. Labahn, and W. Lee study the problem of interpolating a black box polynomial with approximate (floating point) coefficients from it’s values. The authors show that sparse interpolation can be accomplished both efficiently and robustly if one evaluates at *random* roots of unity.

(ii) The `RegularChains` software library for solving systems of algebraic equations symbolically and for studying the properties of their solutions. This is an implementation of the triangular decomposition techniques of [46]. It was shipped with Maple 10 and updated for Maple 11. The library also provides Maple with the following features, that are not offered in any other computer algebra system:

- linear algebra over direct products of fields,
- equiprojectable decompositions of polynomial systems, and
- automatic case discussion with automatic case recombination.

A number of new algorithms have been developed to accompany the design of this library. One highlight is "Lifting techniques for triangular decompositions" (2005) [29], which received the *Distinguished Student Author award* at ISSAC.

A copy of this paper is included with the submission.

(iii) In "Symmetry classification using invariant differential operators" by Lisle and Reid (2006) a generalization of exact differential elimination methods to the noncommutative case is applied. These elimination methods are a generalization of Gröbner Bases (or more precisely triangular decomposition) to the differential case. By application of a finite number differentiations and eliminations they canonically simplify such sysetms, determine the missing constraints for PDE systems, and complete them to a form amenable to numerical or exact solution techniques.

(iv) Many symbolic computations in polynomial rings such methods as Gröbner Bases etc require an input ordering of the variables. However they can not be applied to systems with inexact input directly since these processes are discontinuous. For zero dimensional polynomial systems the paper [47] introduced a new concept approximate equiprojectable decomposition, based on earlier work by Moreno Maza, Schost et al on the exact equiprojectable decomposition. Unlike other numerical processes which manipulate with the equations, for example, numerical Gröbner basis, our methods first compute geometric objects, specifically isolated points, by a stable process. The equations are reconstructed by a modified Lagrange interpolation method.

(v) In “Rational Simplification Modulo a Polynomial Ideal” Monagan and Pearce (2006) developed algorithms (based on Groebner bases for modules) for the simplification of formulae modulo a polynomial ideal. The software has been integrated into Maple’s simplifier for Maple 11.

(vi) In “Understanding expression simplification” J. Carette has developed a theoretical model for simplification based Kolgormorov complexity of a formula. This gives a rigorous justification to some ideas that have used in formula simplifiers.

(vii) The Number Theory aspect of our previous project proposal centered around an attack of the Merit factor problem, the problem of Golay, and integer factorization using large scale grid computation. “The Merit Factor Problem” , P. Borwein, R. Ferguson and J. Knauer is to be submitted to the Proceedings of Number Theory and Polynomials Conference, to be held in Bristol, 3-7 April 2006. The work

reports on searches (using a genetic type algorithm on G5 and Beowolf clusters) to find sequences of high merit factor at lengths up to 250. The authors report on the discovery of approximately 2,500 new sequences having merit factor $> 8$ and 23 with merit factor $> 9$.

(viii) "Integer Factorization and Computing Discrete Logarithms in Maple" by Bradford, Monagan and Percival (2006) describes some work by Colin Percival, a PDF, who holds the record for computing the binary representation of $\pi$. Percival has implemented a self initializing quadratic sieve (SIQS) for factoring integers in C (the sieve and solving the linear system) and in Maple. This is the algorithm of choice for integers of length 30 to 120 digits, beyond which the number field sieve is faster. The assumption is that most Maple users will have a dual core processor available and will want to factor integers in less than 1 CPU-day, and therefore good performance on integers which can be factored within this time bound is more important than optimal performance on integers which would take months or years to factor.

(ix) In "On the derivatives of the Bessel and Struve functions with respect to the order" by Brychkov and Geddes, the problem of computing derivatives of these functions with respect to order was given and integrated into Maple 10. In a previous work, paper [15], formulae for the differentiation of hypergeometric functions with respect to parameters was given. These derivative operations (with respect to parameters) give a key step for evaluating families of definite integrals via the "method of differentiating under the integral sign".

(x) In "Hybrid Symbolic-Numeric Integration in Multiple Dimensions via Tensor-Product Series" by Carvajal, Chapman and Geddes, a new hybrid symbolic-numeric method for the fast and accurate evaluation of multiple integrals was given. The method is effective both in high dimensions and with high accuracy. This work will form the basis for an implementation in Maple of state-of-the-art algorithms for multidimensional integration. Initial results have shown that for some frequently encountered families of integrands, the new method breaks the curse of dimensionality that afflicts numerical integration in higher dimensions.

## 3 (c) Objectives

The emphasis of the various projects is the development and implementation of software for finding exact as opposed to numerical solutions to mathematical problems. This is an overlap of computer science with pure and applied mathematics, requiring deep analysis and sophisticated algorithmic development. Many researchers develop software tools for their specific needs. The investment required to adapt these tools for wider applicability is huge. However, the importance and benefits of providing such good working tools to the general mathematical community is difficult to overestimate. This is an expanding area of expertise, offering considerable opportunity in both academic and industrial environments.

Objectives of the projects in this consortium may be summarized as follows:

- new algorithm development for the various problems
- software development in our various projects
- applying our software in other research projects
- interaction with industrial suppliers of software
- delivery of products to general users

## 3 (d) Methodology

Canada's position as a major participant in the development of mathematical software stems from the early eighties and the very successful creation of Maple at the University of Waterloo. Systems such as Maple, and its main competitor Mathematica, and the related numerically-oriented system Matlab, are now primary research and development tools in mathematics. These are now substantial sized businesses, with Maple employing well over one hundred personnel. Even at its current size, Maple interacts closely with the research groups at the the three principal sites of this proposal. Indeed these labs are Maple's primary source of both leading-edge research and prototype development. This unique synergistic relationship continues to foster a thriving computer algebra community in Canada. This unique relationship was recognized in 2004 with an NSERC Synergy award between Maplesoft and the three computer algebra labs at Waterloo, Western and Simon Fraser. The problems under investigation are determined in conjunction with Maplesoft Inc. on the basis of our common interests. As a corporation, Maplesoft is comfortable with and understands that it will get the best product from people doing what most interests them. It is a very satisfactory partnership.

One of the important directions in computer algebra is to provide algorithms that allow symbolic computation to play a significant role in solving industrial problems. In order to be effective, computer algebra systems need to address problems of efficiency. Current PC technology implies that in two years most standard PCs will have parallel processing capabilities, supporting multi-core and multiprocessor configurations in consumer models. In anticipation of this, the next release of Maple will provide tools that will allow for parallel computation. The next step, and one that we will be following in this proposal, is to use such parallel support tools for parallel algorithms. In particular we will investigate new parallel algorithms for both Groebner bases and linear algebra computations.

High performance for symbolic computation also enters into two other primary directions taken in this grant. The first is the use of specialized, compiled (rather than interpreted) libraries for exact linear algebra computation. The second is to develop a series of new algorithms that have optimal bit complexity and which have features that make them practical for fast implementation.

Our approach also takes the point of view that overall performance gains are obtained by making core operations fast, particularly those dealing with polynomial algebra and linear algebra. This includes alternative representations of such objects and a treatment of numerical (floating point) coefficients. Finally, while performance gains are important, we also plan to make progress on extending the applicability of symbolic computation by developing new algorithms for fundamental computations.

## 3 (e) Descriptions of subprojects

### 3.1 High-performance computer algebra

With the need for high performance in practical problems, research in symbolic computation has entered a new era where implementation techniques have become as important as theoretical developments. This phenomenon is heavily accentuated by the revival and the democratization of parallel architectures. The previous works in parallel symbolic computing are obsolete due to the emerging programing paradigms and architectures; they had also a limited impact and, the attention of the symbolic computation community was withdrawn from parallelism during the last decade.

The aim of our theme High-performance computer algebra is to deliver the mathematical algorithms, implementation techniques and software for symbolic computation to exploit these new computing resources, from SMP to multi-core laptops. Many asymptotically fast algorithms and processor efficient parallel algorithms in computer algebra reduce to linear algebra operations. The LinBox project, Section 3.1.1, therefore

plays a central role: it provides an implementation framework supporting our other projects on symbolic linear algebra:

- Section 3.1.1 presents a new component of LinBox with many applications: fast algorithms for integer matrices of very large dimension,
- Section 3.1.2 deals with the fundamental questions of structured matrices and Padé approximants,
- Section 3.1.3 investigates the parallelization of linear solvers over finite fields, a central operation for algorithms relying on modular techniques.

