Maple Transactions

Dealing with the integer solutions of, possibly parametric, systems of linear equations and inequalities in Maple

YUZHUO LEI, The University of Western Ontario, Canada
RUI-JUAN JING, Jiangsu University, China
CHRISTOPHER MALIGEC, The University of Western Ontario, Canada
LIN-XIAO WANG, The University of Western Ontario, Canada

Recommended Reference Format:
Yuzhuo Lei, Rui-Juan Jing, Christopher Maligec, and Lin-Xiao Wang. 2024. Dealing with the integer solutions of, possibly parametric, systems of linear equations and inequalities in Maple. 1, 1 (May 2024), 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Overview

When solving systems of polynomial equations and inequalities, the task of computing their solutions with integer coordinates is a much harder problem than that of computing their real solutions or that of computing all their solutions. In fact, in the presence of non-linear constraints this task may simply become an undecidable problem. However, studying the integer solutions of systems of equations and inequalities is of practical importance in various areas of scientific computing. Two such areas are *combinatorial optimization* (in particular integer linear programming) and *compiler optimization* (in particular, the analysis, transformation and scheduling of for-loop nests in computer programs), where a variety of algorithms solve questions related to the points with integer coordinates belonging to a given polyhedron. Another area is at the crossroads of computer algebra and polyhedral geometry, with topics such as toric ideals and Hilbert bases, see [19] as well the manipulation of Laurent series, see [1].

One can ask different questions about the integer points of a polyhedral set, ranging from "whether or not a given rational polyhedron has integer points" to "describing all such points". Answers to that latter question can take various forms, depending on the targeted application. For plotting purposes, one may want to enumerate all the integer points of a 2D or 3D polytope. Meanwhile, in the context of combinatorial optimization or compiler optimization, more concise descriptions are sufficient and more effective. For a rational convex polyhedron $P \subseteq \mathbb{Q}^d$, defined either by the set of its facets or that of its vertices, one such description is the *integer hull* P_I of P, that is, the convex hull of $P \cap \mathbb{Z}^d$. The set P_I is itself polyhedral and can be described either by its facets, or its vertices.

Another concise description was proposed in [9, 8] where the integer points of a polyhedral set are represented by (finitely many) triangular systems with specialization properties similar to those of regular chains and lexicographical Gröbner bases.

An even more concise answer is given by counting the number of integer points of a polytope. This problem has efficient algorithmic solutions, in particular Barvinok's algorithm [2], as well as

Authors' addresses: Yuzhuo Lei, The University of Western Ontario, 1151 Richmond St, London, Canada, ylei83@uwo.ca; Rui-Juan Jing, Jiangsu University, Jingkou District, Zhenjiang, China, rjing@ujs.edu.cn; Christopher Maligec, The University of Western Ontario, 1151 Richmond St, London, Canada, cmaligec@uwo.ca; Lin-Xiao Wang, The University of Western Ontario, 1151 Richmond St, London, Canada, lwang739@uwo.ca.

https://doi.org/10.1145/nnnnnnn.nnnnnnn

numerous applications, such as counting the number of memory locations (or cache lines) accessed by a for-loop nest, see [15, 6].

Over the past 10 years, a series of projects has equipped the computer algebra system Maple with a number of tools for dealing with the integer points of systems of linear equations and inequalities, even in the presence of parameters. These tools implement novel and effective algorithms; they are part in the `PolyhedralSets` library and they are the main subject of this paper.

With these tools, one can either decide whether integer point solutions exist, or count them. One can also describe them in a compact way, or enumerate them, see Sections 2 and 3. Last but not least, one can deal with parametric polytopes and count the number of their integer points, see Section 4.

