On using sums-of-squares for exact computations without strict feasibility

David Monniaux

On using sums-of-squares for exact computations without strict feasibility

David Monniaux
CNRS / VERIMAG
2 avenue de Vignate
38610 Gières
France
David.Monniaux@imag.fr

ABSTRACT

One can reduce the problem of proving that a polynomial is nonnegative, or more generally of proving that a system of polynomial inequalities has no solutions, to finding polynomials that are sums of squares of polynomials and satisfy some linear equality obtained by a Positivstellensatz. This problem itself reduces to a feasibility problem in semidefinite programming. Unfortunately, this last problem is in general not strictly feasible — the solution set then is a convex with empty interior, which precludes direct use of numerical optimization methods. We propose a workaround for this difficulty.

We implemented our method and illustrate its use with examples including automatically showing that various nonnegative polynomials that have been shown not to be sums of squares of polynomials are indeed nonnegative, as quotients of two sums of squares of polynomials.

Categories and Subject Descriptors

G.1.6 [**Numerical analysis**]: Optimization—*Convex programming*; F.4.1 [**Mathematical logic and formal languages**]: Mathematical logic—*Mechanical theorem proving*

General Terms

Algorithms, Verification

1. INTRODUCTION

Consider the following problem: given a conjunction of polynomial equalities, and (wide and strict) polynomial inequalities, with integer or rational coefficients, decide whether this conjunction is satisfiable over $\mathbb{R}$; that is, whether one can assign real values to the variables so that the conjunction holds.

The decision problem for real polynomial inequalities can be reduced to *quantifier elimination*: given a formula F, whose atomic formulas are polynomial (in)equalities, containing quantifiers, provide another, equivalent, formula F', whose atomic formulas are still polynomial (in)equalities, containing no quantifier. An algorithm for quantifier elimination over the theory of *real closed fields* (roughly speaking, $(\mathbb{R}, 0, 1, +, \times, \leq)$ was first proposed by Tarski [26] and Seidenberg [24], but this algorithm had non-elementary complexity and thus was impractical. Later, the *cylindrical algebraic decomposition* (CAD) algorithm was proposed by Collins [7], with a doubly exponential complexity, but despite improvements [8] CAD is still slow in practice and there are few implementations available.

Quantifier elimination is not the only decision method. Basu et al. [2, Theorem 3] proposed a satisfiability testing algorithm with complexity $s^{k+1}d^{O(k)}$, where s is the number of distinct polynomials appearing in the formula, d is their maximal degree, and k is the number of variables. We know of no implementation of that algorithm. Tiwari [27] proposed an algorithm based on rewriting systems that is supposed to answer in reasonable time when a conjunction of polynomial inequalities has no solution.

Many of the algebraic algorithms are complex, which leads to complex implementations. This poses a methodology problem: can one trust their results? The use of computer programs for proving lemmas used in mathematical theorems was criticized in the case of Thomas Hales' proof of the Kepler conjecture. Similarly, the use of complex decision procedures (as in the proof assistant PVS[1]) or program analyzers (as, for instance, Astrée[2]) in order to prove the correctness of critical computer programs is criticized on grounds that these verification systems could themselves contain bugs.

One could either formally prove correct the implementation of the algorithm using a proof assistant such as Coq [12], as suggested by Mahboubi [17], or one could arrange for the decision procedure to provide a *witness* of its result. The answer of the procedure is correct if the witness is correct, and correctness of the witness can be checked by a much simpler procedure. The idea is to reduce the size and complexity of the software to trust.

We know how to provide unsatisfiability witnesses for systems of complex equalities or linear rational inequalities. An additional benefit is that in both cases, the witness directly indicates the subset of hypotheses that was used to prove unsatisfiability; this is not only precious for the end-user (which

[1] `http://pvs.csl.sri.com/`
[2] `http://www.astree.ens.fr/`

hypotheses are actually useful) but also means that such procedures can be used as theory solvers within DPLL(T) satisfiability modulo theory decision procedures [15, ch. 11]. It is therefore tempting to seek unsatisfiability witnesses for systems of polynomial inequalities.

Harrison [11], Parrilo [18] have suggested looking for proof witnesses whose existence is guaranteed by the *Positivstellensatz* [25]. They reduce the original problem to: given polynomials P_i and R, find polynomials Q_i that are sums of squares such that $\sum_i P_i Q_i = R$. This last problem easily reduces to a *semidefinite programming* (sdp) feasibility problem, solved by numerical methods. Peyrl and Parrilo [19] explain how to obtain suitable witnesses for a sub-problem of the general problem that we study in this article, namely proving that a given polynomial is a sum of squares. Unfortunately, their method suffers from a caveat: it applies only under a *strict feasibility* condition: a certain convex geometrical object should not be degenerate, that is, it should have nonempty interior. Unfortunately it is very easy to obtain problems where this condition is not true. Equivalently, the method of rationalization of certificates suggested by Kaltofen et al. [13] has a limiting requirement that the rationalized moment matrix remains positive semidefinite.

In this article, we explain how to work around the degeneracy problem: we propose a method to look for rational solutions to a general sdp feasibility problem. We have implemented our method and applied it to example.

2. WITNESSES

For many interesting theories, it is trivial to check that a given valuation of the variables satisfies a quantifier-free formula. A satisfiability decision procedure will in this case tend to seek a *satisfiability witness* and provide it to the user when giving a positive answer

In contrast, if the answer is that the problem is not satisfiable, the user has to trust the output of the satisfiability testing algorithm, the informal meaning of which is "I looked carefully everywhere and did not find a solution.". In some cases, it is possible to provide *unsatisfiability witnesses*: solutions to some form of dual problem that show that the original problem had no solution. In order to introduce the *Positivstellensatz* approach, we first briefly explain two simpler, but similar, problems with unsatisfiability witnesses.

2.1 Unsatisfiability Witnesses for Linear Inequalities

Let C be a conjunction of (strict or wide) linear inequalities. A satisfiability witness is just a valuation such that the inequalities hold, and can be obtained by linear programming for instance.

Can we also have unsatisfiability witnesses? For the sake of simplicity, let us consider the case where all the inequalities are wide and take C to be $L_1(x_1, \ldots, x_m) \geq 0 \wedge \cdots \wedge L_n(x_1, \ldots, x_m) \geq 0$ where the L_i are affine linear forms. Obviously, if $\alpha_1, \ldots, \alpha_n$ are nonnegative coefficients, then if C holds, then $\sum \alpha_i L_i(x_1, \ldots, x_m) \geq 0$ also holds. Thus, if one can exhibit $\alpha_1, \ldots, \alpha_n \geq 0$ such that $\sum \alpha_i L_i = -1$ — otherwise said, a *trivial contradiction* —, then C does not hold. The vector $(\alpha_1, \ldots, \alpha_n)$ is thus an unsatisfiability witness.