The project of Section 3.1.4 focuses on the implementation challenges in parallel symbolic computation that are not specific to linear algebra; its ultimate objectives are to support efficient parallel solvers for systems of algebraic and differential equations.

Our consortium members are all world leaders on the theme *High-performance computer Algebra* working actively on asymptotically fast algorithms (Section 3.1.2), modular algorithms (Sections 3.1.4,3.1.5,3.2.1), implementation techniques (Section 3.1.1), high-precision numerical computing (Sections 3.2.3,3.3.3), parallelism (Sections 3.1.5,3.3.3). Our members have initiated the renaissance of the conference Parallel Symbolic Computation, whose last edition was in 1997 in Hawaii. The next one will be held at the University of Western Ontario, in July 2007, organized by M. Moreno Maza and S.M. Watt.

### 3.1.1 The LinBox generic library for symbolic linear algebra

The goal of the LinBox project is to address the many challenges in symbolic, exact and arbitrary precision linear algebra which arise in modern symbolic computation algorithms. Our approach is a high-performance component library, targeted at solving large, sparse and structured systems over symbolic and exact domains. This group of more than thirteen researchers (and many more postdocs and graduate students), including:

- B. Caviness, E. Kaltofen, A. Lobo, C. Meyer, B.D. Saunders, Q. Xiang (USA)
- J.-G. Dumas, T. Gautier, J.-L. Roch, G. Villard (France)
- W. Eberly, M. Giesbrecht, A. Storjohann (Canada).

is implementing this library in C++, with a specific eye towards generic data types, and flexible sparse and black-box matrix representations. Some of the current work is described at [1]. This work has been supported by the NSF under their Information Technology Research (ITR) program, and by the CNRS in France. Under the previous MITACS proposal we initiated an interface with the `Maple` Computer Algebra System. A first version of this interface is now distributed with the LinBox library. A goal of the current project is to extend this interface, in a wide set of directions. One example is to investigate the ability for the Maple interface to take advantage of the linbox facilities for working with generic domains. A second direction is to make use of linbox in order to implement and test the optimal algorithms proposed for our work on fast linear algebra as described in the next subsection.

### 3.1.2 Fast linear algebra with integer matrices

The goal of this subproject is to design and implement new algorithms specifically for the exact solution of linear algebra problems on integer matrices. The focus is on input matrices with very large dimension and possibly with some additional features such as sparsity or small entry size. Recently invented techniques

such as the shifted number system (for obtaining certified exact results using approximate arithmetic) and high-order lifting and integrality certification (for incorporating matrix multiplication) have led to reductions to integer matrix multiplication (and hence theoretically nearly optimal) algorithms for integer linear system solving and determinant computation.

Recently we have found an algorithm for solving sparse integer linear systems which apparently requires a sub-cubic number of machine operations [32]. This method uses "standard" matrix and integer arithmetic, and is relatively space efficient. An initial implementation in LinBox demonstrates that this method can be faster than previous methods in practice, at least on very sparse matrices. Both the theory and implementation warrants more study. Moreover, solving linear systems is the lynch-pin for most other linear algebra operations: our new methods will yield faster algorithms for many other linear algebra operations, such as determinants and Hermite and Smith normal forms.

### 3.1.3 Order bases and minimal approximant bases

Problems such as those given by Hermite-Padé approximation provide techniques where one can solve a wide variety of inverse problems. For example,one can be given a power series and asked to find a linear ODE (with specified bounds on the degrees of the coefficients) which is solved by the power series. Such computations arise naturally in many problems of reconstruction in algebraic computation. The sigma basis algorithm of Beckermann and Labahn [9] provides a fast way of computing all solutions of such problems, describing these solutions in terms of a module basis. It also gives deterministic algorithms for a wide variety of problems [6].

However, the sigma-basis algorithm often computes far more than is needed for many problems. In particular, when there are degree constraints it often computes basis elements which do not meet degree constraints and hence never in fact enter into an combination giving a solution (i.e. the coefficient of the basis element is always zero). In addition, the deterministic algorithms in [6] are often far from optimal. This is particularly the case when the degree bounds are highly skewed or the gaps between degrees of the basis elements and the degree constraints is quite large.

We will be investigating new techniques for obtaining optimal algorithms for the above problems and in particular for the general problem of minimal basis computation. A first step in this direction was given in [36]. In the general case we expect to build on techniques described in [12] and follow the recent approach of Storjohann.

### 3.1.4 Parallel solution of linear systems over cyclotomic fields

Let $A$ be an $n$ by $n$ matrix of integers and $b$ a vector of $n$ integers. The fastest methods to compute the rational solutions $x$ of the linear system $Ax = b$ are based on $p$-adic lifting and rational number reconstruction. The advantage $p$-adic lifting is that it reduces the $n^3$ factor in the complexity to solving $Ax = b$ modulo a single machine prime $p$ which can be reduced to matrix-matrix and matrix-vector multiplication modulo $p$.

In this project we consider linear systems involving a root of unity $\beta$. Roots of unity occur frequently in many applications. Our application comes from the finite group representation problem where one is constructing matrices for representing group elements. The reason to consider roots of unity here is that the minimal polynomial $f(x)$ for a root of unity splits into linear factors modulo a prime with relatively high probability - with probability approximately $1/d$ where $d = \deg(f)$. This property makes accelerating modular algorithms possible, and it also makes parallelizing the algorithms more feasible.

In current work Monagan, Chen and Dabbighian are looking at modular methods. The first method solves $Ax = b$ modulo a sequence of primes which split the minimal polynomial. This method can be

easily parallelized. The roots of unity appear to complicate the $p$-adic methods and may make them less competitive with the multiple primes approach. Also, the $p$-adic methods do not parallelize easily. The proposal is (1) to continue to search for efficient $p$-adic methods, (2) to attempt a parallel implementation of the multiple primes approach and (3) to look at parallelizing the $p$-adic methods.

#### 3.1.5 Parallel symbolic computing

The current development of efficient symbolic polynomial system solvers, see Section , based on modular techniques, is another stimulation for the renaissance of parallel symbolic computation. Indeed, as demonstrated in [43, 44], modular techniques help increasing opportunities for parallel execution in a significant manner. Several challenges remain, however, such as load balancing of highly irregular tasks and heavy data communication. In addition, parallel symbolic polynomial system solvers combine several levels of parallelism:

- a low level or, arithmetic level where operations, such as polynomial GCDs, can reduce to symbolic linear algebra,
- a high-level, of task management, that we call component level where operations are of geometric nature and do not reduce directly to symbolic linear algebra.

To deal with these challenges, our project will focus on three topics:

- support for parallelism in high-level programing language environment, extending the work on thread-level speculation for parallel computation [50],
- algorithms and libraries for parallel polynomial arithmetic, which leads naturally to the projects of Section ,
- parallel symbolic solvers implementing arithmetic level and component level, targeting multi-threaded parallelism on SMP and multi-cores.

We anticipate that the solutions suggested in [44] would lead to efficient parallel solvers, implemented in the `Aldor` programming language and available from `Maple`.

### 3.2 Symbolic Solvers

Despite the simple statement and long history of most problems related to polynomial algebra, many questions in this area remain active and important research topics. In fact, the need to achieve high performance in speed and accuracy bring new challenges for symbolic solvers of polynomial systems. As mentioned in Section , this need is stimulated by the increasing availability of computer parallel architectures, from SMP to multi-core laptops.

The expertise of our consortium covers most active research topics in polynomial algebra. We have chosen to describe below five of our research projects in this area, representing the different topics:

- Modular techniques for solving polynomial systems, in section 3.2.1, provides a major research direction. In addition, they give rich opportunities for parallelism [44],
- Polynomial GCD computations modulo ideals, in section 3.2.2, are fundamental routines in polynomial system solving,

- Alternative bases for representing polynomials, in section 3.2.3, gives powerful techniques for dealing with expression swell in symbolic polynomial computations,
- Differential algebra, in section 3.2.4, is becoming one of the most challenging and attractive subjects in computer algebra.
- Symbolic integration and special functions, in section 3.2.5, is one of the major areas of success for the computer algebra community. It is also complement the other sections and provides stimulation for the other areas of symbolic solving.