2 Computing integer hulls

One important family of algorithms for computing the vertex set of the integer hull P_I of the polyhedral set $P \subseteq \mathbb{Q}^d$, relies on the so-called *cutting plane method*, originally introduced by Gomory in [7] to solve integer linear programs. Chvátal [5] and Schrijver [17] developed a procedure to compute P_I based on that latter method. Schrijver gave a full proof and a complexity study of this method in [16]. Another approach for computing P_I uses the *branch and bound method*, introduced by Land and Doig in the early 1960s in [10]. This method recursively divides P into sub-polyhedra, then the vertices of the integer hull of each part of the partition are computed. Yet another approach for computing P_I is used in the software `Normaliz` [4]: by embedding P in dimension $d+1$, the problem of computing P_I is reduced to that of computing generators (with integer coordinates) of a rational polyhedral cone C; indeed, once those generators are known, Gordan's lemma produces a generating set of the affine monoid $C \cap \mathbb{Z}^{d+1}$ to which a convex hull algorithm is applied to finally deduce P_I; hence, the bulk of the work is to compute generators of C, which is achieved by Fourier-Motzkin Elimination (FME).

```
> with(PolyhedralSets) :
> ineqs := [2 x + 5 y ≤ 64, 7 x + 5 y ≥ 20 , 3 x − 6 y ≤ −7] :
> poly := PolyhedralSet(ineqs, [x, y]);
```

$$poly := \begin{cases} Coordinates & : & [x, y] \\ Relations & : & \left[-x - \frac{5y}{7} \le -\frac{20}{7}, x - 2y \le -\frac{7}{3}, x + \frac{5y}{2} \le 32\right] \end{cases} \quad (1)$$

```
> IntegerHull(poly);
```

$$[[[12, 8], [-8, 16], [-7, 14], [-5, 11], [0, 4], [1, 3], [3, 3], [11, 7]], [\,]] \quad (2)$$

```
> IntegerHull(poly, returntype = polyhedralset);
```

$$\begin{cases} Coordinates & : & [x, y] \\ Relations & : & \left[-y \le -3, -x - y \le -4, -x - \frac{5y}{7} \le -\frac{20}{7}, -x - \frac{2y}{3} \le -\frac{7}{3}, -x - \frac{y}{2} \le 0, x - 2y \le -3, x - y \le 4, x + \frac{5y}{2} \le 32\right] \end{cases}$$

```
>
```

Fig. 1. Using the `IntegerHull` command

In [13], the authors proposed a new algorithm for computing P_I, which is implemented as in the command `IntegerHull` of the `PolyhedralSets` library. This new algorithm has three main steps:

- *normalization*, during which P is replaced by a rational polyhedron $Q \subseteq \mathbb{Q}^d$ so that the supporting hyperplane of each facet of Q has integer points and $P_I = Q_I$ holds;
- *partitioning*, during which we search for integer points inside Q and use them to partition Q into "smaller polyhedral sets", the integer hulls of which can easily be computed;

- *merging*, during which the integer hulls of the parts of the partition are merged by means of the convex hull algorithm.

The efficiency of this algorithm depends mainly on the shape of the input while the size of the input has little impact. Comparative experimentation shows this new algorithm can deal with inputs of very large volumes that algorithms depending on enumeration, such as those of `Normaliz` can not process. Figure 1 illustrates the use of the `IntegerHull` command of the `PolyhedralSets` library.

3 Describing $\mathbb{Z}$-polyhedra

Following [18], we call *integer lattice* of $\mathbb{Z}^d$ any set of the form

$$\{A\mathbf{x} + \mathbf{b} \mid \mathbf{x} \in \mathbb{Z}^d\}$$

where $A \in \mathbb{Z}^{d \times d}$ is a full-rank matrix and $\mathbf{b} \in \mathbb{Z}^d$ is a vector; such a set is denoted by $\mathcal{L}(A, \mathbf{b})$.

Following [18] again, we call a $\mathbb{Z}$-*Polyhedron* the intersection of a polyhedron with an integer lattice. The purpose of this notion is, for us, to support the description of the integer points of a polyhedron $P \subseteq \mathbb{Q}^n$, that is, the description of the set $P \cap \mathbb{Z}^n$.

To understand why lattices appear naturally when solving for the integer points of a polyhedron, consider a couple of elementary examples. First, consider the problem of solving a Diophantine equation over $\mathbb{Z}$, say in 2 variables x and y. For instance, consider $3x - 4y = 7$; its solutions, as computed by Maple, are of the form

$$x = 5 + 4\,\_Z1, y = 2 + 3\,\_Z1,$$

the description of which requires the use of the auxiliary variable $\_Z1$.