This refutation method is evidently *sound*, that is, if such a vector can be exhibited, then the original problem had no solution. It is also *complete*: one can always obtain such a vector if the original problem C is unsatisfiable, from Farkas' lemma [10, §6.4, theorem 6].

2.2 Unsatisfiability Witnesses for Complex Polynomial Equalities

Let C be a conjunction of polynomial equalities $P_1(x_1, \ldots, x_m) = 0 \wedge \cdots \wedge P_n(x_1, \ldots, x_m) = 0$ whose coefficients lie in a subfield K (say, the rational numbers $\mathbb{Q}$) of an algebraically closed field K' (say, the complex numbers $\mathbb{C}$). C is said to be satisfiable if one can find a valuation in K' of the variables in C such that the equalities hold. We shall now show that one can get unsatisfiability witnesses that are checkable using simple methods, only involving adding and multiplying polynomials over K.

Obviously, if one can find $Q_1, \ldots, Q_n \in K[x_1, \ldots, x_m]$ such that $\sum_i P_i Q_i = 1$, then C has no solution. Again, this method of finding a trivial contradiction is both sound and complete for refutation. The completeness proof relies on a theorem known as *Nullstellensatz*:

THEOREM 1 (HILBERT). *Let K' be an algebraically closed field, let I be an ideal in $K'[x_1, \ldots, x_n]$. Let P be a polynomial in $K'[x_1, \ldots, x_n]$. P vanishes over the common zeroes of the ideals in I if and only if some nonnegative power of P lies in I.*

Apply that theorem to $P = 1$ and I the ideal generated by $P_1, \ldots, P_m$. $P = 1$. By the theorem, the P_i have no common solution if and only if 1 lies in I, that is, there exists $\bar{Q}_1, \ldots, \bar{Q}_m \in K'[x_1, \ldots, x_n]$ such that $\sum_i \bar{Q}_i P_i = 1$. K' is a vector space over K, thus K has a supplemental space S in K'. By projecting the coefficients of the $\bar{Q}_i$ onto K, one obtains polynomials $Q_i \in K[x_1, \ldots, x_n]$ such that $\sum_i Q_i P_i = 1$. Those Q_i constitute a unsatisfiability witness for C. One can compute such a witness by dividing the unit polynomial by a Gröbner basis of the ideal generated by the Q_i; the remainder is zero if such a witness exists [9].

Note that this algorithm is sound *but incomplete* when K' is not algebraically closed (e.g. the real field $\mathbb{R}$). For instance, the polynomial $x^2 + 1$ has no real solution, yet the polynomial 1 is not a member of the ideal generated by it and this method will thus not conclude that $x^2 + 1 = 0$ has no real solution.

2.3 Nonnegativity Witnesses

To prove that a polynomial P is nonnegative, one simple method is to express it as a sum of squares of polynomials. One good point is that the degree of the polynomials involved in this sum of squares can be bounded, and even that the choice of possible monomials is constrained by the Newton polytope of P, as seen in §3.

Yet, there exist nonnegative polynomials that cannot be expressed as sums of squares, for instance this example due to Motzkin [22]:

$$M = x_1^6 + x_2^4 x_3^2 + x_2^2 x_3^4 - 3 x_1^2 x_2^2 x_3^2 \quad (1)$$

However, Artin's answer to Hilbert's seventeenth problem is that any nonnegative polynomial can be expressed as a sum of squares of rational functions [1]. It follows that such a polynomial can always be expressed as the quotient Q_2/Q_1 of two sums of squares of polynomials, which forms the nonnegativity witness, and can be obtained by solving $P.Q_1 - Q_2 = 0$.

2.4 Unsatisfiability Witnesses for Polynomial Inequalities

For the sake of simplicity, we shall restrict ourselves to wide inequalities (the extension to mixed wide/strict inequalities is possible). Let us first remark that the problem of testing whether a set of wide inequalities with coefficients in a subfield K of the real numbers is satisfiable over the real numbers is equivalent to the problem of testing whether a set of *equalities* with coefficients K is satisfiable over the real numbers: for each inequality $P(x_1, \ldots, x_m) \geq 0$, replace it by $P(x_1, \ldots, x_m) - \mu^2 = 0$, where μ is a new variable. Strict inequalities can also be simulated as follows: $P_i(x_1, \ldots, x_m) \neq 0$ is replaced by $P_i(x_1, \ldots, x_m).\mu = 1$ where μ a μ is a new variables. One therefore does not gain theoretical simplicity by restricting oneself to equalities.

Stengle [25] proved two theorems regarding the solution sets of systems of polynomial equalities and inequalities over the reals (or, more generally, over real closed fields): a *Nullstellensatz* and a *Positivstellensatz*. Without going into overly complex notations, let us state consequences of these theorems. Let K be an ordered field (such as $\mathbb{Q}$) and K' be a real closed field containing K (such as the real field $\mathbb{R}$). The corollary of interest to us [16] is:

THEOREM 2. *Let $\{Z_1, \ldots, Z_{n_z}\}$, $\{P_1, \ldots, P_{n_p}\}$ be two (possibly empty) sets of polynomials in $K[x_1, \ldots, x_m]$. Then $Z_1(x_1, \ldots, x_m) = 0 \wedge \cdots \wedge Z_{n_z}(x_1, \ldots, x_m) = 0 \wedge P_1(x_1, \ldots, x_m) \geq 0 \wedge \cdots \wedge P_{n_p}(x_1, \ldots, x_m) \geq 0$ has no solution in K'^m if and only if there exist some polynomials A and B such that $A + B = -1$, $A \in I(Z_1, \ldots, Z_{n_z})$ and $B \in S(P_1, \ldots, P_{n_p})$, where $I(Z_1, \ldots, Z_{n_z})$ is the ideal generated by the $Z_1, \ldots, Z_{n_z}$ and $S(P_1, \ldots, P_{n_p})$ is the semiring generated by the positive elements of K and $P_1^2, \ldots, P_{n_p}^2$.*

Note that this result resembles the one used for linear inequalities (Section 2.1), replacing nonnegative numbers by sums of squares of polynomials.

For a simple example, consider the following system, which obviously has no solution:

$$\begin{cases} -2 + y^2 \geq 0 \\ 1 - y^4 \geq 0 \end{cases} \tag{2}$$

A *Positivstellensatz* witness is $y^2(-2+y^2)+1(1-y^4)+2y^2 = -1$. Another is $\left(\frac{2}{3} + \frac{y^2}{3}\right)(-2 + y^2) + \frac{1}{3}(1 - y^4) = -1$.

Consider the conjunction C: $P_1 \geq 0 \wedge \cdots \wedge P_n \geq 0$ where $P_i \in \mathbb{Q}[X_1, \ldots, X_m]$. Consider the set S of products of the form $\prod_{w \in \{0,1\}^{\{1,\ldots,n\}}} P_i^{w_i}$ — that is, the set of all products of the P_i where each P_i appears at most once. Obviously, if one can exhibit nonnegative functions Q_j such that $\sum_{P_j \in S} Q_j P_j = -1$, then C does not have solutions. Theorem 2 guarantees that if C has no solutions, then such functions Q_j exist as sum of squares of polynomials. We have again reduced our problem to the following problem: given polynomials P_j and R, find sums-of-squares polynomials Q_j such that $\sum Q_j P_j = R$.

