Intersecting Quadrics: An Efficient and Exact Implementation

Sylvain Lazard* Luis Mariano Peñaranda† Sylvain Petitjean‡

ABSTRACT
We present the first complete, exact and efficient C++ implementation of a method for parameterizing the intersection of two implicit quadrics with integer coefficients of arbitrary size. It is based on the near-optimal algorithm recently introduced by Dupont et al. [2].

Unlike existing implementations, it correctly identifies and parameterizes all the connected components of the intersection in all the possible cases, returning parameterizations with rational functions whenever such parameterizations exist. In addition, the coefficient field of the parameterizations is either minimal or involves one possibly unneeded square root.

1. INTRODUCTION
Computing an explicit representation of the intersection of two general quadrics is a fundamental problem in many areas like solid modeling, computational geometry and computer graphics. The range of applications covers well-known problems like sweeping arrangements [12, 15], boundary evaluation [14] and convex hull computation [5].

Past work. Until recently, the only known general method for computing a parametric representation of the intersection between two arbitrary quadrics was due to J. Levin [7]. It is based on an analysis of the pencil generated by the two quadrics, i.e. the set of linear combinations of the two quadrics.

Though useful for curve tracing, Levin’s method has serious limitations. When the intersection is singular or reducible, a parameterization by rational functions is known to exist, but the pencil method fails to find it and generates a parameterization that involves the square root of some polynomial. In addition, since it introduces algebraic numbers of very high degree (for instance in the computation of eigenvalues and eigenvectors), a correct implementation using exact arithmetic is essentially out of reach. And when a floating point representation of numbers is used, the method may output results that are wrong (geometrically and topologically) and it may even fail to produce any parameterization at all and crash.

Over the years, Levin’s seminal work has been extended and refined in several different directions. Wilf and Manor [21] use a classification of quadric intersections by the Segre characteristic (see [1]) to drive the parameterization of the intersection by the pencil method. Recently, Wang, Goldman and Tu [19] further improved the method by making it capable of computing structural information on the intersection and its various connected components and able to produce a parameterization by rational functions when it exists. Whether the refined algorithm is numerically robust is open to question.

Another method of algebraic flavor was introduced by Farouki, Neff and O’Connor [3] when the intersection is degenerate. In such cases, using a combination of classical concepts (Segre characteristic) and algebraic tools (factorization of multivariate polynomials), the authors show that explicit information on the morphological type of the intersection curve can be reliably obtained. A notable feature of this method is that it can output an exact parameterization of the intersection in simple cases, when the input quadrics have rational coefficients. No implementation is however reported.

Rather than restricting the type of the intersection, others have sought to restrict the type of the input quadrics, taking advantage of the fact that geometric insights can then help compute the intersection curve [11, 16]. Specialized routines are devised to compute the intersection curve in each particular case. Even though such geometric approaches are numerically more stable than the algebraic ones, they are essentially limited to the class of so-called natural quadrics, i.e., the planes, right cones, circular cylinders and spheres.

Apart from [2], perhaps the most interesting of the known algorithms for computing an explicit representation of the intersection of two arbitrary quadrics is the method of Wang, Joe and Goldman [20]. This algebraic method is based on a birational mapping between the intersection curve and a plane cubic curve. The cubic curve is obtained by projection from a point lying on the intersection. Then the classification and parameterization of the intersection are obtained by invoking classical results on plane cubics. The authors claim that their algorithm is the first to produce a complete topological classification of the intersection (singularities, number and types of connected components, etc.). Numerical robustness issues have however not been studied and the intersection may not be correctly classified. Also, the center of projection is currently computed using Levin’s (enhanced) method: with floating point arithmetic, it will in general not exactly lie on the curve, which is another source of numerical instability.

Contributions. In this paper, we present the first exact, robust, efficient and usable implementation of an algorithm for parameterizing the intersection of two arbitrary implicit quadrics. It is based on the new parameterization method described in [2].

More precisely, our implementation has the following features:
• it computes an exact parameterization of the intersection of two quadrics with integer coefficients of arbitrary size;
it correctly identifies, separates and parameterizes all the connected components of the intersection and gives all the relevant topological information;

it places no restriction of any kind on the type of the intersection or the type of the input quadrics;

the parameterizations are rational when the intersection is singular and involve the square root of a polynomial when the intersection is a smooth quartic;

the parameterizations are either optimal in the degree of the extension of \( \mathbb{Z} \) on which their coefficients are defined or, in a small number of well-identified cases, involve one extra possibly unnecessary square root;

it is carefully designed so that the size of the coefficients is kept small;

it is fast and efficient and can routinely compute parameterizations of the intersection of quadrics with input coefficients having ten digits in less than 50 milliseconds on a mainstream PC.

Our C++ implementation can be queried via a web interface at the following URL: http://www.loria.fr/isa/qi. Its code will be released to educational and research audiences shortly.

Paper outline. The paper is organized as follows. In Section 2, we recall the main ideas of our parameterization algorithm and describe its implementation. In Section 3, we focus on two cases covering the two main parameterization philosophies used in our implementation, prove theoretical bounds on the size of the output coefficients and compare those bounds to observed values. We give more experimental results and performance evaluation in Section 4. Finally, we give some examples of running our algorithm in Section 5, before concluding.

2. ALGORITHM AND IMPLEMENTATION

In this section, we give a brief presentation of the basic ideas underpinning our near-optimal parameterization method [2]. We then move on to a description of the main design choices we made to implement it.