In his Omega test [14, 15] William Pugh extended that idea for solving arbitrary systems of linear equations over $\mathbb{Z}$. For instance, for the system

$$\begin{cases} 7x + 12y + 31z &= 17 \\ 3x + 5y + 14z &= 7 \end{cases}$$

the Omega test produces

$$\begin{cases} z &= -t_0 - 1 \\ y &= -5t_0 - 3 \\ x &= 13t_0 + 12 \end{cases}$$

Of course, the introduction of the parameter t_0 can be avoided by re-writing x and z as a function of z, (simply by using Hermite normal forms) leading to:

$$\begin{cases} x &= -1 - 13z \\ y &= 2 + 5z \end{cases}$$

Consider now this other polyhedron P of $\mathbb{Q}^3$:

$$\begin{cases} x &= 19 \\ y &= 25 + (1/2)z \\ z &\leq 18 \\ -z &\leq 0 \end{cases}$$

Because of the presence of the rational number $1/2$, the above input system cannot be considered as a description of the set $P \cap \mathbb{Z}^3$ and the Omega test produces:

$$\begin{cases} x &= 19 \\ y &= 25 + t_0 \\ z &= 2\,t_0 \\ -t_0 &\leq 0 \\ t_0 &\leq 9 \end{cases}$$

Inspired by the work[18], and the Omega test, we have designed and implemented an algorithm [9, 8] for representing $\mathbb{Z}$-polyhedra. In particular, for the above example, we obtain in a MAPLE session, the result shown on Figure 2. On the left-hand side of Figure 2, we retrieve our original polyhedron

$$\left[\left\{\begin{array}{l} x = 19 \\ y = 25 + \frac{z}{2} \\ z \le 18 \\ -z \le 0 \end{array}\right\}, \left[\begin{array}{c} x \\ y \\ z \end{array}\right], \left[\left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{array}\right], \left[\begin{array}{c} 0 \\ 0 \\ 0 \end{array}\right]\right]\right]$$

Fig. 2. A $\mathbb{Z}$-Polyhedron in PL format.

P and on the right-hand side, we have the lattice $\mathcal{L}$ of $\mathbb{Z}^n$ consisting of the points (x, y, z) where $z/2$ is integer. The intersection $P \cap \mathcal{L}$ is exactly $P \cap \mathbb{Z}^3$. More generally, encoding the integer points of a polyhedron using the above format, that we call the *PL format*, and thus using lattices, allows us to totally avoid the recourse to auxiliary variables. In addition, it is easy to convert any set of the form $P \cap \mathbb{Z}^n$ (where $P \subseteq \mathbb{Q}^n$ is a polyhedron) from PL format, say $P \cap \mathcal{L}(C, \dot{})$, to the Omega test format, simply by substitution.

The subpackage `ZPolyhedralSets` of the `PolyhedralSets` library is dedicated to the manipulation of $\mathbb{Z}$-Polyhedra. This subpackage offers a number of functionalities, among them:

- `EnumerateIntegerPoints` for enumerating the points of a $\mathbb{Z}$-Polytope,
- `PlotIntegerPoints3d` for plotting 3D $\mathbb{Z}$-Polyhedra,
- `IsContained` to test inclusion between $\mathbb{Z}$-Polyhedra.

The core operation is `IntegerPointDecomposition` which, given an input $\mathbb{Z}$-Polyhedron K computes a finite partition of K into $\mathbb{Z}$-Polyhedra $K_1, \ldots, K_e$ with additional properties, see [9, 8]. In particular, none of the $\mathbb{Z}$-Polyhedra $K_1, \ldots, K_e$ is empty. Under genericity assumptions, `IntegerPointDecomposition` runs in single exponential time (w.r.t. the number of variables); its worst case complexity is doubly exponential.

We conclude this section with an example of a decomposition computed by `IntegerPointDecomposition`. Since for each of the involved $\mathbb{Z}$-Polyhedra, except the last one, the lattice part is simply the entire $\mathbb{Z}^3$, we do not print it. For the last $\mathbb{Z}$-Polyhedron, we use the Omega test format.