3. SOLVING THE SUMS-OF-SQUARES PROBLEM

In §2.3 and §2.4, we have reduced our problems to: given polynomials $(P_j)_{1 \leq j \leq n}$ and R in $\mathbb{Q}[X_1, \ldots, X_m]$, find polynomials that are sums of squares Q_j such that

$$\sum_j P_j Q_j = R \tag{3}$$

We wish to output the Q_j as $Q_j = \sum_{i=1}^{n_j} \alpha_{ji} L_{ji}^2$ where $\alpha_{ji} \in \mathbb{Q}^+$ and L_{ji} are polynomials over $\mathbb{Q}$. We now show how to solve this equation.

3.1 Reduction to Semidefinite Programming

Lemma 4 ensures that each Q_j can be expressed as $M_j \hat{Q}_j M_j^T$ where $\hat{Q}_j$ is a symmetric positive semidefinite matrix (noted $\hat{Q}_j \succeq 0$) and M_j is a vector of monomials.

Assume that we know the M_j, but we do not know the matrices $\hat{Q}_j$. The equality $\sum_j P_j(M_j \hat{Q}_j (M_j)^t) = R$ directly translates into a system (S) of affine linear equalities over the coefficients of the $\hat{Q}_j$: $\sum_j (M_j \hat{Q}_j (M_j)^t) P_j - R$ is the zero polynomial, so its coefficients, which are affine linear combinations of the coefficients of the $\hat{Q}_j$ matrices, should be zero; each of these combinations thus yields an affine linear equation. The additional requirement is that the $\hat{Q}_j$ are positive semidefinite.

One can equivalently express the problem by grouping these matrices into a block diagonal matrix $\hat{Q}$ and express the system (S) of affine linear equations over the coefficients of $\hat{Q}$. By Gaussian elimination in exact rational arithmetic, or other faster exact linear algebra methods, we can obtain a system of generators for the solution set of (S): $\hat{Q} \in -F_0 + \mathrm{vect}(F_1, \ldots, F_m)$. The problem is then to find a positive semidefinite matrix within this *search space*; that is, find $\alpha_1, \ldots, \alpha_m$ such that $-F_0 + \sum_i \alpha_i F_i \succeq 0$.

This is the problem of *semidefinite programming*: finding a positive semidefinite matrix within an affine linear variety of symmetric matrices, optionally optimizing a linear form [28, 5]. Powers and Wörmann [20], Parrilo [18, chapter 4] and others have advocated such kind of decomposition for finding whether a given polynomial is a sum of squares. Harrison [11] generalized the approach to finding unsatisfiability witnesses.

For instance, the second unsatisfiability witness we gave for constraint system 2 is defined, using monomials $\{1, y\}$, 1 and $\{1, y\}$, by:

$$\left(\begin{array}{cc|c|cc} \frac{2}{3} & 0 & & & \\ 0 & \frac{1}{3} & & & \\ \hline & & \frac{1}{3} & & \\ \hline & & & 0 & 0 \\ & & & 0 & 0 \end{array}\right)$$

Conjunction C has no solution if and only if there exists a set of monomials and associated positive semidefinite matrices verifying some linear relations. Positive semidefiniteness is a semialgebraic property of the matrix coefficients, defined by the signs of the coefficients of the characteristic polynomial of the matrix, which are polynomials in the coefficients of the matrices. Thus, C has no solution if and only there is a set of monomials such that some set of wide polynomial inequalities has a solution.

It looks like finding an unsatisfiability witness for C just amounts to a sdp problem. There are, however, several problems to this approach:

1. For the general *Positivstellensatz* witness problem, the set of polynomials to consider is exponential in the number of inequalities.

2. Except for the simple problem of proving that a given polynomial is a sum of squares, we do not know the degree of the Q_j in advance, so we cannot choose finite sets of monomials M_j. The dimension of the vector space for Q_j grows quadratically in $|M_j|$.

3. Some sdp algorithms can fail to converge if the problem is not *strictly feasible* — that is, the solution set has empty interior, or, equivalently, is not full dimensional (that is, it is included within a strict subspace of the search space).

4. sdp algorithms are implemented in floating-point. If the solution space is not full dimensional, they tend to provide solutions $\hat{Q}$ that are "almost" positive semidefinite (all eigenvalues greater than $-\epsilon$ for some small positive ϵ), but not positive semidefinite.

Let us first consider the first two problems. Lombardi [16] provides a bound to the degrees of the polynomials necessary for the unsatisfiability certificates, but this bound is nonelementary (asymptotically greater than any tower of exponentials), so it is not of practical value. This bound, however, is only needed for the *completeness* of the refutation method: we are guaranteed to find the certificate if we look in a large enough space. It is not needed for *soundness*: if we find a correct certificate by looking in a portion of the huge search space, then that certificate is correct regardless. This means that we can limit the choice of monomials in M_j and hope for the best.

Regarding the second and third problems : what is needed is a way to reduce the dimension of the search space, ideally up to the point that the solution set is full dimensional. Peyrl and Parrilo [19] suggests applying a result of Reznick [21, Th. 1]: in a sum-of-square decomposition of a polynomial P, only monomials $x_1^{\alpha_1} \dots x_n^{\alpha_n}$ such that $2(\alpha_1, \dots, \alpha_n)$ lies within the Newton polytope[3] of P can appear. This helps reduce the dimension if P is known in advance (as in a sum-of-squares decomposition to prove positivity) but does not help for more general equations.

Kaltofen et al. [14] suggest solving the sdp problem numerically and looking for rows with very small values, which indicate useless monomials that can be safely removed from the basis; in other words, they detect "approximate kernel vectors" from the canonical basis. Our method is somehow a generalization of theirs: we detect kernel vectors whether or not they are from the canonical basis.

In the next section, we shall investigate the fourth problem: how to deal with solution sets with empty interior.

3.2 How to deal with degenerate cases

In the preceding section, we have shown how to reduce the problem of finding unsatisfiability witnesses to a sdp feasibility problem, but pointed out one crucial difficulty: the possible degeneracy of the solution set. In this section, we explain more about this difficulty and how to work around it.

Let $\mathcal{K}$ be the cone of positive semidefinite matrices. We denote by $M \succeq 0$ a positive semidefinite matrix M, by $M \succ 0$ a positive definite matrix M. $\mathbf{y}$ denotes a vector, y_i is the i-th coordinate of $\mathbf{y}$. $\tilde{x}$ denotes a floating-point value close to an ideal real value x.