2.1 Preliminaries

In what follows, all the matrices considered are \((n + 1) \times (n + 1)\) real square matrices, \(n = 2\) or \(3\). We call quadratic associated with a matrix \(S\) the quadratic hypersurface defined as

\[ Q_S = \{ x \in \mathbb{P}^n \mid x^T S x = 0 \}, \]

where \(\mathbb{P}^n = \mathbb{P}^n(\mathbb{R})\) denotes the real projective space of dimension \(n\). In the rest of this paper, points and parameterizations are assumed to live in projective space, i.e. a point of \(\mathbb{P}^3\) has four coordinates.

We define the inertia of \(S\) and \(Q_S\) as the pair

\[ \sigma_S = (\max(\sigma^+, \sigma^-), \min(\sigma^+, \sigma^-)), \]

where \(\sigma^+\) (resp. \(\sigma^-\)) is the number of positive (resp. negative) eigenvalues of \(S\). Recall that Sylvester’s Inertia Law asserts that the inertia of \(S\) is invariant by a real projective transformation [6].

The rank of \(S\) is the sum \(\sigma^+ + \sigma^-\). Assume now that \(n = 3\). The smooth quadrics of \(\mathbb{P}^3\) are the quadrics of rank 4. We call cones the quadrics of rank 3 and real cones the cones of inertia \((2, 1)\). Similarly, we call pairs of planes the quadrics of rank 2 and real pairs of planes the planes of inertia \((1, 1)\). All the quadric surfaces except those of inertia \((3, 1)\) are ruled surfaces.

Given two matrices \(S\) and \(T\), let \(R(\lambda, \mu) = \lambda S + \mu T\). The set

\[ \{ R(\lambda, \mu) \mid (\lambda, \mu) \in \mathbb{P}^1 \} \]

calls the pencil of matrices generated by \(S\) and \(T\). Associated to it is a pencil of quadrics \(\{ Q_{R(\lambda,\mu)} \mid (\lambda,\mu) \in \mathbb{P}^1 \}\).

For the sake of simplicity, we sometimes write

\[ R(\lambda) = \lambda S - T, \quad \lambda \in \mathbb{R} \cup \{ \infty \}. \]

Recall that the intersection of two distinct quadrics of a pencil is independent of the choice of the two quadrics.

The binary quartic form \(\det R(\lambda,\mu)\) is called the determinant equation of the pencil. The quadrics of the pencil of rank less than or equal to 3 are exactly those quadrics \(R(\lambda,\mu)\) such that \(\det R(\lambda,\mu) = 0\). The binary cubic form \(\det R_\lambda(\lambda,\mu)\), where \(R_\lambda(\lambda,\mu)\) is the upper left 3 \times 3 submatrix of \(R(\lambda,\mu)\), is called the principal subdeterminantal equation of the pencil.

2.2 Near-optimal parameterization algorithm

2.2.1 Main ideas

Let \(\{ Q_{R(\lambda,\mu)} \mid (\lambda,\mu) \in \mathbb{P}^1 \}\), with \(R(\lambda,\mu) = \lambda S + \mu T\), be a pencil of quadrics. The main idea of methods for parameterizing the intersection of two quadrics based on an analysis of their pencil (Levin’s and derivatives) is as follows: find a quadric \(Q_R\) of some particularly simple form in the pencil generated by \(Q_S\) and \(Q_T\) (assume \(Q_R \neq Q_S\)), parameterize it, plug this parameterization \(X\) in the equation of \(Q_S\), solve the resulting equation \(X^T S X = 0\), and plug the result in \(X\), giving the parameterization of the intersection.

The key to make this procedure work in practice is to find a quadric \(Q_R\) having a parameterization that is linear in one of its parameters (i.e. is ruled), so that the equation \(X^T S X = 0\) has degree 2. Levin’s main result was to prove that a pencil of quadrics always contains at least one “simple” ruled quadric [7]. However, since such quadrics are found by first finding the zeros of the principal subdeterminantal equation and since cubic equations have generically no rational root (by Hilbert’s Irreducibility Theorem), we see that Levin’s algorithm introduces non-rational numbers at an early stage and floating-point arithmetic has to be used, resulting in numerical robustness issues.

The principal contribution of [2] was to show that, by a careful choice of the intermediate quadric \(Q_R\), the appearance of algebraic numbers can be kept to a minimum. One major result is encapsulated in Theorem 3 of [2]: except when the intersection is reduced to two real points, the pencil contains at least one ruled quadric whose coefficients are rational. In addition, thanks to new and worst-case optimal (in the number of square roots) parameterizations of ruled projective quadrics, we can always find such a rational ruled quadric \(Q_R\) with a parameterization involving only one square root.

Some of the basic ingredients used in our algorithm or to infer information about the intersection are the Segre classification of pencils and its refinement over the reals (the
Canonical Form Theorem for pairs of real symmetric matrices – see [18]), a projective setting, ad hoc projective transformations to compute the canonical form of a projective quadric, and Sylvester’s Inertia Law.

The basic principles underlying the design of our implementation are as follows:

- compute structural information on the intersection and its various real components as early as possible;
- use the structural information gathered to drive the parameterization process and make the right choices so that the output is optimal or near-optimal from the point of view of the degree of the extension of \( Z \) on which its coefficients are defined.

In our implementation, we were interested not just in optimizing the number of square roots in the output but also in minimizing the size of the output coefficients. For this reason, the basic philosophy is to use as intermediate ruled quadric \( Q_R \) a quadric with rational coefficients of the smallest rank that we can easily find, the rationale being, for instance, that the parameterization of a cone involves coefficients of smallest asymptotic size than the parameterization of a quadric of inertia \((2, 2)\). There are essentially two cases: (i) \( Q_R \) has rank 4; (ii) \( Q_R \) has rank 3 or less.

