TRIP - a computer algebra system dedicated to celestial mechanics and perturbation series

Mickaël Gastineau
Astronomie et Systèmes Dynamiques
IMCCE-CNRS UMR8028, Observatoire de Paris, UPMC
77 Avenue Denfert-Rochereau
75014 Paris, France
gastineau@imcce.fr

Jacques Laskar
Astronomie et Systèmes Dynamiques
IMCCE-CNRS UMR8028, Observatoire de Paris, UPMC
77 Avenue Denfert-Rochereau
75014 Paris, France
laskar@imcce.fr

ABSTRACT

TRIP is an interactive computer algebra system that is devoted to perturbation series computations, and specially adapted to celestial mechanics. Its development started in 1988, as an upgrade of the special purpose FORTRAN routines elaborated by J. Laskar for the demonstration of the chaotic behavior of the Solar System. TRIP is now a mature and efficient tool for handling multivariate generalized power series, and contains also a numerical kernel. A limited version of TRIP is freely available on the WEB at www.imcce.fr/trip.

1. INTRODUCTION

Since the onset of computers, astronomers have searched to automate the computation of the analytical series that Le Verrier or Delaunay used to perform by hand in daunting tasks that lasted for several years. Astronomers have thus been among the first to use computers to perform analytical calculations, developing programs to manipulate their series. These programs were usually dedicated and more like a set of routines usable through a programming language. Among them, one should quote MAO, written in FORTRAN and then PL/1[32, 10], TRIGMAN (in FORTRAN)[22, 23], CAMAL [2, 3, 14] as well as several other ones between 1970 and 1990 that were written in various languages, such as FORTRAN[5, 9, 6, 8] or LISP[10]. For a detailed review on computer algebra systems in celestial mechanics, one can consult [4, 20, 7]. A spectacular outcome of these efforts was the verification of Delaunay's Lunar theory, which revealed to be correct up to order 9, except for a one erroneous coefficient at order 7 in the computation of the reduced Hamiltonian[11].

Twenty years ago, none of the general purpose computer algebra systems could handle the most standard perturbation computations in celestial mechanics. Therefore, when J. Laskar undertook his analytical averaging of the gravitational equations for the computation of the long-time behavior of the Solar System, he also had to write some very dedicated FORTRAN programs that hid most of the analytical computations in the series products or derivations[24]. Using these tools, he could then compute to a never reached state a 150 000 terms complex polynomial in 32 variables that was then numerically integrated for the demonstration that the Solar System planetary motion is chaotic[25, 26].

The problem of these efficient, but dedicated routines is that they were not very flexible and requested specific programming for each different application. The absence, at this time, of any available efficient general computer algebra tool led then J. Laskar to start the development of TRIP, an interactive computer algebra system dedicated to celestial mechanics and perturbation methods[27].

Although it has been essentially developped by two persons (J. Laskar and M. Gastineau since 1998), with the help of several master students, TRIP is now a mature tool, that has somewhat reached our goal in being both as efficient as the first dedicated FORTRAN routines used for the Solar System stability computations, and as easy to use as a general computer algebra system. TRIP is currently used in our group for celestial mechanics computations[28, 29, 31], and the numerical kernel with limited algebraic capabilities is freely available on the WEB at www.imcce.fr/trip.

2. FEATURES

TRIP has its own procedural language and provides a command line interface to execute the user command. This procedural language supports loops, such as while and for statements, and conditional instructions, such as if-then-else and switch-case statement. Function arguments are passed by value or by reference. The interpreter converts the source code to an abstract syntax tree to optimize the code before execution. TRIP embeds two kernels, a symbolic and a numerical kernel. This numerical kernel communicates with Gnuplot [21] or Grace [33] to plot the graphics and allows one to plot the numerical evaluation of symbolic objects.