Other priority research directions are listed below with recent publication of our members: fast algorithms for polynomial arithmetic [30, 11, 13] and implementation techniques for polynomial arithmetic [33, 41, 40]. These form a bridge toward our theme on high-performance computer algebra.

### 3.2.1 Modular techniques for solving polynomial systems

Solving systems of polynomial equations, linear or non-linear, algebraic or differential equations, remains a fundamental problem in mathematical sciences, which is hard for both numerical and symbolic approaches. For systems of linear equations, symbolic methods can compete today with numerical ones in terms of running times; moreover there are input systems for which numerical methods fail to provide accurate solutions while symbolic methods always do. For systems of non-linear equations, when both symbolic and numerical methods can be applied, the latter ones have the advantage of speed for most problems whereas the former ones have that of exactness.

A first goal of this proposed research is to develop and implement symbolic algorithms for solving non-linear systems that run nearly as fast as numerical methods when the comparison makes sense, that is, for systems that have finitely many solutions, with exact input coefficients and with symbolic output of moderate size. Under these hypothesis, we anticipate that the successful methods of the LinBox project, Section , could be extended to the non-linear case. In fact, a first step in this direction has been made in [29] using modular arithmetic techniques and was delivered to `Maple` [39].

A second objective is to design specific methods for polynomial systems with parameters, since there is an increasing need for such tools in robotics, geometric modeling, stability analysis of dynamical systems and other areas. Modular techniques, and thus efficient implementations, exist only for particular types of parametric systems, not covering most practical cases. However, triangular decompositions [46, 54, 31, 28, 44] appear as a very promising direction for the general case. This should lead to significant enhancement of the symbolic solvers in the Maple computer algebra system, in particular for computations involving automatic case discussion.

### 3.2.2 Polynomial GCD computation modulo ideals

The most frequent instances of such computations arise for polynomials with coefficients in algebraic function fields, e.g., $L = Q(\sqrt{s}, \sqrt{1-st})$. This includes algebraic numbers such as $\sqrt{2}$ and $i = \sqrt{-1}$ as a special case.

The goal of this project is to develop efficient modular algorithms for computing the greatest common divisor (GCD) of two multivariate polynomials over $L$. In a computer algebra system where one is computing with formulae involving roots, most of the time is spent computing GCDs.

In 2004, Monagan and van Hoeij [38] developed a *dense modular* algorithm for one field extension and Boulier, Moreno Maza, Oancea [14] developed a *Hensel lifting* algorithm for computing modulo a

triangular set. In current work Monagan and Javadi are considering a sparse algorithm. To efficiently treat the non-monic case, the algorithm uses "a variable at a time" rational function interpolation, an extension of the sparse polynomial interpolation algorithm of Zippel (1979, 1990). The first objective of this project is to develop this new approach, focusing on the sparse rational function interpolation, and investigating its parallelization.

Our project aims also at developing efficient methods for polynomial GCDs over more general coefficient rings, potentially with zero-divisors. A very practical example, which arises when solving systems of polynomial (algebraic or differential) equations, is that of a total ring of fractions of a residue class ring $R/I$ where $R$ is a polynomial ring and $I$ a radical ideal of $R$. Such a ring is, in fact, a direct product of fields. Theoretically, the celebrated D5 Principle and triangular decomposition techniques reduces this general case to that of algebraic function fields.

In 2005, Dahan, Moreno Maza, Schost and Xie [30] have extended the asymptotically fast algorithms for polynomial GCDs over fields to the case of direct product of fields. Moreover, for direct product of finite fields, their approach has a bit complexity which is nearly optimal. The second objective of this project is to put this theoretical result into practice. Finally, combining the sparse modular methods and fast algorithms of this project, we anticipate to obtain nearly optimal and practical algorithms for the general case of direct product of algebraic function fields.

### 3.2.3 Polynomial algorithms in alternative bases

Algorithms for exact polynomial arithmetic such as the computation of the greatest common divisor of two polynomials have traditionally been given under the assumption that the input polynomials are represented in the standard power basis. However in many cases polynomials are more naturally represented in alternative bases, for example, polynomials given in terms of Newton bases or in terms of orthogonal polynomials. This is the case, for example, in problems in linear control theory [8, Sec 5.3-5.4], interpolation problems and rational interpolation problems. In these cases the historical methods, until the pioneering work of Barnett and others in the 1970's, and Carstensen in the 1980's, required the input to be converted to the standard power bases, followed by the exact computation and then finishing with the result converted back to alternative form. Similarly, in the Computer-Aided Geometric Design community, much work has been done on implementation of algorithms for polynomials expressed in terms of Bernstein-Bézout polynomials, again avoiding unstable or expensive conversion to the monomial basis.

We plan to investigate a number of polynomial arithmetic computations where the input is represented in alternative form but where the algorithms work directly with the alternative form. This work will follow up on the refinements to the gcd problem as presented in [22], and the work on the Lagrange basis begun with the new linearization for matrix polynomials expressed in the Lagrange basis given in [4, 27, 26]. Exact computation in alternative bases ensures that one can avoid loss of sparsity or coefficient growth which often arises in the conversion process.

Our plan is to investigate such algorithms in both scalar and multivariate forms. In addition, we plan to investigate matrix normal form computations (such as Popov computations) when the matrix elements are represented in alternative bases. Finally, we will investigate algorithms in Ore domains where the polynomial coefficients are represented in alternative form. In this case the $\sigma, \delta$ rules take an unusual form, requiring different strategies for noncommutative computations as one-sided Ore gcd computations. Indeed in this case, the noncommutativity rule would need to be given in terms of the $n$-th basis element. In a standard power basis the rule would be $Dx^n = x^n D + nx^{n-1}$ while, for example, if coefficient polynomials in an Ore domain are represented in terms of Hermite polynomials $H_n(x)$, then the non-commutativity rule

would be expressed as

$$DH_n = H_n D + 2nH_{n-1}.$$

The resulting algorithms would have to be based on such rules, a considerable difference to their present representations.

In the case of approximate coefficients the work in the Lagrange basis, also known as "polynomial algebra by values", has entered an exciting phase, in particular with new developments in the characterization of pseudospectra of matrix polynomials. Intriguing problems remain, most notably the characterization of an 'optimal' set of nodes that gives the 'tightest' pseudospectra. Finally, we are interested in the problem of "optimal degree reduction" in the Bernstein basis, a problem of significant interest in the CAGD community.

### 3.2.4 Differential algebra

The level of development of algorithms and software for general differential polynomial systems is much less advanced than in the algebraic case. This is probably due to the fact that, the theoretical foundations in the differential case have required, and still require, more developments.

The breakthrough reported in [37] opens the door to the lifting of the most sophisticated algebraic techniques such as [46, 29] to the differential case. We anticipate to provide the first parallel symbolic solver for systems of partial differential polynomials.

In the ordinary case, developing efficient methods for the first order linear system of the form $X' = AX$ is motivated by many applications. In such forms it is possible, for example, to determine if there are rational solutions and exponential solutions. For linear-differential-algebraic systems it is possible to convert higher order systems to first systems via the computation of the differential Popov form of a matrix of differential operators. We expect to develop tools for this strategy that will lead to interesting invariants, understood directly in terms of higher order systems and which ultimately will be more efficient to compute in an one-step, rather than two-step process. In addition, the process of determining a differential Popov form, is similar to the computation of a differential Gröbner basis. We hope to either understand the tools for higher order linear systems in terms of known tools used for Gröbner basis computation or to extend these linear tools to the nonlinear case.

### 3.2.5 Symbolic integration and special functions

Computer algebra systems have had outstanding success in finding closed form solutions of indefinite integrals, particularly in the case where the integrand is an elementary function. However the same cannot really be said about definite integrals, particularly those with special functions in the integrand. In addition, there are still open problems associated with definite integration of elementary functions which do not have closed forms in terms of elementary functions. Computing closed form solutions for the case of Elliptic integrals is a prime example of the later problem.