### 2.2.2 \( Q_R \) has rank 4

The main case where \( Q_R \) has rank 4 is the smooth quartic case (we leave the other cases aside). In that situation, the quartic determinantal equation \( \det R(\lambda) \) has no multiple root. It could well be that at least one of its simple roots is rational and that a \( Q_R \) with rank less than 4 could have been used, but checking so with the Rational Root Theorem can be very time consuming. Since generically a degree-four equation has no rational root, we prefer instead to isolate the real zeros of the determinantal equation using an implementation of Uspekovsky’s algorithm [13]. We then take (at most two) rational test points \( \lambda_i \) outside the isolating intervals in the areas where \( \det R(\lambda) > 0 \). If one of the quadrics \( R(\lambda_i) \) has inertia \((4, 0)\), the intersection is empty (it is a complex smooth quartic), a consequence of Finsler’s Theorem (see [2]). Otherwise, we proceed.

We now have a quadric \( R_0 = R(\lambda_0) \) of inertia \((2, 2)\) and a range of values \( I = [a, b] \) such that \( \lambda_0 \in I \) and \( \det R(\lambda) > 0 \) for all \( \lambda \in I \). In the worst case, the parameterization of \( Q_R \) involves two square roots [2]. We can improve this situation as follows. First, compute a point \( p_0 \) on \( Q_{R_0} \). Approximate this point by a point \( p \) with integer coordinates. Find the quadric \( Q_R = Q_{R(\lambda_i)} \) through \( p \). If \( p \) is close enough to \( p_0 \), then \( \lambda_i \in I \) and \( \det R > 0 \). We thus have a quadric of inertia \((2, 2)\) containing a point in \( \mathbb{P}^3(Z) \): such a quadric can be parameterized with at most one square root [2].

Plugging the parameterization \( X((u, v), (s, t)) \) of \( Q_{R_0} \), with \((u, v), (s, t) \in \mathbb{P}^1 \), in the equation of any other quadric of the pencil gives a bihomogeneous equation that has degree two in \((u, v)\) and two in \((s, t)\). Solving this equation for \((s, t)\) in terms of \((u, v)\) and replugging in the parameterization of \( Q_R \) gives a parameterization of the smooth quartic:

\[
X(u, v) = X_1(u, v) \pm X_2(u, v) \sqrt{\Delta(u, v)},
\]

where \( X_1(u, v) \) (resp. \( X_2(u, v) \)) is a vector of homogeneous polynomials of degree 3 (resp. 1) and \( \Delta(u, v) \) is a homogeneous polynomial of degree 4.

If \( \delta = \det R \) is a square, then all of these polynomials have rational coefficients and the parameterization is optimal in terms of the degree of the extension of \( Z \) on which it is defined. If \( \delta \) is not a square, then we can only conclude that the parameterization is near-optimal: it might well be that there exists another quadric \( Q_{R'} \) of inertia \((2, 2)\) in the pencil such that \( \det R' \) is a square, implying that \( \sqrt{\delta} \) could have been avoided in the output (see Section 5.2 for an example). Finding such a quadric however implies, in general, finding a rational point on a hyperelliptic curve, a problem known to be very hard [2].

### 2.2.3 \( Q_R \) has rank < 4

Though not generic, the situation where \( Q_R \) has rank < 4 happens quite often in practice since it covers all the types of intersection corresponding in the Segre characterization to the determinantal equation having a single multiple root \( \lambda_0 \). Indeed, in that case, the multiple root is both real (otherwise its complex conjugate would also be a multiple root of \( \det R(\lambda) \)) and rational (otherwise its algebraic conjugate would also be a multiple root of \( \det R(\lambda) \)). So the associated quadric \( Q_R = Q_{R(\lambda_0)} \) has rational coefficients and has rank 3 or less.

The general philosophy for parameterizing the intersection is to parameterize \( Q_R \), plug the parameterization in any other quadric of the pencil, and solve the resulting equation in the parameters. There are however many situations in which this procedure can be simplified by the fact that we can find a rational point on \( Q_R \) and thus parameterize \( Q_R \) rationally, and that we know enough information on the intersection to greatly simplify the solving and factorization of the equation in the parameters.

Let us illustrate this on the example of an intersection made of a cubic and a line that are tangent. The determinantal equation in this case has a quadruple root corresponding to a real cone \( Q_R \). By the above argument, \( Q_R \) has rational coefficients. So the vertex \( c \) of \( Q_R \) has rational coordinates. \( c \) is the point of tangency of the cubic and the line of the intersection.

Assume \( Q_R \neq Q_S \). The line of the intersection is necessarily rational (otherwise its conjugate would be in the intersection). It can be found by intersecting the cone \( Q_R \) with the plane tangent to \( Q_S \) at \( c \). Picking any point with rational coordinates on this line other than \( c \) gives a rational point on the cone. A projective cone with a rational point on it outside its singular locus can be rationally parameterized. Plugging this parameterization in \( Q_S \) gives an equation in the parameters of the cone which can rationally be solved.

The cubic factor of this equation gives the cubic component of the intersection.

### 2.3 Implementation

Our implementation builds upon the LiDIA [9] and GMP [4] C/C++ libraries. LiDIA was originally developed for computational number theory purposes, but includes many types of simple parameterized and template classes that are useful for our application. Apart from simple linear algebra routines and algebraic operations on univariate polynomials, we
use its number theory package and its ability to manipulate vectors of polynomials, polynomials having other polynomials as coefficients, ... On top of it, we have added our own data structures. We have compiled LiDIA so that it uses GMP as multiprecision integer arithmetic. Other bignum packages can be used in a transparent way since the kernel is separated from the application programs by an interface in which the declarations and functions dealing with multiprecision integers (which LiDIA calls `bigint`) are standardized.