We consider a sdp feasibility problem: given a family of symmetric matrices $F_0, F_i, \dots, F_m$, find $(y_i)_{1 \leq i \leq m}$ such that

$$F(\mathbf{y}) = -F_0 + \sum_{i=1}^{m} y_i F_i \succeq 0. \tag{4}$$

The F_i have rational coefficients, and we suppose that there is at least one rational solution for $\mathbf{y}$ such that $F(\mathbf{y}) \succeq 0$. The problem is how to find such a solution.

If nonempty, the solution set $S \subseteq \mathbb{R}^m$ for the $\mathbf{y}$, also known as the *spectrahedron*, is semialgebraic, convex and closed; its boundary consists in $\mathbf{y}$ defining singular positive semidefinite matrices, its interior are positive definite matrices. We say that the problem is strictly feasible if the solution set has nonempty interior. Equivalently, this means that the convex S has dimension m.

Interior point methods used for semidefinite feasibility, when the solution set has nonempty interior, tend to find a solution in the interior away from the boundary. Around any solution $\mathbf{y}$ in the interior of S, there exists an open ball $B(\mathbf{y}, \epsilon) \subseteq S$. It follows that if we take $\mathbf{y}_{\mathbb{Q}}$ a rationalization of $\mathbf{y}$ such that $\|\mathbf{y}_{\mathbb{Q}} - \mathbf{y}\| \leq \epsilon$, then $\mathbf{y}_{\mathbb{Q}}$ is also in S, thus the problem is solved. This is why earlier works on sums-of-square methods [19] have proposed finding rational solutions only when the sdp problem is strictly feasible. In this article, we explain how to do away with the strict feasibility clause.

Some problems are not strictly feasible. Geometrically, this means that the linear affine space $\{-F_0 + \sum_{i=1}^{m} y_i F_i \mid (y_1, \dots, y_m) \in \mathbb{R}^m\}$ is tangent to the semidefinite positive cone $\mathcal{K}$. Alternatively, this means that the solution set is included in a strict linear affine subspace of $\mathbb{R}^m$. Intuitively, this means that we are searching for the solution in "too large a space"; for instance, if $m = 2$ and $\mathbf{y}$ lies in a plane, this happens if the solution set is a point or a segment of a line. In this case, some sdp algorithms may fail to converge if the problem is not strictly feasible, and those that converge, in general, will find a point slightly outside the solution set. The main contribution of this article is a workaround for this problem.

3.3 Simplified algorithm

If S has nonempty interior, then an interior point solving method will lead to a solution $\tilde{\mathbf{y}}$ in this interior, assuming enough precision is used. Choose a very close approximation $\mathbf{y}$ to this vector using rational numbers (perhaps with only small denominators), then unless we are unlucky $\mathbf{y}$ is also in the interior of S. Thus, $F(\mathbf{y})$ is a solution of problem 4. We shall thus now suppose the problem has empty interior.

Suppose we have found a numerical solution $\tilde{\mathbf{y}}$, but it is nearly singular — meaning that it has some negative eigenvalues extremely close to zero. This means there is $\mathbf{v}$ such that $|\mathbf{v}.F(\tilde{\mathbf{y}})| \leq \epsilon$. Suppose that $\tilde{\mathbf{y}}$ is very close to a rational solution $\mathbf{y}$ and, that $\mathbf{v}.F(\mathbf{y}) = 0$, and also that $\mathbf{y}$ is in the relative interior of S — that is, the interior of that set relative to the least linear affine space containing S. Then, by lemma 9, all solutions $F(\mathbf{y}')$ also satisfy $\mathbf{v}.F(\mathbf{y}') = 0$. Remark that the same lemma implies that either there is no rational solution in the relative interior, either that rational solutions are dense in S.

How can finding such a $\mathbf{v}$ help us? Obviously, if $\mathbf{v} \in \bigcap_{i=0}^{m} \ker F_i$, its discovery does not provide any more information than already present in the linear affine system

[3]The Newton polytope of a polynomial P, or in Reznick's terminology, its *cage*, is the convex hull of the vertices $(\alpha_1, \dots, \alpha_n)$ such that $x_1^{\alpha_1} \dots x_n^{\alpha_n}$ is a monomial of P.

$-F_0 + \text{Vect}\,(F_1, \ldots, F_m)$. We thus need to look for a vector outside that intersection of kernels; then, knowing such a vector will enable us to reduce the dimension of the search space from m to $m' < m$.

We therefore look for such a vector in the orthogonal complement of $\bigcap_{i=0}^{m} \ker F_i$, which is the vector space generated by the rows of the $F_0, \ldots, F_m$. We therefore compute a full rank matrix B whose rows span the exact same space; this can be achieved by echelonizing a matrix obtained by stacking $F_0, \ldots, F_m$. Then, $\mathbf{v} = \mathbf{w}B$ for some vector $\mathbf{w}$. We thus look for $\mathbf{w}$ such that $G(\mathbf{y}).\mathbf{w} = 0$, with $G(\mathbf{y}) = BF(\mathbf{y})B^T$.

The question is how to find such a $\mathbf{w}$ with rational or, equivalently, integer coefficients. Another issue is that this vector should be "reasonable" — it should not involve extremely large coefficients, which would basically amplify the floating-point inaccuracies.

We can reformulate the problem as: find $\mathbf{w} \in \mathbb{Z}^m \setminus \{0\}$ such that both $\mathbf{w}$ and $G(\tilde{\mathbf{y}}).\mathbf{w}$ are "small", which can be combined into a single constraint $\alpha^2 \|G(\tilde{\mathbf{y}}).\mathbf{w}\|_2^2 + \|\mathbf{w}\|_2^2$, where $\alpha > 0$ is a coefficient for tuning how much we penalize large values of $\|G(\tilde{\mathbf{y}}).\mathbf{w}\|_2$ in comparison to large values of $\|\mathbf{w}\|_2$. If α is large enough, the difference between $\alpha G(\tilde{\mathbf{y}})$ and its integer rounding M is small. We currently choose $\alpha = \alpha_0 / \|G(\tilde{\mathbf{y}})\|$, with $\|M\|$ the Frobenius norm of M (the Euclidean norm for $n \times n$ matrices being considered as vectors in $\mathbb{R}^{n^2}$), and $\alpha_0 = 10^{15}$.

We therefore try searching for a small (with respect to the Euclidean norm) nonzero vector that is an integer linear combination of the $l_i = (0, \ldots, 1, \ldots, 0, m_i)$ where m_i is the i-th row of M and the 1 is at the i-th position. Note that, because of the diagonal of ones, the l_i form a free family.

This problem is known as finding a short vector in an integer lattice, and can be solved by the Lenstra-Lenstra-Lovász (LLL) algorithm. This algorithm outputs a free family of vectors s_i such that s_1 is very short. Other vectors in the family may also be very short.