We propose to work on two important integration problems. The first is to implement and complete the investigation of the Salvy method for the definite (improper) integral of the special product of two holonomic functions [52]. This method takes as input linear ODEs solved by each holonomic function and, using Mellin transforms and recurrence operators, finds the linear ODE solved by the integral. The solution of the computed linear ODE results in the closed form solution of the improper integral. Issues that need to be resolved include the difficult problem of parameters in the integrand, where multiple solutions are possible (depending on possible regions for the parameters). This is already a difficult problem, even in the

case where the input is a product of Meijer G functions, the simplest examples of this method and the case where answers in integration texts are readily available (c.f. the classic texts by Prudnikov et al).

A second problem involves different forms of closed form solutions for definite elliptic integrals. Currently, there are transformations which result in closed form solutions for these integrals in terms of the three Legendre functions: $F$, $E$ and $\Pi$, or alternatively, in terms of the Weierstrass $P$ function and its derivative. However, elliptic integrals provide many examples of notoriously complicated closed-form outputs in the symbolic case. We plan to investigate methods for finding "smaller" expressions for these closed form solutions, in particular by looking at forms given in terms of so-called Carlson functions [17]. Carlson functions have added significance since they are intended for inclusion in the NIST Digital Library. Unfortunately, this form relies strongly on the linear factorization of the polynomial which defines the algebraic relation in the integrand. In particular, the linear factors are often extremely difficult to work with efficiently (because of their size and form) and also because of the need to work with branch cuts of the square root function. We propose to investigate the use of implicit methods for this closed form.

## 3.3 Symbolic Numeric Computation

The growing demand of speed, accuracy, and reliability in scientific and engineering computing has been accelerating the merging of symbolic and numeric computations, two types of computation coexisting in mathematics yet separated in traditional research of mathematical computation. The aim of our research theme Symbolic-Numeric Computation (SNC) is to facilitate the symbolic-numeric interaction and integration. Problems involving polynomials with inexactly-known coefficients arise, for example, when physical measurements or numerical computations are used to specify a polynomial system. In this context, the usual exact algorithms of symbolic computations are not applicable and new approaches must be devised: for example, singular value decomposition [25] and probabilistic models for stability analysis [48]. Major contributions to the SNC area have been made by our consortium, for instance the Symbolic Numeric Approximate Polynomial (SNAP) library in Maple. In addition, the next SNC conference will be organized at the University of Western Ontario by M. Moreno Maza and S.M. Watt.

Our first project in this SNC theme, discussed in Section 3.3.1, aims at providing reliable and sparse arithmetic operations for polynomials with approximate coefficients. Our second project, presented in Section 3.3.2, investigates innovative directions for computing multidimensional integrals. Our third project, summarized in Section 3.3.3, is concerned with SNC high-performance computing, including parallelism, and could also be part of our first theme, High-performance computer algebra, see Section 3.1. In addition, it applies to the computations of multidimensional integrals.

### 3.3.1 Approximate polynomial interpolation

An important problem in representing a function or other mathematical object with respect to some basis is whether it can be captured with a small number of non-zero terms. In the case of a polynomial in the power basis this corresponds to sparsity, i.e., few non-zero terms. The problem is particularly difficult when the data is noisy or of limited precision, the typical and important case of modelling measured data, a problem which has difficult conditioning [5]. In [42] we give a numerically robust solution to the problem of sparse interpolation in the power basis. We evaluate at random primitive roots of unity to achieve efficient and probabilistically (and provably) numerically robust solutions. We will investigate a number of important directions. First, we will consider the restriction to real data, where interpolation is more inherently unstable. In a related effort, we will continue to investigate non-standard bases. Some initial success with Chebyshev

and trigonometric bases was achieved in [35]. Finally, when data points are expensive to obtain, determining the minimum number of required points is critical. Again, the use of randomness holds considerable promise here, and we will continue to pursue this direction.

### 3.3.2 Hybrid symbolic-numeric algorithms for multidimensional integration

An ISSAC'05 paper [19] by Carvajal, Chapman and Geddes presented a new hybrid symbolic-numeric method for the fast and accurate evaluation of multiple integrals, effective both in high dimensions and with high accuracy. This work evolved from Carvajal's Master's thesis [20] supervised by K.O. Geddes.

We propose to develop state-of-the-art implementations in Maple of algorithms for computing multidimensional integrals, based on the above work and ideas evolving from it.

The overall approach employs a recursive algorithm and the two-dimensional case can be considered as the base of the recursion. In two dimensions, we apply an adaptive algorithm for double integration of continuous functions over general regions based on approximating the integrand by tensor product series. This reduces the double integral to a series of one-dimensional integrals which can be evaluated by a numerical quadrature method or by symbolic techniques. The theory of tensor product series was developed in Chapman's Ph.D thesis [21], supervised by K.O. Geddes.

Higher dimensions are handled using a novel Deconstruction Approximation Reconstruction Technique (DART), which facilitates the dimensional reduction of families of integrands with special structure over hyperrectangular regions. Initial results have shown that for some frequently encountered families of integrands, DART breaks the curse of dimensionality that afflicts numerical integration in higher dimensions.

Due to the hybrid symbolic-numeric nature of our approach, we expect to be able to handle integrands with isolated singularities successfully by removing these singularities through symbolic analysis prior to integration. Such an approach has been applied successfully to one-dimensional integrals in past work.

Since our approach reduces each multiple integral to a collection of independent integrals with half the dimension of the original, our approach is very well-suited for parallel implementation on multiprocessor architectures. The inherent coarse-grain parallelism of our approach requires a minimum of interprocess communication and should therefore result in very low overhead.

### 3.3.3 Parallel and distributed high precision numerical integration

Jointly `Maple` and Dalhousie-DDRIVE (http://www.cs.dal.ca/ddrive/) are interested in high-performance computing, including distributed and parallel computing , symbolic-numeric computing, and intend to proceed towards algorithms for solving industrial-size real world problems.

Jon Borwein, in conjunction with D. Bailey (Lawrence Berkeley Labs) and R. Crandall (HPC head at Apple Computers) are engaged in a long-term project of providing efficient tools for arbitrary precision computation of one-dimensional integrals and very high precision (over 100 digits in 2-5 dimensions) multidimensional integrals such as those arising in many areas of mathematical physics.

Such precision is necessary, for example, to apply integer relation methods to important constants. Good examples can be found in [7]. The underlying algorithms are described in J. Borwein et al [10]. Attention will also be paid to correlate special function algorithms needed to make such integration efficiently parallelizable.

## 3.4 Emerging Technologies in Symbolic Computation

The quest to deliver efficient symbolic solvers for polynomial equations, algebraic or differential, linear or non-linear, has captured much of the efforts of the computer algebra community, directly or indirectly, over the last 40 years. It is indeed a fundamental problem in mathematical sciences and the number of applications for symbolic solvers has increased dramatically, with packages used in robotics, cryptology, computer aided design and modeling of dynamical systems.

That focus has eclipsed other important problems in symbolic mathematical computation. Our state of technology and, moreover, our basic knowledge for symbolic computation in other important areas has suffered and we see embarrassing lacunae in the areas of mathematics that have been "mechanized." For instance, automatic manipulation of polynomials with symbolic exponents, see Section 3.4.1, is a new research direction, which has applications in the study of biological dynamical systems [51]. Non-commutative polynomial algebra, see Section 3.4.2, receives an increasing attention, as it has demonstrated its potential in reducing expression swell in commutative algebra. This is important in the implementation of Cartan's equivalence method [18] by S. Neut [49]. Another area generating surprising results is holonomic programs, discussed in Section 3.4.3, with applications to reverse engineering of computer programs.

### 3.4.1 Polynomials with symbolic exponents

The goal of this project is to start to bridge the gap between "computer algebra" and "symbolic computation." Although many use the terms synonymously, we find it useful to make a distinction between the two. By "computer algebra," we mean the treatment of mathematical objects as values in some algebraic domain, for example performing ring operations on polynomials. In this view, $x^2-1$ and $(x-1)\times(x+1)$ are the same. By "symbolic computation," we mean working with terms over some set of symbols. In this view, the two expressions $x^2-1$ and $(x-1)\times(x+1)$ are different. It is difficult in computer algebra to treat problems with unknown values e.g. with a matrix with unknown size, a polynomial of unknown degree or with coefficients in a field of unknown characteristic). It is difficult in symbolic computation to use any but the most straightforward mathematical algorithms before falling back on general term re-writing.