Our implementation consists of more than 17,000 lines of source code, which is essentially divided in the following chapters:

- **data structures**: structures for intersections of quadrics, for components of the intersection, for homogeneous polynomials with `bigint` coefficients (coordinates of components), for homogeneous polynomials with `bigint` polynomials as coefficients, and basic operations on these structures...

- **elementary operations**: computing the inertia of a quadric of `bigint`s, the coefficients of the determinantal equation, the gcd of the derivatives of the determinantal equation, the adjoint of a matrix, the singular space of a quadric, the intersection between two linear spaces, applying Descartes’s Sign Rule, the Gauss decomposition of a quadratic form into a sum of squares, isolating the roots of a univariate polynomial using Uspensky’s method, ...

- **number theory and optimizations**: gcd optimizations of the `bigint` coefficients of a polynomial, a vector or a matrix, optimizations of the coefficients of pairs and triples of vectors, optimizations of the columns of a matrix under constraint (on a given row, the entry in the first column times the entry in the second equals the entry in the third, ...), reparameterization of lines so that its representative points have small height, ... 

- **global parameterization procedures**: parameterization of a quadric of inertia (2, 2) with `bigint` coefficients going through a rational point, of a real cone (resp. conic), of a real cone (resp. conic) with a rational point, of a real pair of planes, of a real pair of lines, ... 

- **case-by-case pencil analysis**: dedicated procedures to handle the cases induced by the Segre characterization (no multiple root, one multiple root, two double roots, vanishing determinantal equation, ...) and all the subcases corresponding to various types of intersection, ...

- **printing and debugging**: turning on debugging information with the `DEBUG` preprocessor directive, checking whether the computed parameterizations are correct, pretty printing the parameterizations, ....

3. HEIGHT OF OUTPUT COEFFICIENTS

In this section, we prove theoretical bounds on the height of the coefficients of the parameterizations output by our intersection software. We do this for the two cases already outlined in the previous section: smooth quartic and cubic and tangent line.

The smooth quartic case is important because it is the generic intersection situation (given two random quadrics, the intersection is a smooth quartic with probability 1) and because it is also the worst case from the point of view of the height of the coefficients involved. The cubic and tangent line case is used to validate a key design choice we made, which is to take the quadric with rational coefficients of lowest possible rank to parameterize the intersection. We compare what happens for this type of intersection when we use for $QR$ a quadric of inertia (2, 2) or a quadric of inertia (2, 1).

Assume the input matrices are $S = (s_{ij})$ and $T = (t_{ij})$. In what follows, the size of the input coefficients is defined as

$$s = \log_{10} (\max (|s_{ij}|, |t_{ij}|)).$$

The height of the coefficients of some quantity $E$ in terms of the coefficients of $S$ and $T$ is defined as

$$h = \log_2 |e|, \quad e = \max \text{ of coeffs of } E.$$

### 3.1 The smooth quartic case

Let $QR$ be the quadric of inertia (2, 2) used to parameterized the intersection and $p$ a point of $\mathbb{P}^4(\mathbb{Z})$ on $QR$, as described in Section 2.2.2. We prove the following result.

**Proposition 3.1.** The parameterization of a smooth quartic

$$X(u, v) = X_1(u, v) \pm X_2(u, v) \sqrt{\Delta(u, v)}$$

is such that the height in terms of the coordinates of $S$ and $T$ is:

- 27 for the coefficients of the polynomials of $X_1$,
- 8 for the coefficients of the polynomials of $X_2$,
- 38 for the coefficients of $\Delta(u, v)$.

**Proof.** Let $M_0$ be the projective transformation sending the point $p$ to the point $p_0 = (1, 0, 0, 0)^T$. Let $R_0$ denote the quadric obtained from $R$ through the projective transformation $M_0$: $R_0 = M_0^T R M_0$. It follows from Sylvester’s Inertia Law that $R_0$ has the same inertia as $R$, i.e. (2, 2). Moreover, the point $p_0$ belongs to $Q_{R_0}$ since $M_0 p_0 = p$.

Note that the height of $R_0$ in terms of the coefficients of $S$ and $T$ is the same as the height of $R$. For clarity, we rename $R_0$ and $p_0$ by $R$ and $p$, respectively.

Let $x$ denote the vector $(x, y, z, w)^T$.

Let $L$ be half the differential of quadric $QR$ at $p$ (one can trivially show that the matrix of $L$ is the first row of $R$) and let $i$ be such that $R[1,i] \neq 0$ (such an $i$ necessarily exists). We compute the Euclidean division of $Q_R = x^T Rx$ by $Lx$ with respect to the variable $x[i]$. One can prove that it is equal to

$$R[1,i]^2 (x^T Rx) = (Lx) (L^T x) + A,$$

where

$$L^T[\xi] = -R[i,i] R[1,\xi] + 2 R[1,i] R[i,\xi]$$

for $\xi = 1, \ldots, 4$ and

$$A = c[j] x[j]^2 + c[k] x[k]^2 + 2 c[jk] x[j] x[k]$$

where $j$ and $k$ are equal to the two values in $\{2, 3, 4\}$ distinct from $i$, and

$$c[\xi] = R[\xi,\xi] R[1,1]^2 + R[\xi,1] R[1,1]^2 - 2 R[\xi,1] R[1,\xi], \quad \xi \in \{j, k\},$$

We assume in the following that \( c[j] \neq 0 \) (if \( c[j] = 0 \) but \( c[k] \neq 0 \), we exchange the roles of \( j \) and \( k \); otherwise the analysis is different but similar and we omit it here). For clarity we denote in the following \( c = c[j] \) and \( r = R[1, i] \).