For $K : \left\{\begin{array}{rcl} 3x_1 - 2x_2 + x_3 & \le & 7 \\ -2x_1 + 2x_2 - x_3 & \le & 12 \\ -4x_1 + x_2 + 3x_3 & \le & 15 \\ -x_2 & \le & -25 \end{array}\right.$, with $x_1 > x_2 > x_3$, the output consists of 5 $\mathbb{Z}$-polyhedra K_1, K_2, K_3, K_4, K_5:

$$\left\{\begin{array}{rcl} 3x_1 - 2x_2 + x_3 & \le & 7 \\ -2x_1 + 2x_2 - x_3 & \le & 12 \\ -4x_1 + x_2 + 3x_3 & \le & 15 \\ 2x_2 - x_3 & \le & 48 \\ -5x_2 + 13x_3 & \le & 67 \\ -x_2 & \le & -25 \\ 2 \le x_3 & \le & 17 \end{array}\right., \left\{\begin{array}{rcl} x_1 & = & 15 \\ x_2 & = & 27 \\ x_3 & = & 16 \end{array}\right., \left\{\begin{array}{rcl} x_1 & = & 18 \\ x_2 & = & 33 \\ x_3 & = & 18 \end{array}\right., \left\{\begin{array}{rcl} x_1 & = & 14 \\ x_2 & = & 25 \\ x_3 & = & 15 \end{array}\right., \left\{\begin{array}{rcl} x_1 & = & 19 \\ x_2 & = & 50 + t \\ x_3 & = & 50 + 2t \\ -25 & \le & t \le -16. \end{array}\right.$$

To use the terminology of William Pugh, the $\mathbb{Z}$-polyhedron K_1 corresponds to the dark shadow of the input polyhedron K, while K_2, K_3, K_4, K_5 correspond to its grey shadow.

4 Dealing with parameters

In practice, polyhedral sets are often *parametric*. Consider for instance the for-loop nest, written in a programming language (say C) of a dense matrix multiplication algorithm. At compile time, the upper bound of the value range of each loop counter is a symbol. To be more precise, the iterations of that for-loop nest are the integer points of a polyhedral set P given by a system of linear inequalities $A\vec{x} \leq \vec{b}$ where A is a matrix with integer coefficients, $\vec{b}$ is a vector of symbols (actually the parameters of the polyhedral set) and $\vec{x}$ is the vector of the loop counters. At execution time, different values of $\vec{b}$ yield different shapes and numbers of vertices for P_I. So what can be done at compile time to answer a question like what is the number of memory accesses performed by that for-loop nest?

For non-parametric polyhedral sets, the most studied method for counting the integer points of a polytope is certainly due to Alexander Barvinok in [2]. The Authors of [11] and [21] report on an implementation of Barvinok's algorithm respectively in the software libraries LattE and barvinok, both developed in C/C++. Moreover, the latter library has an adaptation of Barvinok's algorithm to parametric polyhedral sets.

The `PolyhedralSets` also offers an implementation of Barvinok's algorithm adapted to count the integer points of parametric polyhedral sets. This led us to implement various Maple packages which are interesting in their own right.

At the core of Barvinok's algorithm, one finds the computation of the generating function of a polytope. In the parametric case, such a function is a multivariate rational function where some exponents of some monomials depend on the parameters. To manipulate those expressions in a convenient way, we have implemented quasi-polynomials in Maple. A *quasi-polynomial* is a polynomial-like object, the coefficients of which are instead periodic functions with integral period.

A second challenge in adapting Barvinok's algorithm from non-parametric to parametric polytopes is due to the complexity of the case discussion. To make that observation clear, let us first recall how one solves a parametric linear system, say by Gaussian elimination: the search for pivots, one after another, dynamically unfolds a *decision tree* where:

- internal nodes are candidate pivots (with a left child for the "non-zero" case and a right child for the "zero" case), and
- leaves are solutions.

Let us now return to the integer point counting of a parametric polytope $\mathcal{P}(\mathbf{b})$. One must first determine the vertices of $\mathcal{P}(\mathbf{b})$. This leads to a number of parametric problems which can be solved independently, thus leading to a number of decision trees. Once all these decision trees are computed, they must be merged into a single decision tree. Next, the vertex cone of each vertex must be determined. This again creates a number of decision trees, which must be merged into a single decision tree.