We are interested in exploring what can be done to bridge the gap between these two views, making computer algebra more symbolic, providing effective algorithms for a broader class of mathematical expressions. We have begun to explore this topic in recent papers [56, 57]. There, we have formalized the notion of *symbolic polynomials* for multivariate polynomials where the degrees of the terms are themselves integer-valued polynomials in several variables. We have shown this domain has the natural algebraic structure of a group ring, and moreover, that for coefficient domains of interest, it is a unique factorization domain. It makes sense to ask to compute factorizations such as

$$x^{n^4-6n^3+11n^2-6(n+2m-3)} - z^{6m}$$

$$\begin{aligned} &= x^{-12m}\times\left(x^{2p}+z^m x^{p+2m}+z^{2m}x^{4m}\right)\times\left(x^p+z^m x^{2m}\right) \\ &\qquad \times\left(x^{2p}-z^m x^{p+2m}+z^{2m}x^{4m}\right)\times\left(x^p-z^m x^{2m}\right) \\ p &= 1/6n^4-n^3+11/6n^2-n+3 \end{aligned}$$

We have proposed two families of algorithms for GCD, factorization and related problems. The first uses the algebraic independence of $x_i$, $x_i^{n_1}$, $x_i^{{n_1}^2}$, $x_i^{n_1 n_2}$, *etc.* By introducing a number of new variables,

we reduce the problem to one on usual multivariate polynomials. The second approach is to use an evaluation/interpolation scheme on the exponent variables. This has combinatorial problems when the polynomials have a limited set of distinct coefficients. Careful use of evaluation, and solving equations in symmetric polynomials points to a direction that avoids exponential combinations. Early results show that sparse lifting techniques applied to exponents may provide interesting results.

Within the scope of this MITACS project, our goal is to see how the scope of these algorithms may be extended to related problems. Already it is clear that the structure of symbolic polynomials will allow computer algebra systems to simplify an important class of expressions that they (embarrassingly) cannot now handle.

### 3.4.2 Combinatorial tools from non-commutative polynomial algebra

The goal of this project is to use non-commutative algebras of linear operators (Ore Algebras) to produce tools useful, in part, for combinatorics, specifically, for generating function manipulation. In the first stage, we generalize a notion of closure of differential operator [55] to a wider class of linear operators. The closure is defined using extension and contraction of the D-module ideal generated by the operator, and in fact this corresponds to the polynomial torsion module. Direct applications of this study include algorithms for desingularizing (q-) recurrence relations and (q-) differential equations and reducing the order of recurrences satisfied by coefficients of power series solutions of linear differential equations. Our desingularization algorithms are faster than other known methods [3, 2], and the recurrences we generate are of the lowest possible order. We have a complete, documented implementation in Maple, and this will be distributed as part in the algolib library of the Algorithms project, at INRIA Rocquencourt, France.

The second stage of this project involves a new multivariate version of closure. The main challenge is finding the appropriate generalization and defining algorithms. A motivating application is improved algorithms for computing the scalar product of symmetric functions. This has implications in combinatorial enumeration [34], and would be an efficiency improvement on our previous work [23]. This project is in collaboration with Frederic Chyzak, Philippe Dumas, and Bruno Salvy at INRIA Rocquencourt in the Algorithms project, and Ha Le, a postdoctoral fellow at University of Waterloo and Maple.

### 3.4.3 Holonomic programs

Some promising preliminary work [16] defined a new member of the holonomic family: "holonomic programs". Sequences are said to be holonomic if they satisfy a linear recurrence equation with polynomial coefficients; functions are said to be holonomic if they satisfy a linear differential equation with polynomial coefficients. To these objects, one can naturally associate programs, say via gfun [53]. Holonomic functions and sequences have many applications, driven by effective algorithms [24, 45]. Thus holonomic programs are defined to be either discrete programs which satisfy a linear recurrence equation, or programs of a continuous variable which satisfy a linear differential equation (with polynomial coefficients). Using a variation of denotational semantics of programs which we call “symbolic semantics”, we are able to derive such equations. The main departure is in the denotation for while loops, which is modified to one closer to its operational semantics, and this leads directly to recurrences. An interesting side-effect of this analysis is that some constants, which hitherto have not been accessible via holonomic approaches, still correspond to holonomic programs (ie those programs which approximate that constant).

We propose to expand this work in three directions: first, to expand and solidify our early prototype implementation, second, to provide classification theorems detailing which programs exactly should be called “holonomic”, and third get a better understanding of the continuous case. It should be noted that even

though holonomy concerns itself with linear operators, the natural recurrences obtained from programs are non-linear; nevertheless, many of these are merely strange encodings of more complex linear recurrences. In the continuous case, it would make sense to ask if there is such a thing as (formal) Mellin/Laplace transforms of programs that effect the translation between continuous and discrete variables in the univariate case.

There are applications of these techniques to the areas of "reverse engineering" as well as "program understanding", for scientific programs [16]. For example, given either a Fortran 77 or a Maple program which purports to compute a special function via a series expansion, our prototypes will produce the system of equations that this special function must satisfy, and often even a closed-form as well. The user can then check that the program indeed corresponds to the intended function, a valuable aid in debugging such code.

## 3.5 Other New Projects

### 3.5.1 MapleNet Modelling

In mathematical modelling of complex systems, it is often important to be able to have an effective visual interface to the model. This aids researchers in understanding and developing the model, facilitates the use of mathematical models as educational tools, and enables model developers to communicate the modelling results to end-users.

The web browser is the most portable communications tool available on the internet and MapleNet provides the means for bringing mathematical modelling in Maple to the world wide web. The Complex Systems Modelling Group at IRMACS has developed a number of mathematical models.

(i) Models for the management of surgical wait lists have been developed for the BC Ministry of Health.

(ii) Models for capacity analysis of the integrated criminal justice system developed for the RCMP.

(iii) Modelling of the social determinants of obesity has been conducted with the Institute of Nutrition, Metabolism and Diabetes at SFU.

We propose to develop MapleNet implementations of some of these models. Also under consideration are developing models for the disease dynamics of HIV in collaboration with the BC Centre for Disease Control.

Exporting models to the world wide web will entail two steps. First, the model must be implemented in Maple. Our current surgical wait list models are being developed in Maple; however, some of our other simulation models utilise other software packages. Developing more modelling capacity in Maple will require writing some specialised modelling packages for Maple. Finally, the model must be "published from Maple to MapleNet. It is important that the user interface be both friendly and utilise visual presentations to aid understanding. To this end, specialised applets and maplets need to be written.

### 3.5.2 Parallel Integer Factorizaton

Borwein, Monagan, and Percival want to attempt a parallel implementation of the Self Initializing Quadratic Sieve integer factoring algorithm of Percival. The current performance is about six times faster than the quadratic sieve implementation in Magma at 90 digits. It should not be difficult to do this for a modest (2 to 4) number of processors as approximately 90% of the time is in the sieve (which is easily parallelized) and 10% is solving the linear system modulo 2 (not easily parallelized).

## 4. Development of Highly Qualified Personnel

a) Describe HQP involvement with partner organizations.

The personnel trained through the MOCAA consortium are in general PDFs, graduate students at the Masters and PhD level, and senior undergraduate students. They interact with faculty and industry personnel through

- professional conferences (see Section 5 Networking),
- MITACS internships which means 2 months on site at the company,
- the three group meetings at SFU, UW and UWO,
- the monthly seminar organized at UW and UWO,
- software training provided by company personnel.

Students and PDFs are encouraged to give talks and demo software at the biweekly group meetings at SFU, UW and UWO, present posters and demo software at the MITACS, CECM, ECCAD, ISSAC and MAPLE conferences/meetings. Also, the annual Maple conference in Waterloo is a prime opportunity for HQP to meet and interact with company personnel. HQP present posters, demo software and company personnel hold "technical chat" sessions.