We consider the projective transformation \( M \) such that, in the new projective frame, the quadric \( Q_R \) has equation (up to a factor)

\[
x^T M^T R M x' = 4x'y' + z'^2 - cw'^2.
\]

In accordance with Equation (1) we choose \( x' = L x \), \( y' = L'y \). We apply the Gauss decomposition of quadratic forms into sum of squares to \( A \) and set \( z' = c[j] x[j] + c[jk] x[k] \) and \( w' = x[k] \). Precisely, we define \( M \) such that its adjoint has its first row equal to \( L \), its second row equal to \( L' \), and the last two rows equal to zero except for the coordinate \( [3, j] \) equal to \( c[j] \), the coordinate \( [3, k] \) equal to \( c[k] \), and the coordinate \( [4, k] \) equal to 1.

Straightforward computations show that the four columns of \( M \) can be simplified by the factors \( rc \), \( r \), \( 2r \), and \( 2r^2 \), respectively. We then get

\[
x^T M^T R M x = r^2 c(4xy + z^2 - \det(R) w^2). \tag{2}
\]

If \( i, j, k \) are equal to 2, 3, 4 respectively, \( M \) is equal to

\[
M = \begin{pmatrix}
-r & -c & R[1,3] & -r & R[1,3] \\
0 & 0 & r & 0 & 0 \\
0 & 0 & 0 & r & 0
\end{pmatrix}
\]

\[
\]

\[
\]

\[
\]

We can easily parameterize the quadric of Equation (2) and the parameterization of the original \( Q_R \) is, with \( \delta = \det(R) \) and \( (u, v) \) and \((s, t)\) in \( \mathbb{P}^1(R) \),

\[
M \begin{pmatrix} \delta \sqrt{\delta}, & sv \sqrt{\delta}, & (us - tv) \sqrt{\delta}, & us + tv \end{pmatrix}^T. \tag{3}
\]

To evaluate the height of the coefficients of this parameterization in terms of the coefficients of \( S \) and \( T \), first note that the matrix \( R \) is the matrix \( \lambda S + \mu T \) of the pencil such that \( (\lambda, \mu) \in \mathbb{P}^1 \) is solution of

\[
p^T (\lambda S + \mu T) p = 0.
\]

So \( (\lambda, \mu) = (-p^T T p, p^T S p) \) has height 1 and \( R = \lambda S + \mu T \) has height 2 in terms of the coefficients of \( S \) and \( T \). Also, \( \delta = \det(R) \) has height 4 in terms of the coefficients of \( R \) and thus height 8 in terms of the coefficients of \( S \) and \( T \). When \( \delta \) is a square, \( \sqrt{\delta} \) has height 4.

The heights of the four columns of \( M \) are 1, 3, 2 and 4 respectively in terms of the coefficients of \( R \), and so 2, 6, 4 and 8 respectively in terms of the coefficients of \( S \) and \( T \).

\[\text{Figure 1: Evolution of the height of } \Delta(u, v) \text{ (smooth quartic case) as a function of the size of the input, with the standard deviation displayed on the optimized plot.}\]

\[T. \] The worst case for the height of the coefficients of the parameterization of \( Q_R \) happens when \( \sqrt{\delta} \) is a square, which we assume for the rest of the proof. It then follows from (3) that the coordinates of the parameterization of the \( Q_R \) are polynomials of the form

\[
\rho_1 ut + \rho_2 sv + \rho_3 us + \rho_4 tv
\]

where the heights of \( \rho_1, \ldots, \rho_4 \) are 6, 10, 8 and 8, respectively, in terms of the coefficients of \( S \) and \( T \). Let \( h_s = 10 \) and \( h_t = 8 \) denote the maximum heights of the coefficients of \( s \) and \( t \), respectively.

When substituting the parameterization into the equation of one of the initial quadrics (say \( S \)), we obtain an equation which can be written as

\[
a s^2 + b st + c t^2 = 0, \tag{5}
\]

where \( a, b, c \) depend on \((u, v)\). It follows from (4) that the heights of \( a, b, c \) are \( h_a = 21 \) \((= 2h_s + 1)\), \( h_b = 19 \) \((= h_s + h_t + 1)\) and \( h_c = 17 \) \((= 2h_t + 1)\), respectively. When substituting the solution \((s = 2c, t = -h \pm \sqrt{b^2 - 4ac})\) into the parameterization (3) we obtain a parameterization of the smooth quartic in which each coordinate has the form

\[
\alpha(u, v) \pm \beta(u, v) \sqrt{b^2 - 4ac}.
\]

The height of the coefficients of \( \alpha \) is 27 \((= h_s + h_t = h_s + h_c)\), the height of the coefficients of \( \beta \) is 8 \((= h_t)\) and the coefficients of \( \Delta \) is 38 \((= 2h_s = h_a + h_c)\).

The dependence of the coefficients of the parameterization on the rational point \( p \) can be computed by trivially adapting the above proof. The end result is that the height in terms of the coordinates of \( p \) is:

- 37 for the coefficients of the polynomials of \( X_1 \),
- 12 for the coefficients of the polynomials of \( X_2 \),
- 50 for the coefficients of \( \Delta(u, v) \).