Several kinds of objects are managed by this interpreter, such as variables, characters string, arrays, numerical vectors, truncations, series, ... In TRIP, the truncations are indeed formal objects. Our software is very flexible concerning truncation, which can be defined as total degree in various sets of variables, or partial degrees in a given variable. All these can be combined as lists. The user thus defines truncation independently of the computations, and activates a truncation when needed. The multivariate generalized power series (1) are the main objects in the symbolic kernel [27].

$$S(X_{1\ldots n}, \lambda_{1\ldots m}) = \sum C_{j,k} X_1^{j_1} \ldots . X_n^{j_n} \exp^{\sqrt{-1}(k_1\lambda_1 + \ldots . + k_m\lambda_m)} \tag{1}$$

$$j_n \in \mathbb{Z} \text{ and } k_m \in \mathbb{Z}$$

The complex expressions of the form $\exp^{\imath\lambda_m}$ are encoded as variables, so negative exponents are permitted. The number and the degree of the variables in these series is limited to a signed 32-bit integer on all operating systems. The order of the variables is automatically handled by the kernel or can be changed by the user. The coefficients $C_{j,k}$ can be numerical coefficients or rational functions. The user controls the kind of numerical coefficients used during the session. Several types of numerical coefficients are supported

- double and quadruple precision floating-point numbers.
- multiple-precision floating-point numbers through the MPFR [15] and MPC [12] library.
- multiple precision integer or rational numbers through the GNU MP library [19].
- fixed rational numbers.
- double and quadruple precision floating-point interval.
- complex numbers with the same previous attributes.

Series representation

As TRIP is tuned to compute large series with millions of terms depending on several variables, the internal representation of these series in the computer memory should depend on their sparsity in the studied problem. For example, some symmetries are present in celestial mechanics problems, such as d'Alembert relations in the planetary motion (see Laskar, [24]). This implies that sparse series appear during computations. Most of computer algebra systems support only one representation for polynomials or power series, such as recursive list or distributed list. Instead TRIP supports multiple memory representations of the series, which are selected by the user.

- **recursive sparse**. The polynomial in n variables is considered as a polynomial in one variable with coefficients in the polynomial ring in $n-1$ variables. The polynomial is stored in a recursive list. Each element of this list contains the exponent and the non-zero coefficients. These coefficients are in the polynomial ring in $n-1$ variables and are recursive lists too. When a coefficient is a polynomial, a reference to this polynomial is stored in the element.
- **recursive dense**. This representation is similar to the previous one but instead of storing the coefficient and exponent in a recursive list, only the coefficients are stored in a dense vector. The minimal and maximal degree are kept in the header of the array. All coefficients are stored, as well as the zero coefficients, between the minimal and maximal degree.
- **homogeneous polynomials**. The recursive structures introduce many memory references which produce less locality of the data. Indeed, most of the time, the pointer does not reference the memory block just after the structure which owns this pointer but it references any location in memory. Modern processors have multiple levels of cache (up to 3 for the moment) which have different access time. The closer the data is from the processor element, the faster is the access to the data.

  This representation stores in the same memory block, noted $BH_\delta(X_1, ..., X_n)$, all terms which have the same total degree δ. $S(X_{1\ldots n}, \lambda_{1\ldots m})$ is represented as

  $$S(X, \lambda) = \sum_{\delta,k} BH_\delta(X_{1\ldots n}) \exp^{\sqrt{-1}(k_1\lambda_1 + \ldots . + k_m\lambda_m)}$$

  TRIP stores the numerical coefficients of the terms of degree δ, even zero, into the same array. The exponents of the homogeneous polynomials are kept in a shared table for the degree δ. As the size of the block $BH_\delta(X_1, ..., X_n)$ is $C^\delta_{\delta+n-1} = \frac{(\delta+n-1)!}{\delta!(n-1)!}$, only non-zero coefficients, combined with an index array to the exponent table, are stored in the arrays if the series are sparse. These blocks could not store variables with negative exponents.
- **d'Alembert polynomials**. The d'Alembert relations on the degree of the variables in the planetary motion imply that some terms never exist during the computation of the series. So the exponent table of polynomials could be rewritten taking into account these relations. The size of the exponent table is reduced by a large factor. The d'Alembert representation is the same as the homogeneous representation except the exponent table.