Some of the HQP are involved in writing software which will be incorporated under contract into Maple. The students work by email and directly with someone from the Maple company on the installation.

b) Describe training activities initiated (summer schools, tutorials, curriculum development, etc.)

Faculty regularly provide additional courses in computer algebra and related topics for HQP to take. For example, this summer, Michael Monagan gave a special topics course MATH800 on computational algebraic geometry at SFU, next spring, Mark Giesbrecht will be giving an advanced symbolic computation course at UW and David Jeffrey will be teaching AM475/563 Applied Computer Algebra at UWO. Marc Moreno Maza taught CS 424, an introductory course to computer algebra in the last 3 academic years. He also taught several more advanced courses on symbolic polynomial system solving: CS 855b Symbolic Parallel Computation and Algebraic Geometry in 2005-2006 CS 867b Algorithmic properties of polynomial rings in 2004-2005, CS 652b - Algorithms and software for symbolic solvers of polynomial systems in 2003-2004 and CS 874b - Advanced computer algebra: asymptotically fast methods for exact computations in 2002-2003.

Each summer at Simon Fraser, Borwein and Monagan have three to six summer NSERC fellows (senior undergraduate students) working on various computer algebra/computational number theory projects to attract them to graduate school in these areas. We describe one project that the Maple company initiated.

**GraphTheory project.**

MapleSoft asked the group at SFU for help in developing a new package for graph theory. We approached this as a way to attract senior undergraduate students in discrete mathematics and/or computing

science to computer algebra, and as a training opportunity for our graduate students and PDFs to learn Maple, learn how to write software and learn how to prepare Maple documentation and tests. It was also an opportunity for us to get two colleagues at SFU, Peter Lisonek and Luis Goddyn, more involved in the project. We have had one PDF of Monagan and Lisonek (Jeff Farr), one PhD student of Lisonek, one PhD student of Goddyn, two Masters students of Monagan, and three senior undergraduate students in discrete mathematics and computing science work on the project.

For the first version of the package, which has been incorporated into Maple 11, we have focused on graph drawing, algorithms, graph coloring (Goddyn's PhD student, Mohammad Ghebleh is working in this area), and other NP-complete and NP-hard problems.

It was a success because it was doable and directly of interest to the students. The students were able to present their work (two papers, two posters and demos) to Maple company personnel, who got excited about this work.

The three senior undergraduate students who had taken a first course in computer algebra worked on one aspect of the graph theory package for two consecutive summers as well as on a computer algebra problem. Al Erickson worked on generating random graphs and polynomial factorization, Moe Ebrahimi worked on graph drawing and first order systems of ODE, and Simon Lo worked on graph algorithms and computing characteristic polynomials of integer matrices. It took the students one summer to become productive. Each has produced a poster, given a talk and a demo at the Maple conference. Ebrahimi has gone to grad school in applied mathematics at UCSD (his parents live in San Diego). Lo is starting grad school at SFU in computer algebra in January 2007. Erickson is completing his 4th year.

# 5. Networking

In this section we provide details of future planned networking including those with industry partners (involving diffusion of results, technology transfer, collaborative research or industry linkage), both within the project and within MITACS themes.

**Regular Meetings and Seminars.**

The Computer Algebra Group at SFU, the Symbolic Computation Group at UW, and the Symbolic Computation Lab at UWO hold weekly meetings/seminars. The two groups at Waterloo and Western collectively form the Ontario Research Center for Computer Algebra (ORCCA).

ORCCA meets regularly in London and Waterloo, along with affiliated members Jacques Carette from McMaster in Hamilton, David Aruliah of UOIT, and members from the Math group at MapleSoft on the second Friday of every month for a seminar, followed by a poster session and networking over lunch.

Individuals from the two groups at UW and UWO meet regularly with people at MapleSoft.

**Conference Involvement.**

Many of the faculty and students on the project, and people from MapleSoft have attended and will continue to attend the following annual scientific meetings.

- Joint MITACS/CMS Conference in Winnipeg, May 30 - June 3, 2007. There will be a special session on computer algebra organized by Michael Monagan at the CAIMS summer meeting in Winnipeg which will primarily involve members of our MOCAA project.
- International Symposium on Symbolic and Algebraic Manipulation (ISSAC). Since next years ISSAC in Waterloo, Ontario, July 29 to August 2, 2007 this will be a second opportunity for everyone to meet.
- Maple Conference, Wilfred Laurier University, date expected to immediately follow the ISSAC meeting next year. This is an opportunity for networking with members of the company. It is also an opportunity for students, pdfs, and faculty from Simon Fraser University to present their work to Maple company personnel as happened in 2004, 2005, and 2006.
- East Coast Computer Algebra day, held in the spring.

There are two “satellite” workshops that we are organizing in 2007 around the ISSAC meeting in Waterloo.

- Stephen Watt is organizing the Symbolic-Numeric Computation (SNC ’07) conference for July 25-27, 2007.
- Marc Moreno-Maza is organizing the Parallel Symbolic Computation (PASCO ’07) conference for July 27-28, 2007.

Other events we are involved in 2007.

- The group at SFU organizes an annual one day meeting, CECM day in Computational Mathematics, held late summer.

- CAIMS '07, Banff, May 20-24, 2007 will have a mini symposium on Spectra and Pseudospectra of Matrix Polynomials: confirmed speakers. Amir Amiraslani, Rob Corless, Nick Higham, Peter Lancaster, Nargol Rezvani, Azar Shakoori, three of whom were MITACS-funded students (Amir is now a PIMS postdoc with Peter Lancaster).

Other individual networking trips.

- Bryan Krawetz from MapleSoft will visit SFU later this fall (2006) to work with Michael Monagan on a specific project.
- Juergen Gerhard from MapleSoft will run a "Maple Training Days" event later this fall (2006) at MapleSoft.
- Juergen Gerhard will again visit SFU in early 2007 for a Maple training session and project work.
- Keith Geddes from Waterloo is on sabbatical in Vancouver with the SFU group for September 2006 through June 2007.

# 6. Knowledge Exchange and Technology Transfer

"Describe the Intellectual Property (IP) generated, including software, since the last Project CV (November 2004). Describe the state of the IP and its readiness for use by other institutions or industry, if applicable. Indicate wehether the IP has or will be licensed. Are patents or other protection being sought?"

The IP generated by our MOCAA project consists mainly of Maple software which includes Maple programs, C programs, documentation, Maple demo worksheets, and test files.

**Proprietary software packages.**

The contributions listed here have been contributed in the period November 1, 2004 through November 1, 2006 under a contract with Maplesoft. Contributions that have already been integrated into Maple 10 or Maple 11 are identified.

1. A. Bradford and M. Monagan (2006), Maple 11. A Maple implementation of the index calculus algorithm for computing discrete logarithms in the integers modulo $p$.

2. Y. Brychkov, E. Cheb-Terrab, and K. Geddes (2005), Maple 10. Differentiation of Bessel and Struve functions wrt a parameter.

3. R. Corless and D. Jeffrey (2005), Maple 10. Maple code the numerical evaluation, power series expansion, and simplifications of the WrightOmega function. Numerical evaluation is nontrivial near the branch cuts.

4. R. Corless, M. Benghorbal, R. Morris, and E. Cheb-Terrab (2005), Maple 10. Maple code for fractional-order differentiation.

5. R. Corless and E. Cheb-Terrab (2005) Maple 10. Maple code for symbolic order differentiation.

6. Robert Corless (2006), Maple 11. Matrix polynomial support for the Bernstein basis, the Lagrange basis, and some bug fixes. Strong linearizations implemented in CompanionMatrix allow nonlinear eigenvalue problems to be solved by converting them to generalized eigenvalue problems. The linearization for the Lagrange basis is new. Numerical evaluation uses the barycentric form. The de Casteljau algorithm was implemented for evaluation in the Bernstein case.

7. R. Corless and N. Rezvani (2006). Nearest polynomial with a given zero. This routine in the SNAP package uses the witness vector from Holder's inequality to find the nearest polynomial (in any p-norm on the vector of coefficients, in any of several polynomial bases) with a given zero. The zero may be at infinity. Linear constraints may be supplied, allowing structured problems to be solved.