These values are slightly bigger than the heights in terms of the coefficients of \( S \) and \( T \). In practice, however, the “complexity” of the coefficients of the parameterization is governed by their dependence on \( S \) and \( T \). Indeed, we have observed that most of the times the coordinates of \( p \) are integers between \(-2 \) and \( 2 \) and they have almost no impact on the result.
Figure 1 shows how the observed height of the coefficients of $\Delta(u,v)$ evolves as a function of the input size $s$ for three different versions of our implementation (see Section 4 for the details). For each value of $s$ in a set of samples between 0 and 60, we have generated random quadrics with coefficients in the range $[-10^6, 10^6]$, computed the height of the coefficients of the parameterizations of the smooth quartic and averaged the results. The plots of Figure 1 show that the observed height of the coefficients converges to 38 when no gcd computation is performed for simplifying the output parameterization, and seems to converge to 36 when gcd computation is performed. We ran experiments with inputs of size up to 10,000 and observed the same limit of 36 on the height of the coefficients when gcd computation is performed. As of now, we do not have a satisfactory explanation for this behavior.

3.2 Cubic and tangent line

We now consider the case of an intersection made of a cubic and a tangent line. In this case, we can parameterize the intersection using an intermediate rational quadric $Q_R$ of inertia either (2, 2) or (2, 1): the pencil contains an instance of both types of quadrics.

We prove the following theoretical bounds on the height of the coefficients of the parameterizations of the cubic and the line.

**Proposition 3.2.** When a quadric $Q_R$ of inertia (2, 2) is used to parameterize the intersection, the coefficients of the parameterizations of the cubic and the line have height at most 27 in terms of the coefficients of $S$ and $T$.

**Proof.** In this situation, the determinant of the intermediate quadric $Q_R$ is necessarily a square because $Q_R$ contains a rational line (see [2]). So the bounds found in the proof of Proposition 3.1 apply. In particular the heights $h_a, h_b, h_a, h_b,$ and $h_c$ of the coefficients of Equations (4) and (5) apply. Equation (5) factors into two terms, one of degree 0 and the other of degree 2 in, say, $(u, v)$, and both linear in, say, $(s, t)$; it can be written as

$$(as + bt)(α's + β't) = a^2 + b s t + c t^2 = 0$$

where $α, β$ are constants and $α', β'$ are polynomials in $(u, v)$. Since $αβ + 3αβ = b, α$ and the coefficients of $α'$ have height at most $h_a$. Similarly, $ββ + c$ thus $β$ and the coefficients of $β'$ have height at most $h_b$. Substituting the solutions ($s = β, t = -α$) and ($s = β, t = -α'$) into the parameterization (3), we get parameterizations of the cubic and the line whose coefficients have height at most $max(h_a, h_b, h_b + h_b) = max(10 + 17, 8 + 19) = 27$. □

**Proposition 3.3.** When a quadric $Q_R$ of inertia (2, 1) is used to parameterize the intersection, the coefficients of the parameterization of the line have height 11 and the coefficients of the parameterization of the cubic have height 22 in terms of the coefficients of $S$ and $T$.

**Proof.** We follow the algorithm outline given in Section 2.2.3 to determine the height of the output.

Let $λ_0$ be the quadruple real root of the determinantal equation. The determinantal equation writes down as

$$\det R(λ) = γ(λ - λ_0)^4.$$  

Since the coefficients of $\det R(λ)$ have height 4 in terms of the coefficients of $S$ and $T$ (they are $4 \times 4$ determinants), $λ_0$ has height 1. So the coefficients of the cone $R = R(λ_0) = λ_0S - T$ have height 2.

Finding the singular point of $Q_R$ amounts to finding a point $c ∈ P^3(Z)$ in the kernel of $R$, i.e. such that $Rc = 0$. If we decompose $R$ such that $R_u$ is the upper left 3 × 3 matrix of $R$ and $r_u$ is the first three coordinates of the last column of $R$, $c$ such that $c_u$ is the first three coordinates and $c_l$ is the last and we assume $det R_u ≠ 0$ ($R$ has rank 3 so one of its $3 × 3$ minors is non-zero), then $c$ is found by solving

$$R_u c_u = -c_l r_u.$$  

A solution is thus $c = (-R_u r_u, det R_u)$, where $R_u$ is the adjoint of $R_u$. The coefficients of $R_u$ and $r_u$ have height respectively 2 and 1 in terms of the coefficients of $R$. So the coordinates of $c$ have height 3 in terms of the coefficients of $R$, i.e. 6 in terms of the coefficients of $S$ and $T$.

Since the line of the intersection is the (double) intersection of $Q_R$ and the tangent plane to $Q_S$ at $c$, a point $p$ on this line is any point satisfying

$$R p \subset Sc.$$

(Observe that if $p$ is solution, any $a_1 p + a_2 c$ is also solution.) The right-hand side of (6) has height $6 + 1 = 7$ in the coefficients of $S$ and $T$. As above, one can assume that det $R_u ≠ 0$ and there is a unique point $p$ having zero as last coordinate. This point satisfies $p_u = R_u (Sc)_u$ and the height of its coordinates is $4 + 7 = 11$. Overall, the coefficients of the line $(c, p)$ have height 11.