Once we have such a small vector $\mathbf{w}$, we can compute $F'_0, \ldots, F'_{m'}$ such that

$$\left\{ -F'_0 + \sum_{i=1}^{m'} y'_i F'_i \mid (y_1, \ldots, y_{m'}) \in \mathbb{R}^{m'} \right\} = \left\{ -F_0 + \sum_{i=1}^{m} y_i F_i \mid (y_1, \ldots, y_m) \in \mathbb{R}^m \right\} \cap \{F \mid F.\mathbf{v} = 0\} \quad (5)$$

This is just linear algebra over rational numbers and may be performed by Gaussian elimination, for example. The resulting system has lower search space dimension $m' < m$, yet the same solution set dimension. By iterating the method, we eventually reach a search space dimension equal to the dimension of the solution set.

3.4 More efficient algorithm

In lieu of performing numerical sdp solving on $F = -F_0 + \sum y_i F_i \succeq 0$, we can perform it in lower dimension on $-(BF_0B^T) + \sum y_i (BF_iB^T) \succeq 0$. Recall that the rows of B span the orthogonal complement of $\bigcap_{i=0}^{m} \ker F_i$, which is necessarily included in $\ker F$; we are therefore just leaving out dimensions that always provide null eigenvalues.

The reduction of the sums-of-squares problem (Eq. 3) provides matrices with a fixed block structure, one block for each P_j: for a given problem all matrices $F_0, F_1, \ldots, F_m$ are block diagonal with respect to that structure. We therefore perform the test for positive semidefiniteness of the proposed $F(\mathbf{y})$ solution block-wise, in exact arithmetic. This test may for instance be done by computing the Gaussian reduction of F using the algorithm for conversion of a positive semidefinite matrix to a sum-of-square from §A.2, which fails if and only if the matrix is not positive semidefinite. For the blocks not found to be positive semidefinite, the corresponding blocks of the matrices B and $F(\tilde{\mathbf{y}})$ are computed, and LLL is performed.

As described so far, only a single $\mathbf{v}$ kernel vector would be supplied by LLL for each block not found to be positive semidefinite. In practice, this tends to lead to too many iterations of the main loop: the dimension of the search space does not decrease quickly enough. We instead always take the first vector $\mathbf{v}^{(1)}$ of the LLL-reduced basis, then accept following vectors $\mathbf{v}^{(i)}$ if $\|\mathbf{v}^{(i)}\|_1 \leq \beta.\|\mathbf{v}^{(1)}\|_1$ and $\|G(\tilde{\mathbf{y}}).\mathbf{v}^{(i)}\|_2 \leq \gamma.\|G(\tilde{\mathbf{y}}).\mathbf{v}^{(1)}\|_2$. For practical uses, we took $\beta = \gamma = 10$.

When looking for the next iteration $\tilde{\mathbf{y}'}$, we use the $\tilde{\mathbf{y}}$ from the previous iteration as a hint: instead of starting the sdp search from an arbitrary point, we start it near the solution found by the previous iteration. We perform least-square minimization so that $-F'_0 + \sum_{i=1}^{m'} y'_i F'_i$ is the best approximation of $-F_0 + \sum_{i=1}^{m} y_i F_i \mid (y_1, \ldots, y_m)$.

3.5 Implementation details

The reduction from the problem expressed in Eq. 3 to sdp with rational solutions was implemented in Sage.[4] For efficiency, the rational sdp solving was implemented in C++. Exact rational arithmetic uses the GMP extended precision library;[5]; exact linear algebra is performed by the C++ template library Boost uBlas.[6] Numerical sdp solving is performed using DSDP[7] [4, 3] as a library. LLL reduction is performed by fpLLL.[8] Least square projection is performed using Lapack's DGELS. The implementation is available from the author's web page.[9]

Exact precision rational linear arithmetic is expensive when the size of the numerators and denominators grows, which is unfortunately the case in our settings. Our choice of implementation is obviously from optimal; we expect clever integer linear arithmetic such as provided by Linbox to perform better.[10]

3.6 Extensions

As seen in §4, our algorithm tends to produce solutions with large numerators and denominators in the sum-of-square decomposition. We experimented with methods to get $F(\mathbf{y}') \approx F(\mathbf{y})$ such that $F(\mathbf{y}')$ has a smaller common denominator. This reduces to the following problem: given $\mathbf{v} \in \mathbf{f}_0 + \text{vect}(\mathbf{f}_1, \ldots, \mathbf{f}_n)$ a real (floating-point) vector and $\mathbf{f}_0, \ldots, \mathbf{f}_n$ rational vectors, find $\mathbf{y}'_1, \ldots, \mathbf{y}_n$ such that $\mathbf{v}' = \mathbf{f}_0 + \sum_i y'_i \mathbf{f}_i \approx \mathbf{v}$ and the numerators of $\mathbf{v}'$ have a tunable magnitude (pa-

[4] Sage is a computer algebra system implemented within the Python programming language, available under the GNU GPL from `http://www.sagemath.org`.

[5] `http://www.gmplib.org/`

[6] See `http://www.boost.org`

[7] DSDP is a sdp tool available from `http://www.mcs.anl.gov/DSDP/`

[8] fpLLL is a LLL library from Damien Stehlé et al., available from `http://perso.ens-lyon.fr/damien.stehle/`

[9] `http://www-verimag.imag.fr/~monniaux/download/safe_sdp.zip`

[10] `http://www.linalg.org/linbox-html/index.html`

rameter μ). One can obtain such a result by LLL reduction of the rows of:

$$M = \begin{pmatrix} \mathbb{Z}(\beta\mu(\mathbf{f}_0 - \mathbf{v})) & \mathbb{Z}(\beta\mathbf{f}_0) & 1 & 0 & \dots & 0 \\ \mathbb{Z}(\beta\mu\mathbf{f}_1) & \mathbb{Z}(\beta\mathbf{f}_1) & 0 & 1 & \dots & \\ \vdots & \vdots & 0 & & \ddots & \\ \mathbb{Z}(\beta\mu\mathbf{f}_n) & \mathbb{Z}(\beta\mathbf{f}_n) & 0 & & & 1 \end{pmatrix} \quad (6)$$

where β is a large parameter (say, 10^{19}) and $\mathbb{Z}(v)$ stands for the integer rounding of v. After LLL reduction, one of the short vectors in the basis will be a combination $\sum_i^n y_i l_i$ where $l_0, \dots, l_n$ are the lines of M, such that $y_0 \neq 0$. Because of the large $\beta\mu$ coefficient, $y_0(\mathbf{f}_0 - \mathbf{v}) + \sum_{i=1}^n y_i \mathbf{f}_i$ should be very small, thus $f_0 + \sum_{i=1}^n y_i \mathbf{f}_i \approx v$. But among those vectors, the algorithm chooses one such that $\sum_{i=0}^n y_i \mathbf{f}_i$ is not large — and among the suitable v', the vector of numerators is proportional to $\sum_{i=0}^n y_i \mathbf{f}_i$.

After computing such a y', we check whether $F(\mathbf{y}') \succeq 0$; we try this for a geometrically increasing sequence of μ and stop as soon as we find a solution. The matrices $\hat{Q}_j$ then have simpler coefficients than the original ones. Unfortunately, it does not ensue that the sums of square decompositions of these matrices have small coefficients.