In the recursive data structures, as the leaf nodes always contain numerical coefficients, the generic containers of the leaf nodes could be replaced by optimized containers. These generic containers require additional processing to check the data type and select the adequate algorithm for the computation. This improvement reduces the execution time by a factor 2 with multiple-precision integers [17] and up to a factor 4 with the double-precision floating-point numbers due to the better usage of the SIMD pipelines.

Operations

Many operations on these series are available, such as derivation, integration, exponentiation, fast evaluation, substitution and selection of terms, ... TRIP embeds useful functions for the Celestial mechanics, such as Poisson brackets, expansions of the basic functions of the two-body problem (r/a, $\cos E$, ...).

As these operations rely on products, the series multiplication has been tuned and takes advantage of the multiple processors and cores available on several computers. The naive (term by term) algorithm for the product is used because the series are sparse and have low degrees in most of celestial mechanics problems. To reduce the creation of many temporary objects during the multiplication of the multivariate polynomials, a *Fused-Mutiply-Add* algorithm [17] is implemented. The computation is distributed on the multiple processor elements using a task-stealing model. Each thread has its own work-queue. If a thread becomes idle, it steals some work from other threads queues. The recursive data structures require many memory allocations, such as elements in the list. But the operating system allocators do not have linear speedup when many memory operations are performed at the same time. Therefore, TRIP uses its own lock-free allocators to avoid this bottleneck.

Instead of performing the full product, the product could be truncated on the partial or total degree of one or more variables. To take this into account, TRIP embeds a *monomial truncated product* which computes the product of the series but keeps only the terms satisfying some degree condition. The numerical coefficients are often decreasing according to the degree of the variables, so a truncation on the amplitude of the coefficient is available to keep fast operation. The truncated product uses the same way as the full product to distribute the computation on multiple cores.

Communications

As users need functions which are not available in TRIP but exist in some more general computer algebra system, such as Maple, a communication layer is available to exchange objects between TRIP and the other computer algebra systems and to perform remote computations on them. The first mechanism uses the standard MathML 2.0 [1] to exchange data with Maple. But the semantics of the exchanged objects is not well defined in the standard MathML. The new communication layer uses the emerging "Symbolic Computation Software Composability Protocol" (SCSCP) [16], based on the remote procedure call model and allows the communications between any SCSCP-compliant software located on the same or different machine. This protocol encodes the messages and the mathematical objects in the OpenMath format [34]. TRIP uses the open-source "SCSCP C Library" [18] to support this protocol.

3. PERFORMANCE

To compare the efficiency of the implementation of the full product on sparse polynomials, the following benchmarks from [13] and [30] compare the representations of TRIP to other computer algebra systems on one core. Only the SDMP library [30] and TRIP take advantage of multiple cores, their comparisons are performed on eight cores too.

The following computations are performed over $\mathbb{Z}$ on an Intel Xeon 5570 with 8 cores. The integers are encoded using a mixed representation, hardware integers or GMP integers depending on their value, for TRIP, Maple and SDMP. The representations of TRIP are recursive dense with a generic container for the leaf nodes (RDG), recursive sparse with a generic container for the leaf nodes (RSG), recursive dense with an optimized container for the leaf nodes (RDO), recursive sparse with an optimized container for the leaf nodes (RSO) and homogeneous polynomials (BH).

		time (seconds)			
		$p_1 = f_1 \times g_1$		$p_2 = f_2 \times g_2$	
		1 core	8 cores	1 core	8 cores
Maple 13		1943.70		2310.23	
Singular 3.1.1		720.18		398.86	
SDMP		59.83	3.87	13.29	11.63
TRIP 1.1	RDG	59.89	7.82	22.47	3.32
	RSG	65.26	8.72	19.64	2.99
	RDO	22.56	3.02	19.63	3.03
	RSO	28.72	3.76	16.54	2.52
	BH	3.99	0.53	64.42	14.29

$$\text{with}\quad \begin{array}{lcl} f_1 & = & (1+x+y+z+t)^{30} \\ g_1 & = & f_1 + 1 \\ f_2 & = & (1+x+y+2z^2+3t^3+5u^5)^{16} \\ g_2 & = & (1+u+t+2z^2+3y^3+5x^5)^{16} \end{array}$$