8. M. Ebrahimi, M. Monagan, A. Wittkopf (2005), Maple 11. DEplot: Maple and C code for visualizing the direction field of a first order system system of differential equations with randomly placed little fish.

9. M. Ebrahimi and M. Monagan (2006), Maple 11. DEplot[interactive]: A Maple GUI for visualizing first order systems of DEs which contains a database of known systems, e.g., the Kermack McKendrick SIR model.

10. M. Ebrahimi, J. Farr, M. Ghebleh, L. Goddyn, S. Javadi, M. Khatarinejard-Fard, S. Khodadad, P. Lisonek, S. Lo, M. Monagan, and A. Wittkopf (2006) Maple 11. GraphTheory: A Maple package for Graph drawing and Graph theoretic algorithms. Includes facilities for drawing graphs in 2D and 3D, generating random graphs, exporting and importing graphs, and a library of known special graphs.

11. Jeff Farr (2005). A Maple package for multivariate polynomial and rational function interpolation and multivariate Pade approximation. The algorithms generalize Newton interpolation using ideas from the theory of Groebner bases.

12. F. Lemaire, M. Moreno Maza, and Y. Xie. (2004-2005) Maple 10. RegularChains: A Maple library of 49 routines for solving systems of algebraic equations using triangular decomposition techniques and for studying properties of the solutions. Provides automatic case discussion and case recombination, and facilities for linear algebra over a direct product of fields.

13. M. Moreno Maza, and Y. Xie. (2004-2005) mpldoc (Maple 11). Specifications and a prototype in C for structured comments in Maple source code.

14. F. Lemaire, M. Moreno Maza, and Y. Xie with contributions from X Jin, E. Schost, and W. Wu (2005-2006), Maple 11. Modular algorithms for triangular decomposition techniques, (direct solving and change of variable ordering) based on new lifting techniques.

15. S. Lo and M. Monagan (2006), Maple 11. LinearAlgebra:-Generic: A package of "generic" Maple routines for doing linear algebra over commutative rings, integral domains, Euclidean domains and fields. Provides a simple mechanism for coding generic algorithms in Maple.

16. S. Lo, M. Monagan and A. Wittkopf (2005), Maple 11. A Maple implementation of a modular algorithm for computing the characteristic polynomial of an integer matrix using 25 bit floating point primes.

17. S. Lo, M. Monagan and A. Wittkopf (2006), Maple 11. StronglyConnectedBlocks: A Maple procedure for finding the block structure of a matrix. Used to compute determinants and characteristics polynomials of sparse matrices.

18. Michael Monagan (2005), Maple 11. Maple code for computing the GCD of two univariate polynomials over an algebraic function field with one or more parameters. Uses the dense modular algorithm of Monagan and van Hoeij.

19. M. Monagan and R. Pearce (2006). QuotientRings: A Maple package for computations in quotient rings. Includes algorithms for simplifying a fraction in a quotient ring, i.e., simplifying fractions modulo polynomial side relations.

20. Roman Pearce (2006), Maple 11. A Maple implementation of the “F4 algorithm” for computing Groebner bases with rational coefficients using modular linear algebra.

21. C. Percival and P. Borwein (2006), contract pending. An implementation of a self initializing quadratic sieve algorithm for factoring integers.

22. E. Cheb-Terrab (2005), Maple 10. Maple code for the solution of ODEs of Heun type and support for the numerical evaluation, differentiation, series expansion, and simplification of the family of Heun functions.

**Other: publically available software.**

1. J. Borwein and C. Hamilton (2006), `http://ddrive.cs.dal.ca/projects/scat` . SCAT (Symbolic Convex Analysis Toolkit): A Maple package for computing symbolic subdifferentials and (Fenchel) conjugates for real valued functions in one or more dimensions.

2. Akpodigha Filatei, Xin Li, Marc Moreno Maza (2004-2006) Asymptotycally fast algorithms for univariate and multivariate polynomial arithmetic (C, Aldor and AXIOM code). `http://www.csd.uwo.ca/People/gradstudents/xli96/project-00-page.html`

# 7. Additional Information.

The Maple project, represented by George Labahn, Keith Geddes, Stephen Watt, Robert Corless and Michael Monagan, and MapleSoft, were awarded an NSERC Synergy award in late 2004.

This is a major award for university-industry collaboration.

Robert Corless has been awarded the "Distinguished University Professor" title in 2006 at the University of Western Ontario.

**Project Management**

In the two period November 2004 through November 2006, the management team was

Peter Borwein, SFU (Consortium Leader)

Robert Corless, UWO

George Labahn and Mark Giesbrecht, Waterloo (Eastern Group Leader)

Michael Monagan, SFU (Western Group Leader).

Peter Borwein has been leading the project since 1998. He has asked to step down as overall leader so that he has more time for science. The new management team will be

Robert Corless, UWO,

George Labahn, Waterloo (Eastern Group Leader) and,

Michael Monagan, SFU (Western Group Leader).

with Labahn and Monagan co-leading the project.

**Project Funding**

In the last two years our mitacs project has received in excess of $240,000 cash from MapleSoft. For this application Dr. Bernardin from MapleSoft (see attached letter) has offered a modest increase in support to $145,000 ( = $100,000 cash plus $45,000 for six mitacs internships ) per year for the two period April 1, 2007 through March 31, 2009.

We expect also to have $50,000 in funding per year available to support a new initiative from the RCMP. We have not included the contact information yet because Peter Borwein is still waiting to get a formal okay from the RCMP.

## References


[1] LinBox Project. *Exact computational linear algebra*. http://www.linalg.org/.

[2] S. A. Abramov, M. A. Barkatou, and M. van Hoeij. Apparent singularities of linear difference equations with polynomial coefficients. *Appl. Algebra Engrg. Comm. Comput.*, 17(2):117–133, 2006.

[3] S. A. Abramov and M. van Hoeij. Desingularization of linear difference operators with polynomial coefficients. In *Proceedings of the 1999 International Symposium on Symbolic and Algebraic Computation*, pages 269–275, Vancouver, Canada, 1999. ACM.

[4] A. Amiraslani, R. M. Corless, L. Gonzalez-Vega, and A. Shakoori. Polynomial algebra by values. Technical Report TR-04-01, Ontario Research Centre for Computer Algebra, http://www.orcca.on.ca/TechReports, 2004.

[5] G. G. B. Beckermann and G. Labahn. On the numerical condition of a generalized hankel eigenvalue problem. *To appear in Numerische Mathematik*, page 23 pages, 2006.

[6] G. L. B. Beckermann and G. Villard. Normal forms for general polynomial matrices. *Journal of Symbolic Computation*, 41(6):708–737, 2006.

[7] D. Bailey, J. Borwein, and R. Crandall. Integrals of the ising class. *J. Phys. A.*, 39:12271–12302, 2006.

[8] S. Barnett. *Polynomial and Linear Control Systems*. Marcel-Dekker, 1983.

[9] B. Beckermann and G. Labahn. A uniform approach for the fast computation of matrix-type pade approximants. *SIAM J. Matrix Analysis and Applications*, pages 804–823, 1994.

[10] J. Borwein, D. Bailey, N. Calkin, R. Girgenson, R. Luke, and V. Moll. *Experimental Mathematics in Action*. A.K. Peters, 2006.

[11] A. Bostan, P. Flajolet, B. Salvy, and É. Schost. Fast computation of special resultants. *Journal of Symbolic Computation*, 41(1):1–29, 2006.

[12] A. Bostan, C. Jeannerod, and É. Schost. Using fast matrix multiplication to solve structured linear systems, 2006. Extended abstract for talk given at Dagstuhl Seminar No. 06271: Challenges in Symbolic Computation Software, July 2–7, 2006.

[13] A. Bostan and É. Schost. Polynomial evaluation and interpolation on special sets of points. *Journal of Complexity*, 21(4):420–446, 2005.

[14] F. Boulier, M. Moreno Maza, and C.Oancea. A new henselian construction and its application to polynomial gcds over direct products of fields. In *proceedings of EACA'04*, Universidad de Santander, Spain, 2004.

[15] Y. Brychkov and K. Geddes. Differentiation of hypergeometric functions with respect to parameters. pages 15–28. World Scientific, 2004.