We have so far considered sdp feasibility problems only. It is also possible to use the same method for optimization problems. First, the feasibility problem should be solved by the above method; a parametrization $-F'_0 + \text{vect}(F'_1, \dots, F'_m)$ of the linear affine span of the solution set is thus computed, as well as a point $-F'_0 + \sum_i y'_i F'_i \succeq 0$. In this new parametrization, the problem is strictly feasible, and one can use a sdp system to obtain y'' such that $F'(y'') = -F'_0 + \sum_i y''_i F'_i \succeq 0$ numerically and $F'(y'')$ maximizes a linear form. If, after rationalizing y'', $F'(y'')$ is not found to be positive semidefinite (because it has some slightly negative eigenvalues), one can consider a barycenter $F''(\epsilon y' + (1-\epsilon)y'')$ for small ϵ.

3.7 Wrong leads

A crucial issue when using multiple vectors from the output of LLL is to know which vectors can safely be considered to be vectors from the kernel of the approximated matrices: if we take vectors that are not from that kernel, then we risk overconstraining the search space and getting no solution from the algorithm.

We considered computing the signature of a rationalization of the $F(\tilde{\mathbf{y}})$ matrix, using Gaussian reduction, and taking as many leading vectors of th e output of the LLL basis as $F(\tilde{\mathbf{y}})$ has nonpositive eigenvalues. Unfortunately, this does not work — this scheme may take too many vectors.

4. EXAMPLES

The following system of inequalities has no solution (neither Redlog nor QepCad nor Mathematica 5 can prove it; Mathematica 7 can):

$$\begin{cases} P_1 = x^3 + xy + 3y^2 + z + 1 \geq 0 \\ P_2 = 5z^3 - 2y^2 + x + 2 \geq 0 \\ P_3 = x^2 + y - z \geq 0 \\ P_4 = -5x^2z^3 - 50xyz^3 - 125y^2z^3 + 2x^2y^2 + 20xy^3 \\ \quad +50y^4 - 2x^3 - 10x^2y - 25xy^2 - 15z^3 - 4x^2 \\ \quad -21xy - 47y^2 - 3x - y - 8 \geq 0 \end{cases} \quad (7)$$

This system was concocted by choosing P_1, P_2, P_3 somewhat haphazardly and then $P_4 = -(P_1 + (3 + (x+5y)^2)P_2 + P_3 + 1 + x^2)$, which guaranteed the system had no solution. The initial 130 constraints yield a search space of dimension 145, and after four round of numeric solving one gets an unsatisfiability witness (sums of squares Q_j such that $\sum_{j=1}^4 P_j Q_j + Q_5 = 0$). Total computation time was 4.4 s. Even though there existed a simple solution (note the above formula for P_4), our algorithm provided a lengthy one, with large coefficients (and thus unfit for inclusion here).

Motzkin's polynomial M (Eq. 1) cannot be expressed as a sum of squares, but it can be expressed as a quotient of two sums of squares. We attempted solving $M.Q_1 - Q_2 = 0$ for sums of squares Q_1 and Q_2 built from homogeneous monomials of respective total degrees 3 and 6 (lesser degrees yield no solutions). The equality relation over the polynomials yields 66 constraints over the matrix coefficients and a search space of dimension 186. Four cycles of SDP programming and LLL are then needed, total computation time was 4.1 s.

The following solution is obtained:
$Q_1 = 8006878A_1^2 + 29138091A_2^2 + 25619868453870$
$/4003439A_3^2 + 14025608A_4^2 + 14385502A_5^2 + 85108577038951965167$
$/12809934226935A_6^2$
$Q_2 = 8006878B_1^2 + 25616453B_2^2 + 108749058736871$
$/4003439B_3^2 + 161490847987681$
$/25616453B_4^2 + 7272614B_5^2 + 37419351B_6^2 + 13078817768190$
$/3636307B_7^2 + 71344030945385471151$
$/15535579819553B_8^2 + 539969700325922707586$
$/161490847987681B_9^2 + 41728880843834$
$/12473117B_{10}^2 + 1310088572084630189 14/62593321265751B_1 1^2$,
where
$A_1 = -1147341/4003439x_1^2x_3 - 318460/4003439x_2^2x_3 + x_3^3$
$A_2 = x_2x_3^2$, $A_3 = -4216114037644/12809934226935x_1^2x_3 + x_2^2x_3$
$A_4 = x_1x_3^2$, $A_5 = x_1x_2x_3$, $A_6 = x_1^2x_3$ and $B_1 = -1102857$
$/4003439x_1^4x_2x_3 - 5464251/4003439x_1^2x_2x_3^3 + 2563669$
$/4003439x_2^3x_3^3 + x_2x_3^5$, $B_2 = -9223081/25616453x_1^4x_2^2 - 18326919$
$/25616453x_1^2x_2^2x_3^2 + 1933547/25616453x_2^4x_3^2 + x_2^2x_3^4$,
$B_3 = -2617184886847/15535579819553x_1^4x_2x_3 - 12918394932706$
$/15535579819553x_1^2x_2x_3^3 + x_2^3x_3^3$, $B_4 = -26028972147097$
$/161490847987681x_1^4x_3^2 - 13546187584 0584$
$/161490847987681x_1^2x_2^2x_3^2 + x_2^4x_3^2$, $B_5 = -2333331$
$/3636307x_1^3x_2x_3^2 - 1302976/3636307x_1x_2^3x_3^2 + x_1x_2x_3^4$,
$B_6 = -11582471/37419351x_1^5x_3 - 12629854$
$/37419351x_1^3x_2^2x_3 - 4402342/12473117x_1^3x_3^3 + x_1x_2^2x_3^3$,
$B_7 = -x_1^3x_2x_3^2 + x_1x_2^3x_3^2$, $B_8 = -x_1^4x_2x_3 + x_1^2x_2x_3^3$,
$B_9 = -x_1^4x_3^2 + x_1^2x_2^2x_3^2$, $B_{10} = -17362252580967$
$/20864440421917x_1^5x_3 - 3502187840950$
$/20864440421917x_1^3x_2^2x_3 + x_1^3x_3^3$, $B_{11} = -x_1^5x_3 + x_1^3x_2^2x_3$.

We exhibited witnesses that each of the 8 semidefinite positive forms listed by Reznick [22], which are not sums of squares of polynomials, are quotients of sums of squares (Motzkin's M, Robinson's R and f, Choi and Lam's F, Q, S, H and Schmüdgen's q). These examples include polynomials with up to 6 variables and search spaces up to dimension 1155. We did likewise with *delzell*, *laxlax* and *leepstarr2* from [14]. The maximal computation time was 7'. In some cases, it was necessary to initialize the numerical solver with a nonzero solution matrix in order to avoid the undesirable trivial solution $Q_1 = Q_2 = 0$.