At this point, one is essentially ready to apply Barvinok's algorithm, up to the issue solved by the usage of the quasi-polynomials. After doing so, one usually observes a number of cases which are specializations of the same more general case. We are in the process of implementing an algorithm to detect such a pattern and replace those specializations by a single more general case.

```
> relations := [1 ≤ i,j ≤ n, i ≤ m, 3*i ≤ 5*j];
  vars := [i,j];
  paras := [m,n];
```

$$relations := [1 \le i,j \le n, i \le m, 3\,i \le 5\,j]$$

$$vars := [i,j]$$

$$paras := [m,n]$$

```
> test_number_of_points := NumberOfIntegerPoints(relations, vars, paras, false, true);
```

$$test\_number\_of\_points := \left[\left[\left\{\frac{3\,m}{10} - \frac{3\,m^2}{10} + Q\left(\left[m, 5, \left[0, 0, -\frac{2}{5}, -\frac{1}{5}, -\frac{2}{5}\right]\right]\right) + n\,m\right\}, [0 \le m-2, 0 \le 5\,n-7, 0 \le -3\,m+5\,n-1]\right], [\{n\}, [m-1=0, 0 \le 5\,n-4]], \left[\left\{\frac{5\,n^2}{6} + \frac{n}{2} + Q\left(\left[n, 3, \left[0, -\frac{1}{3}, -\frac{1}{3}\right]\right]\right)\right\}, [3\,m-5\,n=0, 0 \le 5\,n-4]\right], \left[\left\{\frac{5\,n^2}{6} + \frac{n}{2} + Q\left(\left[n, 3, \left[0, -\frac{1}{3}, -\frac{1}{3}\right]\right]\right)\right\}, [0 \le 5\,n-4, 0 \le 3\,m-5\,n-1]\right]\right]$$

Fig. 3. Using the `NumberOfIntegerPoints` command

Figure 3 shows the computation of the number of integer points of the parametric polytope given by

$$\begin{cases} 1 & \le & i \\ i & \le & m \\ j & \le & n \\ 3i & \le & 5j \end{cases} \tag{1}$$

in the coordinates (i, j) and with m, n as parameters. The output of the command `NumberOfIntegerPoints` consists 0f 4 pairs *value-constraints*. For the first, the third and the fourth pairs, the value part uses a quasi-polynomial. For instance, the value part of the first pair is:

$$\frac{3m}{10} - \frac{3m^2}{10} + Q([m, 5, [0, 0, -2/5, -1/5, -2/5]]) + nm,$$

which uses the quasi-polynomial

$$Q([m, 5, [0, 0, -2/5, -1/5, -2/5]]),$$

which must be read as follows:

$$\begin{cases} 0 & \text{if} & m \equiv 0 & \mod 5 \\ 0 & \text{if} & m \equiv 1 & \mod 5 \\ -2/5 & \text{if} & m \equiv 2 & \mod 5 \\ -1/5 & \text{if} & m \equiv 3 & \mod 5 \\ -2/5 & \text{if} & m \equiv 4 & \mod 5 \end{cases}$$

We observe that the third and fourth pairs [value-constraints] shown on Figure 3 can be replaced by a single pair [value-constraints], namely:

$$[\{\frac{5n^2}{6} + n/2 + Q([n, 3, [0, -1/3, -1/3]])\}, [3m - 5n = 0, 0 \leq 5n - 4]].$$

As mentioned above, we are in the process of implementing this type of recombination of cases.

References

[1] Ainhoa Aparicio Monforte and Manuel Kauers. Formal laurent series in several variables. *Expositiones Mathematicae*, 31(4):350–367, 2013.

[2] Alexander I Barvinok. A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. *Mathematics of Operations Research*, 19(4):769–779, 1994.

[3] Alberto Bemporad, Komei Fukuda, and Fabio D. Torrisi. Convexity recognition of the union of polyhedra. *Computational Geometry*, 18(3):141–154, 2001.

[4] Winfried Bruns, Bogdan Ichim, and Christof Söger. Introduction to normaliz 2.5. In Komei Fukuda, Joris van der Hoeven, Michael Joswig, and Nobuki Takayama, editors, *Mathematical Software - ICMS 2010, Third International Congress on Mathematical Software, Kobe, Japan, September 13-17, 2010. Proceedings*, volume 6327 of *Lecture Notes in Computer Science*, pages 209–212. Springer, 2010.