TRIP has better timing if the optimized containers are used for the leaf nodes. SDMP has similar timings to the generic containers of TRIP on the first example. With 8 cores, SDMP has a super linear speedup and TRIP has only a linear speedup for all representations. On the second example, SDMP has better timings on one core but has difficulty in taking advantage of eight cores. It reduces the execution timing to 5.7 seconds (speedup of 2.33) with only 5 threads. TRIP continues to have almost linear speedup on eight cores and obtains much better timings.

The timings of the homogeneous polynomials do not include the time to initialize the addressing table for the product. This initialization is performed only one time per session. The time to build the addressing table is about 24 seconds for the $f_1 \times g_1$.

4. PRESENTATION AT ISSAC

The presentation will consist of

1. A slide explaining the reasons for the design of TRIP.
2. A slide giving the features available in TRIP.
3. A picture of the internal architecture.
4. A slide showing the representation of the series available in the software.
5. A benchmark slide detailing the recent improvement of the multiplication of the polynomials.
6. A demonstration of the usage of SCSCP to compute the expansion of the two-body problem on the TRIP server-side and retrieve the result in a second computer algebra system.
7. A demonstration of the computation of the disturbing function and showing the cpu usage.

5. REFERENCES

[1] R. Ausbrooks, S. Buswell, D. Carlisle, S. Dalmas, S. Devitt, A. Diaz, M. Froumentin, R. Hunter, P. Ion, M. Kohlhase, R. Miner, N. Poppelier, B. Smith, N. Soiffer, R. Sutor, and S. Watt. Mathematical markup language (mathml) version 2.0 (2nd edition). World Wide Web Consortium, Recommendation REC-MathML2-20031021, October 2003.

[2] D. Barton, S. R. Bourne, and J. P. Fitch. An algebra system. *The Computer Journal*, 13(1):32–39, Jan. 1970.

[3] D. Barton, S. R. Bourne, and J. R. Horton. The structure of the cambridge algebra system. *The Computer Journal*, 13(3):243–247, Mar. 1970.

[4] D. Barton and J. P. Fitch. Applications of algebraic manipulation programs in physics. *Reports on Progress in Physics*, 35:235–314, Jan. 1972.

[5] R. Broucke and K. Garthwaite. A Programming System for Analytical Series Expansions on a Computer. *Celestial Mechanics*, 1:271–284, June 1969.

[6] R. A. Broucke. A FORTRAN-based Poisson Series Processor and its Applications in Celestial Mechanics. *Celestial Mechanics*, 45:255–265, 1989.

[7] V. A. Brumberg. *Analytical Techniques of Celestial Mechanics*. Springer-Verlag, 1995.

[8] V. A. Brumberg, S. V. Tarasevich, and N. N. Vasiliev. Specialized celestial mechanics systems for symbolic manipulation. *Celestial Mechanics and Dynamical Astronomy*, 45(1):149–162, Mar. 1988.

[9] J. R. Cherniack. A more general system for poisson series manipulation. *Celestial Mechanics and Dynamical Astronomy*, 7(1):107–121, Jan. 1973.

[10] A. Deprit. Simplify or Perish. *Celestial Mechanics*, 45:189–200, 1989.

[11] A. Deprit, J. Henrard, and A. Rom. Lunar ephemeris: Delaunay's theory revisited. *Science*, 168:1569–1570, June 1970.

[12] A. Enge, P. Théveny, and P. Zimmermann. *mpc — A library for multiprecision complex arithmetic with exact rounding*. INRIA, 0.8.2 edition, May 2010. `http://mpc.multiprecision.org/`.