We also exhibited a witness that the Vor1 polynomial cited by Safey El Din [23] is a sum of squares.[11] We could not apply our method to the Vor2 polynomial because of our inefficient implementation of exact linear algebra.

[11]Tool logs and witnesses are available from `http://www-verimag.imag.fr/~monniaux/download/sdp_logs.zip`.

5. CONCLUSION AND FURTHER WORKS

We have described a method for solving sdp problems in rational arithmetic. This method can be used to solve sums-of-squares problems even in geometrically degenerate cases. We illustrated this method with applications to proving the nonnegativity of polynomials, or the unsatisfiability of systems of polynomials. The method then provides easily checkable proof witnesses.

One weakness of the method is that it tends to provide "unnatural" witnesses — they tend to have very large coefficients. These are machine checkable but provide little insights to the reader.

A more serious limitation for proofs of unsatisfiability is the very high cost of application of the Positivstellensatz. There is the exponential number of polynomials to consider, and the unknown number of monomials. It would be very interesting if there could be some simple results, similar to the Newton polytope approach, for reducing the dimension of the search space or the number of polynomials to consider. Another question is whether it is possible to define sdp problems from Positivstellensatz equations for which the spectrahedron has rational points only at its relative boundary.

While our method performed well on examples, and is guaranteed to provide a correct answer if it provides one, we have supplied no completeness proof — that is, we have not proved that it necessarily provides a solution if there is one. This is due to the use of floating-point computations. One appreciable result would be that a solution should be found under the assumption that floating-point computations are precise up to ϵ, for a value of ϵ and the various scaling factors in the algorithm depending on the values in the problem or the solution.

A desirable property of an unsatisfiability proof procedure is that is uses only a minimal set of hypotheses (or, better, a set of hypotheses of minimal cardinality). This would correspond to finding a block diagonal solution matrix $\hat{Q}$ with a maximal set of zero diagonal blocks (or a set of maximal cardinal). Heuristically, one would expect that it helps to look for $\hat{Q}$ of minimal rank within the solution set. This, however, would involve finding solutions on the relative boundary of the solution set, while our method finds solution in the relative interior of that set. In any case, it is possible to use a minimization procedure for the set of hypotheses [6, min function].

It seems possible to combine our reduction method based on LLL with the Newton iterations suggested by [14, 13], as an improvement over their strategy for detection of useless monomials and reduction of the search space. Again, further experiment is needed.

Acknowledgements.

We wish to thank Alexis Bernadet for discussions and ideas regarding the degeneracy problem, Mohab Safey El Din and Assia Mahboubi for their helpful comments and ideas, and François Morain for his idea of trying LLL.

References

[1] E. Artin. *Collected papers*. Addison-Wesley, 1965.

[2] S. Basu, R. Pollack, and M.-F. Roy. On the combinatorial and algebraic complexity of quantifier elimination. *Journal of the ACM (JACM)*, 43(6):1002–1045, 1996. doi: http://doi.acm.org/10.1145/235809.235813.

[3] S. J. Benson and Y. Ye. Algorithm 875: DSDP5—software for semidefinite programming. *ACM Transactions on Mathematical Software*, 34(3):16:1–16–20, May 2008. doi: 10.1145/1356052.1356057.

[4] S. J. Benson and Y. Ye. DSDP5 user guide — software for semidefinite programming. technical memorandum 277, Argonne National Laboratory, 2005.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. URL `http://www.stanford.edu/~boyd/cvxbook/`.

[6] A. R. Bradley and Z. Manna. Property-directed incremental invariant generation. *Formal Aspects of Computing*, 20(4–5):379–405, July 2008. doi: 10.1007/s00165-008-0080-9.

[7] G. E. Collins. Quantifier elimination for real closed fields by cylindrical algebraic decomposition. In *Automata theory and formal languages*, volume 33 of *LNCS*, pages 134–183. Springer, 1975.

[8] G. E. Collins. Quantifier elimination by cylindrical algebraic decomposition — twenty years of progress. In B. F. Caviness and J. R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, pages 8–23, 1998.

[9] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 3 edition, 2007.

[10] G. B. Dantzig. *Linear Programming and Extensions*. Princeton, 1998.

[11] J. Harrison. Verifying nonlinear real formulas via sums of squares. In K. Schneider and J. Brandt, editors, *Proceedings of the 20th International Conference on Theorem Proving in Higher Order Logics, TPHOLs 2007*, volume 4732 of *LNCS*, pages 102–118. Springer, 2007.

[12] Coq. *The Coq Proof Assistant Reference Manual*. INRIA, 8.1 edition, 2006.

[13] E. Kaltofen, B. Li, Z. Yang, and L. Zhi. Exact certification of global optimality of approximate factorizations via rationalizing sums-of-squares with floating point scalars. In *ISSAC (International Symposium on Symbolic and Algebraic Computation)*, pages 155–163. ACM, 2008. doi: 10.1145/1390768.1390792.

[14] E. Kaltofen, B. Li, Z. Yang, and L. Zhi. Exact certification in global polynomial optimization via sums-of-squares of rational functions with rational coefficients. Accepted at J. Symbolic Computation, 2009.

[15] D. Kroening and O. Strichman. *Decision procedures*. Springer, 2008.

[16] H. Lombardi. Une borne sur les degrés pour le théorème des zéros réel effectif. In M. Coste, L. Mahé, and c. Roy, Marie-Fran editors, *Real Algebraic Geometry: Proceedings of the Conference held in Rennes, France in 1991*, volume 1524 of *Lecture Notes in Mathematics*. Springer, 1992.

[17] A. Mahboubi. Implementing the CAD algorithm inside the Coq system. *Mathematical Structures in Computer Sciences*, 17(1), 2007. doi: doi:10.1017/S096012950600586X.

[18] P. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technol-

ogy, 2000.
[19] H. Peyrl and P. A. Parrilo. Computing sum of squares decompositions with rational coefficients. *Theoretical Computer Science*, 409(2):269–281, 2008. doi: 10.1016/j.tcs.2008.09.025.
[20] V. Powers and T. Wörmann. An algorithm for sums of squares of real polynomials. *Journal of pure and applied algebra*, 127:99–104, 1998.
[21] B. Reznick. Extremal PSD forms with few terms. *Duke Mathematical Journal*, 45(2):363–374, 1978. doi: 10.1215/S0012-7094-78-04519-2.
[22] B. Reznick. Some concrete aspects of hilbert's 17th problem. In C. N. Delzell and J. J. Madden, editors, *Real Algebraic Geometry and Ordered Structures*, volume 253 of *Contemporary Mathematics*. American Mathematical Society, 2000.
[23] M. Safey El Din. Computing the global optimum of a multivariate polynomial over the reals. In *ISSAC (International Symposium on Symbolic and Algebraic Computation)*, pages 71–78. ACM, 2008. doi: 10.1145/1390768.1390781.
[24] A. Seidenberg. A new decision method for elementary algebra. *Annals of Mathematics*, 60(2):365–374, Sept. 1954. URL `http://www.jstor.org/stable/1969640`.
[25] G. Stengle. A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 2 (207):87–07, 1973.
[26] A. Tarski. *A Decision Method for Elementary Algebra and Geometry*. University of California Press, 1951.
[27] A. Tiwari. An algebraic approach for the unsatisfiability of nonlinear constraints. In L. Ong, editor, *Computer Science Logic, 14th Annual Conf., CSL 2005*, volume 3634 of *LNCS*, pages 248–262. Springer, Aug. 2005.
[28] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, Mar. 1996.