Now, we need to parameterize a cone ($Q_R$) containing a rational point ($p$). First, we apply to $R$ a projective transformation $P$ sending $c$ to the point $(0, 0, 0, 1)^T$ and $p$ to the point $(0, 0, 1, 0)^T$. $R' = P^T R P$. Taking the upper left $3 \times 3$ submatrix of $R'$, we are left with the problem of parameterizing a non-singular conic going through the origin. Such a conic has equation

$$ax^2 + bxy + cy^2 + dyz + exz = 0$$

and a parameterization is given by

$$(e 0 d \begin{bmatrix} u^2 & v^2 & w^2 \end{bmatrix}, (u, v, w) ∈ 3(\mathbb{R}),$$

where $a, b, c$ have height 2 and $d, e$ have height 11. Lifting this parameterization to the original space by multiplying by matrix $P$, we find a parameterization of $Q$:

$$X(u, v, s), (u, v, s) ∈ P^4(\mathbb{R})$$

(real quasi-projective space) such that the coefficients of $u^2, v^2, w^2, u^2 + v^2 + w^2$ have height $11 + 2 = 13$ and the coefficients of $s$ have height 6. Plugging $X(u, v, s)$ in the equation of any other quadric of the pencil gives an equation in the parameters of the form:

$$as^2 + b(u, v)s + c(u, v) = 0,$$

where $b(u, v)$ is a polynomial of degree 2 and height $1 + 6 + 13 = 20$ and $c(u, v)$ a polynomial of degree 4 and height $2 \times 13 + 1 = 27$. Observe first that $a = 0$ since the singularity of the cone is reached at $(u, v) = (0, 0)$ and at this point $s ≠ 0$. We also know that (8) has a linear factor corresponding to the line of the intersection and it is not too difficult to realize that this factor is $e u + d v$, where $d$ and $e$ are as
in (7). So, after factoring out the linear term, (8) can be rewritten
\[ b'(u, v)s + c'(u, v) = 0, \]
where \( b'(u, v) \) is a polynomial of degree 1 and height 20 – 11 = 9 and \( c'(u, v) \) is a polynomial of degree 3 and height 27 – 11 = 16. We can solve this equation rationally for \( s \) and, multiplying to clear the denominators, we get a parameterization of the cubic of height 16 + 6 = 22.

The difference in the heights of the parameterizations under-scored in the above two propositions is vindicated by some examples could be generated by taking input quadrics with a quadric \((2, 2)\) and \((2, 1)\) and a random transformation. More precisely, given a canonical pair \( S, T \), four random bigints \( r_1, r_2, r_3, r_4 \) and a random real projective transformation \( P \), we take as new input quadrics the quadrics of matrix:

\[ S' = P^T (r_1 S + r_2 T) P, \quad T' = P^T (r_3 S + r_4 T) P. \]

If we take all the random coefficients in the range \([-\sqrt{\text{MAXFACTOR}}, \sqrt{\text{MAXFACTOR}}]\), then the quadrics \( S' \) and \( T' \) have size \( s \) (the size of the canonical pair \( S, T \) can be neglected).

Figure 2 shows the height of the coefficients of the parameterization of the cubic when a quadric \( Q_H \) of inertia \((2, 2)\) or \((2, 1)\) is used (with mild optimizations turned on). The plots clearly show that the coefficients of the cubic are smaller when a cone is used to parameterize the intersection. The fact that the observed heights are, in the limit, so different from the theoretical bounds \((8 \text{ instead of } 22 \text{ when a cone is used})\) is possibly a consequence of the way \( S' \) and \( T' \) are generated: it certainly does not reflect a truly random distribution in the space of quadrics with integer coefficients of size \( s \) intersecting in a cubic and a tangent line.

Figure 3 further reinforces our choice of using a cone: the parameterizations have not just smaller coefficients, they are also faster to compute.

4. EXPERIMENTAL RESULTS
We now report on some experimental results and findings from our implementation.

The experiments were made on a Dell Precision 360 with a 2.60 GHz Intel Pentium CPU. LiDIA, GMP and our own code were compiled with g++ 3.2.2.

4.1 Implementation versions
In what follows, we compare three versions of our implementation:

- **unoptimized**: nothing is done to simplify the coefficients either during the computations or in the parameterizations computed;
- **mildly optimized**: some gcds are performed at an early stage (optimization of the coefficients and of the roots of the determinantal equation, optimization of the coordinates of singular and rational points, . . . ) to avoid hampering later calculations with unnecessarily big numbers;
- **strongly optimized**: mildly optimized, plus extraction of the square factors of some bigints (like in the smooth quartic case, where \( \sqrt{\delta} = \sqrt{\text{det} R} \) can be replaced by \( b_2/\alpha \) if \( \text{det} R = a b_2^2 \)) and gcd simplifications of the coefficients of the final parameterizations.

Concretely, mild optimization is turned on in our implementation by a \#DEFINE OPTIMIZE. Strong optimization uses in addition a \(-o\) argument passed to the binary. For the extraction of the square factors of an integer \( n \), it finds all the prime factors of \( n \) up to \( \min (\sqrt{\text{MAXFACTOR}}, \text{MAXFACTOR}) \), where MAXFACTOR is a predefined global variable.

Let us finally mention that we tried a fourth version of our implementation where the extraction of the square factors is done by fully factoring the numbers (using the Elliptic Curve Method and the Quadratic Sieve implemented in LiDIA [9]). But this version is almost of no interest: for small input coefficients, the strongly optimized version already finds all the necessary factors, and for medium to large input coefficients, integer factoring becomes extremely time consuming.

4.2 Performance evaluation
Let us first discuss the impact of the MAXFACTOR variable on the output. Figure 4 shows that a value of 10⁸ has a dramatic impact on computation time while all values less than 10⁴ are acceptable. We have determined that the best compromise between efficiency and complexity of the output is obtained by setting MAXFACTOR to 10³, which we assume now.