[16] J. Carette. Reverse engineering of holonomic functions and sequences from imperative scientific computation code. 2006.

[17] B. Carlson. Toward symbolic integration of elliptic integrals. *Journal Symbolic Computation.*, 28:739–753, 1999.

[18] E. Cartan. *Les problèmes d'équivalence*, volume 2 of *oeuvres complètes*, pages 1311–1334. Gauthiers–Villars, Paris, 1953.

[19] O. Carvajal, F. Chapman, and K. Geddes. Hybrid symbolic-numeric integration in multiple dimensions via tensor-product series. In M. Kauers, editor, *ISSAC 2005*, pages 84–91, Beijing, China, 2005. ACM Press.

[20] O. A. Carvajal. A hybrid symbolic-numeric method for multiple integration based on tensor-product series approximations. M.Math thesis, School of Computer Science, University of Waterloo, 2004.

[21] F. W. Chapman. *Generalized Orthogonal Series for Natural Tensor Product Interpolation.* Ph.D. thesis, School of Computer Science, University of Waterloo, 2003.

[22] H. Cheng and G. Labahn. On computing polynomial gcd in alternate bases. In *Proceedings of ISSAC'06, Genoa, Italy*, pages 47–54. ACM Press, 2006.

[23] F. Chyzak, M. Mishna, and B. Salvy. Effective scalar products of d-finite symmetric functions. *J. Combin. Theory Ser. A*, 112(1):1–43, 2005.

[24] F. Chyzak and B. Salvy. Non-commutative elimination in Ore algebras proves multivariate holonomic identities. *Journal of Symbolic Computation*, 26(2):187–227, Aug. 1998.

[25] R. Corless, P. Gianni, B. Trager, and S. Watt. The singular value decomposition for polynomial systems. In *Proc. ISSAC'95*, pages 96–103.

[26] R. M. Corless. On a generalized companion matrix pencil for matrix polynomials expressed in the Lagrange basis. In D. W. . L. Zhi, editor, *Proc. Symbolic-Numeric Computation*, pages 1–18, Xi'an, China, 2005.

[27] R. M. Corless and S. M. Watt. Bernstein bases are optimal, but, sometimes, Lagrange bases are better. In *Proceedings of SYNASC, Timisoara*, pages 141–153. MIRTON Press, September 2004.

[28] X. Dahan, X. Jin, M. Moreno Maza, and É. Schost. Change of ordering for regular chains in positive dimension. In I. S. Kotsireas, editor, Maple Conference 2006, pages 26–34, 2006.

[29] X. Dahan, M. Moreno Maza, É. Schost, W. Wu, and Y. Xie. Lifting techniques for triangular decompositions. In *ISSAC'05*, pages 108–115. ACM Press, 2005. ISSAC'05 Distinguished Student Author Award.

[30] X. Dahan, M. Moreno Maza, É. Schost, and Y. Xie. On the complexity of the D5 principle. In *Proc. of* Transgressive Computing 2006, Granada, Spain, 2006.

[31] X. Dahan and É. Schost. Sharp estimates for triangular sets. In *Proc. ISSAC 04*, pages 103–110. ACM Press, 2004.

[32] W. Eberly, M. Giesbrecht, P. Giorgi, A. Storjohann, and G. Villard. Solving sparse rational linear systems. In J.-G. Dumas, editor, *Proc. ISSAC 2006*. ACM Press, 2006.

[33] A. Filatei, X. Li, M. Moreno Maza, and É. Schost. Implementation techniques for fast polynomial arithmetic in a high-level programming environment. In *Proc. ISSAC'06*, pages 93–100. ACM Press, 2006.

[34] I. Gessel. Symmetric functions and p-recursiveness. *J. Combin. Theory Ser. A*, 53(2):257–285, 1990.

[35] M. Giesbrecht, G. Labahn, and W. s Lee. Symbolic-numeric sparse polynomial interpolation in chebyshev basis and trigonometric interpolation. In *Proc. Workshop on Computer Algebra in Scientific Computation (CAS C)*. World Scientific, 2004.

[36] P. Giorgi, C.-P. Jeannerod, and G. Villard. On the complexity of polynomial matrix computations. In R. Sendra, editor, *ISSAC 2003*, pages 135–142. ACM Press, New York, 2003.

[37] O. Golubitsky, M. Kondratieva, M. Moreno Maza, and A. Ovchinnikov. Bounds for algorithms in differential algebra. Submitted to J. Symb. Comp., 2006.

[38] M. v. Hoeij and M. Monagan. Algorithm for polynomial gcd computation over algebraic function fields. In *Proc. ISSAC 2004*. ACM Press, 2004.

[39] F. Lemaire, M. Moreno Maza, and Y. Xie. The `RegularChains` library. In Ilias S. Kotsireas, editor, Maple Conference 2005, pages 355–368, 2005.

[40] F. Lemaire, M. Moreno Maza, and Y. Xie. Making a sophisticated symbolic solver available to different communities of users. In *Proc. of* Asian Technology Conference in Mathematics*'06*, 2006.

[41] X. Li and M. Moreno Maza. Efficient implementation of polynomial arithmetic in a multiple-level programming environment. In A. Iglesias and N. Takayama, editors, *Proc. Mathematical Software - ICMS 2006*, pages 12–23. Springer, 2006.

[42] G. L. M. Giesbrecht and W. s Lee. Symbolic-numeric sparse interpolation of multivariate polynomials. In *Proceedings of ISSAC'06, Genoa, Italy*, pages 116–123. ACM Press, 2006.

[43] M. Moreno Maza and Y. Xie. An implementation report for parallel triangular decompositions on a shared memory multiprocessor. In *Proc. of* Symposium on Parallelism in Algorithms and Architectures*'06*. ACM Press, 2006.

[44] M. Moreno Maza and Y. Xie. Solving polynomial systems symbolically and in parallel. Submited to IPDPS'07, 2006.

[45] L. Meunier and B. Salvy. ESF: An automatically generated encyclopedia of special functions. In J. R. Sendra, editor, *Proc. of ISSAC'03*, pages 199–205. ACM Press, 2003.

[46] M. Moreno Maza. On triangular decompositions of algebraic varieties. Technical Report TR 4/99, NAG Ltd, Oxford, UK, 1999. http://www.csd.uwo.ca/∼moreno.

[47] M. Moreno Maza, G. Reid, R. Scott, and W. Wu. On approximate triangular decompsoitions i. dimension zero. In D. M. Wang and L. Zhi, editors, *Symbolic-Numeric Computation*, Xi'an, China, 2005.

[48] M. Moreno Maza, G. Reid, R. Scott, and W. Wu. On approximate triangular decompositions in dimension zero. *J. Symb. Comp*, 2007. To appear.

[49] S. Neut and M. Petitot. La géométrie de l'équation $y''' = f(x, y, y', y'')$. *C. R. Acad. Sci. Paris*, (335):515–518, 2002.

[50] C. Oancea, J. Selby, M. Giesbrecht, and S. Watt. Distributed models of thread-level speculation. In *Proc. 2005 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'05)*, pages 920–927. CSREA Press, 2005.

[51] W. Pan and D. Wang. Uniform gröbner bases for ideals generated by polynomials with parametric exponent. In *Proc. ISSAC 2006*, pages 269–276. ACM Press, 2006.

[52] B. Salvy. Computation of convolution integrals, 2000. Talk at MSRI.

[53] B. Salvy and P. Zimmermann. Gfun: a Maple package for the manipulation of generating and holonomic functions in one variable. *ACM Transactions on Mathematical Software*, 20(2):163–177, 1994.

[54] É. Schost. Complexity results for triangular sets. *J. Symb. Comp.*, 36(3-4):555–594, 2003.

[55] H. Tsai. Weyl closure of a linear differential operator. *J. Symbolic Comput.*, 29(4-5):747–775, 2000.

[56] S. M. Watt. Algorithms for symbolic polynomials. In *Proc. 9th International Workshop on Computer Algebra in Scientific Computing (CASC 2006)*, page 302. Springer Verlag LNCS 4194, 2006.

[57] S. M. Watt. Making computer algebra more symbolic. In *Proc. Transgressive Computing 2006: A conference in honor or Jean Della Dora (TC 2006)*, pages 43–49, 2006.