[5] Vasek Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. *Discret. Math.*, 4(4):305–337, 1973.

[6] Philippe Clauss, Federico Javier Fernández, Diego Garbervetsky, and Sven Verdoolaege. Symbolic polynomial maximization over convex sets and its application to memory requirement estimation. *IEEE Trans. Very Large Scale Integr. Syst.*, 17(8):983–996, 2009.

[7] Ralph E. Gomory. Outline of an algorithm for integer solutions to linear programs *and* an algorithm for the mixed integer problem. In Michael Jünger, Thomas M. Liebling, Denis Naddef, George L. Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi, and Laurence A. Wolsey, editors, *50 Years of Integer Programming 1958-2008 - From the Early Years to the State-of-the-Art*, pages 77–103. Springer, 2010.

[8] Rui-Juan Jing and Marc Moreno Maza. Computing the integer points of a polyhedron, II: complexity estimates. In Vladimir P. Gerdt, Wolfram Koepf, Werner M. Seiler, and Evgenii V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing - 19th International Workshop, CASC 2017, Beijing, China, September 18-22, 2017, Proceedings*, volume 10490 of *Lecture Notes in Computer Science*, pages 242–256. Springer, 2017.

[9] Rui-Juan Jing and Marc Moreno Maza. Computing the integer points of a polyhedron, I: algorithm. In *CASC 2017, Proceedings*, volume 10490 of *LNCS*, pages 225–241. Springer, 2017.

[10] A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica: Journal of the Econometric Society*, pages 497–520, 1960.

[11] Jesús A. De Loera, Raymond Hemmecke, Jeremiah Tauzer, and Ruriko Yoshida. Effective lattice point counting in rational convex polytopes. *J. Symb. Comput.*, 38(4):1273–1302, 2004.

[12] Marc Moreno Maza and Linxiao Wang. On the pseudo-periodicity of the integer hull of parametric convex polygons. In *CASC 2021, Proceedings*, volume 12865 of *LNCS*, pages 252–271. Springer, 2021.

[13] Marc Moreno Maza and Linxiao Wang. Computing the integer hull of convex polyhedral sets. In François Boulier, Matthew England, Timur M. Sadykov, and Evgenii V. Vorozhtsov, editors, *CASC 2022, Proceedings*, volume 13366 of *Lecture Notes in Computer Science*, pages 246–267. Springer, 2022.

[14] William W. Pugh. The omega test: a fast and practical integer programming algorithm for dependence analysis. In *Proceedings Supercomputing '91, Albuquerque, NM, USA, November 18-22, 1991*, pages 4–13. ACM, 1991.

[15] William W. Pugh. Counting solutions to presburger formulas: How and why. In *Proceedings of the ACM SIGPLAN'94 Conference on Programming Language Design and Implementation (PLDI), Orlando, Florida, USA, June 20-24, 1994*, pages 121–134. ACM, 1994.

[16] Arvind Rajan. Theory of linear and integer programming, by alexander schrijver, wiley, new york, 1986, 471 pp. price $71.95. *Networks*, 20(6):801, 1990.

[17] Alexander Schrijver. On cutting planes. *Combinatorics*, 79:291–296, 1980.

[18] Rachid Seghir, Vincent Loechner, and Benoit Meister. Integer affine transformations of parametric $\mathbb{Z}$-polytopes and applications to loop nest optimization. *ACM Trans. Archit. Code Optim.*, 9(2):8:1–8:27, 2012.

[19] Rekha R. Thomas. Integer programming: Algebraic methods. In Christodoulos A. Floudas and Panos M. Pardalos, editors, *Encyclopedia of Optimization, Second Edition*, pages 1624–1634. Springer, 2009.

[20] Sven Verdoolaege. Integer set coalescing. In *International Workshop on Polyhedral Compilation Techniques, Date: 2015/01/19-2015/01/19, Location: Amsterdam, The Netherlands*, 2015.

[21] Sven Verdoolaege, Rachid Seghir, Kristof Beyls, Vincent Loechner, and Maurice Bruynooghe. Counting integer points in parametric polytopes using barvinok's rational functions. *Algorithmica*, 48(1):37–66, 2007.