[13] R. Fateman. Comparing the speed of programs for sparse polynomial multiplication. *SIGSAM Bull.*, 37(1):4–15, 2003.

[14] J. Fitch. CAMAL 40 years on – is small still beautiful? In *Intelligent Computer Mathematics*, pages 32–44. Springer-Verlag, 2009.

[15] L. Fousse, G. Hanrot, V. Lefèvre, P. Pélissier, and P. Zimmermann. Mpfr: A multiple-precision binary floating-point library with correct rounding. *ACM Trans. Math. Softw.*, 33(2):13, 2007.

[16] S. Freundt, P. Horn, A. Konovalov, S. Linton, and D. Roozemond. *Symbolic Computation Software Composability Protocol (SCSCP) specification*. http://www.symbolic-computation.org/scscp, version 1.3 edition, 2009.

[17] M. Gastineau. Parallel operations of sparse polynomials on multicores - 1. multiplication and poisson bracket, 2010. *Submitted to Parallel Symbolic Computation 2010, 23-26 July 2010.*

[18] M. Gastineau. *SCSCP C Library - A C/C++ library for Symbolic Computation Software Composibility Protocol*. IMCCE, version 0.6.2 edition, May 2010.

[19] T. Granlund. GNU multiple precision arithmetic library 4.2.4, September 2008. `http://swox.com/gmp/`.

[20] J. Henrard. A survey of poisson series processors. *Celestial Mechanics and Dynamical Astronomy*, 45(1):245–253, Mar. 1988.

[21] P. K. Janert. *Gnuplot in Action: Understanding Data with Graphs*. Manning Publications, 1 edition, August 2009.

[22] W. H. Jefferys. A FORTRAN-based list processor for poisson series. *Celestial Mechanics*, 2:474–480, Dec. 1970.

[23] W. H. Jefferys. A precompiler for the formula manipulation system TRIGMAN. *Celestial Mechanics*, 6:117–124, Aug. 1972.

[24] J. Laskar. Accurate methods in general planetary theory. *Astronomy Astrophysics*, 144:133–146, Mar. 1985.

[25] J. Laskar. A numerical experiment on the chaotic behaviour of the solar system. *Nature*, 338:237, Mar. 1989.

[26] J. Laskar. The chaotic motion of the solar system - a numerical estimate of the size of the chaotic zones. *Icarus*, 88:266–291, Dec. 1990.

[27] J. Laskar. Manipulation des séries. In *Modern Methods in Celestial Mechanics, Comptes Rendus de la 13ieme Ecole Printemps d'Astrophysique de Goutelas (France), 24-29 Avril, 1989*, pages 63–88, Gif-sur-Yvette, 1990. Editions Frontieres.

[28] J. Laskar and P. Robutel. Stability of the planetary three-body problem. i. expansion of the planetary hamiltonian. *Celestial Mechanics and Dynamical Astronomy*, 62:193–217, July 1995.

[29] F. Malige, P. Robutel, and J. Laskar. Partial reduction in the N-Body planetary problem using the angular momentum integral. *Celestial Mechanics and Dynamical Astronomy*, 84(3):283–316, Nov. 2002.

[30] M. Monagan and R. Pearce. Parallel sparse polynomial multiplication using heaps. In *ISSAC '09: Proceedings of the 2009 international symposium on Symbolic and algebraic computation*, pages 263–270, New York, NY, USA, 2009. ACM.

[31] P. Robutel. Stability of the planetary three-body problem. ii. kam theory and existence of quasiperiodic motions. *Celestial Mechanics and Dynamical Astronomy*, 62:219–261, July 1995.

[32] A. Rom. Mechanized Algebraic Operations (MAO). *Celestial Mechanics*, 1:301–319, Sept. 1970.

[33] E. Stambulchik. *Grace User's Guide*, 5.1.22 edition, May 2008. `http://plasma-gate.weizmann.ac.il/Grace/`.

[34] The OpenMath Society. *The OpenMath Standard Version 2.0*, June 2004.