Figure 5 shows the evolution of the aggregate computation time in the smooth quartic case, which is the most com-
Deciding to spend time on optimization essentially depends on the application. For most real-world applications, where the size of the input quadrics is small by construction, we believe optimizing is important: it should be kept to mind that the parameterizations computed are used for further processing (like in boundary evaluation) and limiting the growth of the coefficients at an early stage is a good idea.

A last comment that can be made looking at Figure 4 concerns the efficiency of our implementation. Indeed, those plots show that we can compute the parameterization of the intersection of two quadrics with coefficients having 400 digits in 1 second and 1000 digits in 5 seconds (on average).

Efficiency can be measured in a different way. In Figure 6, we have plotted the total computation time, with the strongly optimized version, for a file containing 120 pairs of quadrics covering all intersection situations over the reals. The “random” quadrics were generated as in Section 3.2. For an input size $s = 500$, the total computation time is roughly 72 seconds, on average, for the 120 pairs of quadrics, i.e. 0.6 second per intersection. This should be compared to the 1.7 seconds on average needed to compute the intersection in the smooth quartic case for the same size of input (Figure 4). This difference is simply explained by the fact that very degenerate intersections (like when the determinantal equation vanishes identically, which represents 36 of the 120 quadrics in the file) are usually much faster to compute.

Our last word will be on memory consumption. Our implementation eats up very little of memory. In the smooth quartic case, the total memory chunks allocated sum up to less than 64 kilobytes for input sizes up to 20. It takes input sizes of more than 700 digits to go in the 1 MB range of used memory.

5. EXAMPLES

We now give three examples of parameterizations computed with our algorithm. Other examples can be tested by querying our parameterization server.

Comparing our results with the parameterizations computed with other methods does not make much sense since our implementation is the first to output exact parameterizations in all cases. However, for the sake of illustration, our first two examples are taken from the paper describing the plane cubic curve method of Wang, Joe and Goldman [20].

5.1 Example 1

Our first example is Example 4 from [20]. It is the intersection $C$ of a quadric of inertia $(2, 1)$ (elliptic cylinder) and
Our implementation outputs the following exact and simple result in less than 10 ms:

$$X(u, v) = \frac{2 u^3 - 6 u^2}{2 u^2 + 4 u^2 + 3 v^2} \pm \frac{(-2 v)}{2 u} \sqrt{-3 u^4 + 26 u^2 v^2 - 3 v^4}.$$  

The polynomials involved in the parameterization are defined in $\mathbb{Q}[u, v]$, which means we are in the lucky case where the intermediate quadric of inertia $(2, 2)$ found to parameterize the intersection has a square as determinant. So the parameterization obtained is optimal in the extension of $\mathbb{Z}$ on which its coefficients are defined.

### 5.2 Example 2

Our second example is Example 5 from [20]. It is the intersection of a sphere and an ellipsoid that are very close to one another (see Figure 7.b):

$$C : 19 x^2 + 22 y^2 + 21 z^2 - 20 w^2 = x^2 + y^2 + z^2 - w^2 = 0.$$  

In [20], the authors give the following parameterization for $C$:

$$X_1(u, v) = \begin{pmatrix} \frac{-0.72 u^3 - 0.72 u^2 v + 0.08 u v^2 - 0.08 v^3}{0.0} \\ 0.0 \end{pmatrix},$$  

and $\Delta(u, v) = 905.0967 u^2 - 3328.0 u^2 v^2 + 2896.3094 u^3$. The authors report a computation error on this example (measured as the maximum distance from a sequence of sample points on the curve to the input quadrics) of order $O(10^{-3})$.  

The polynomials involved in the parameterization are defined in $\mathbb{Q}[u, v]$, which means we are in the lucky case where the intermediate quadric of inertia $(2, 2)$ found to parameterize the intersection has a square as determinant. So the parameterization obtained is optimal in the extension of $\mathbb{Z}$ on which its coefficients are defined.
Our last example illustrates the fact that our implementation can be avoided in certain cases. It turns out that in this particular example it can be avoided. We have presented the C++ implementation of an algorithm that it is possible, though not necessary, that the square root parameterized. Plugging this parameterization in the equation of inertia, it is based on the


It turns out that in this particular example it can be avoided. Consider the cone $Q_R$ corresponding to the rational root $(-1, 21)$ of the determinantal equation:

$$ Q_R = -Q_S + 21 Q_T = 2u^2 - y^2 - w^2. $$

$Q_R$ contains the obvious rational point $(1, 1, 0, 1)$, which is not its singular point. It implies that it can be reasonably parameterized. Plugging this parameterization in the equation of $Q_S$ or $Q_T$ gives a simple parameterization for $C$:

$$ X(u, v) = \left( \begin{array}{c} u^2 + 2u^2v^2 \\ 2uv \\ u^2 - 2v^2 \\ 0 \end{array} \right) \pm \sqrt{2u^4 + 4u^2v^2 + 8v^4}. $$

### 5.3 Example 3

Our last example illustrates the fact that our implementation is complete in the sense that it computes parameterizations in all possible cases. It concerns two quadrics intersecting in two tangent conics. The execution trace (with debug information turned off) is given in Output 2.

### 6. CONCLUSION

We have presented the C++ implementation of an algorithm for parameterizing intersections of quadrics. The implementation is exact, efficient and covers all the possible cases of intersection. It is based on the LiDIA library and uses the multiprecision integer arithmetic of GMP.

Future work will be devoted to understanding the gaps between predicted and observed values for the height of the coefficients of the parameterizations, to working out predicates and filters for making the code robust with floating point data (many classes and data structures have already been templated for a future use with floating point coefficients) and to porting our code to the standard geometry library, i.e. CGAL.

### 7. REFERENCES