APPENDIX

A. LEMMAS

A.1 Sums of squares and semidefinite positive matrices

LEMMA 3. *Let $v \in K^n$. Then, $v^T v$ is a $n \times n$ symmetric positive semidefinite matrix.*

PROOF. $v^T v$ is obviously symmetric. Let λ be a eigenvalue for it, and x a corresponding eigenvector: $xv^T v = \lambda x$. Thus, $\|vx^T\|_2^2 = (vx^T)^T(vx^T) = xv^T vx^T = \lambda xx^T = \|x\|_2^2$. Since $x \neq 0$, λ must be nonnegative. □

LEMMA 4. *Let $P \in K[X, Y, \dots]$ be a sum of squares of polynomials $\sum_i P_i^2$. Let $M = \{m_1, \dots, m_{|M|}\}$ be a set such that each P_i can be written as a linear combination of elements of M (M can be for instance the set of monomials in the P_i). Then there exists a $|M| \times |M|$ symmetric positive semidefinite matrix Q with coefficients in K such that $P(X, Y, \dots) = [m_1, \dots, m_{|M|}]Q[m_1, \dots, m_{|M|}]^T$, noting v^T the transpose of v.*

PROOF. Let us decompose $P_i(X, Y, \dots)$ into a linear combination of monomials $\sum_{1 \leq j \leq |M|} p_{i,j} m_j$. Let v_i be the vector $[p_{i,1}, \dots, p_{i,m}]$; then $P_i(X, Y, \dots) = v_i[m_1, \dots, m_{|M|}]^T$. $P_i^2(X, Y, \dots)$ is thus $[m_1, \dots, m_{|M|}]v_i^T v_i[m_1, \dots, m_{|M|}]^T$. $Q_i = v_i^T v_i$, by lemma 3 is symmetric positive semidefinite. $Q = \sum_i Q_i$ thus fulfills the conditions. □

Let us remark that the converse is correct for matrices over $\mathbb{R}$, by diagonalization: any symmetric positive semidefinite matrix is a sum of squares of linear forms. Such a decomposition can also be obtained over $\mathbb{Q}$ by applying the Gaussian reduction algorithm from §B.

A.2 Geometry of the semidefinite cone

Our algorithm from §3.3 relies crucially on some results about the geometry of the boundary of the semidefinite cone.

DEFINITION 5. *The* affine closure *of a set S is the least affine space containing S. The* relative interior *of a set S is the interior of S with respect to the affine closure of S.*

The following lemmas are from basic convex geometry:

LEMMA 6. *Let S be a nonempty convex set. S then has nonempty relative interior, which is obtained by taking $\lambda a + (1-\lambda)b$ for all $a, b \in S$ and $0 < \lambda < 1$. The affine closure of S is the same as the affine closure of the relative interior of S.*

LEMMA 7. *The tangent space of the affine closure of nonempty a convex set S is $\{t(x-y) \mid t \in \mathbb{R}, x, y \in S\}$.*

A basic lemma on quadratic forms:

LEMMA 8. *For $M \succeq 0$, the isotropic cone $\{v \mid v^T.M.v = 0\}$ is the same as the kernel $\ker M = \{v \mid M.v = 0\}$.*

LEMMA 9. *Let E be a linear affine subspace of the $n \times n$ symmetric matrices such that $E \cap \mathcal{K} \neq \emptyset$. F in the relative interior I of $E \cap \mathcal{K}$. Then,*

- *For all $F' \in E \cap \mathcal{K}$, $\ker F \subseteq \ker F'$.*
- *The least affine space containing $E \cap \mathcal{K}$ is $H = \{M \in E \mid \ker M = \ker F\}$.*

PROOF. Let $v \in \ker F$. Let $\Delta = F' - F$. Because I is relatively open, there exists $t > 0$ such that $F + t\Delta \in I$ and $F - t\Delta \in I$. $v^T.(F + t\Delta).v \geq 0$ and $v^T.(F - t\Delta).v \geq 0$ since $F + t\Delta \succeq 0$ and $F - t\Delta \succeq 0$. Since $v^T.F.v = 0$, $v^T.\Delta.v = 0$. Then $v^T.F'.v = 0$, thus, by lemmas 8 and 7, $v \in \ker F'$. Thus $\ker F \subseteq \ker F'$.

Let us now suppose $F' \in I$. The same reasoning could have been held after exchanging F and F', thus $\ker F = \ker F'$. Thus, by lemma 6 for all Δ in the tangent space of the affine closure of $E \cap \mathcal{K}$, $\Delta \in H$.

Let $\Delta \in H$. For sufficiently small t, $F + t\Delta$ has the same number of strictly positive eigenvalues as F, which, since F is semidefinite positive, is $n - \dim\ker F$. Note n_- the number of negative eigenvalues of $F + t\Delta$; then $n = (n - \dim\ker F) + n_- + \dim\ker(F + t\Delta)$. Since $\ker F = \ker\Delta$, $\dim\ker(F + t\Delta) \geq \dim\ker F$, thus $n_- \leq 0$ and $F + t\Delta$ cannot have negative eigenvalues. $F + t\Delta$ is thus semidefinite positive, and also belongs to E. Δ is thus in the tangent space of the affine closure of $E \cap \mathcal{K}$. □

COROLLARY 10. *Let E be a linear affine subspace of the $n \times n$ symmetric matrices such that $E \cap \mathcal{K} \neq \emptyset$. The relative interior of $E \cap \mathcal{K}$ consists of the matrices of maximal rank in $E \cap \mathcal{K}$.*

B. GAUSSIAN REDUCTION AND TEST OF POSITIVE SEMIDEFINITENESS

This appendix is not intended to be included in the conference proceeding. It recalls known results about positive semidefinite forms and their Gaussian reduction.

THEOREM 11 (SYLVESTER'S CRITERION). *A symmetric matrix $A = (a_{ij})_{1 \leq i \leq n, 1 \leq j \leq n}$ is positive definite if and only if its leading principal minors (the determinants of the "top left" $k \times k$ squares, $|(a_{ij})_{1 \leq i \leq k, 1 \leq j \leq k}$, for $k \leq n$) are positive.*

Note that this criterion cannot be immediately extended to semidefinite positive matrices by substituting "nonnegative" for "positive". For instance, all the leading minors of the following matrix are nonnegative, yet the matrix is not semidefinite positive:

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

A straightforward implementation of this criterion would compute n determinants, each with cost $\Theta(n^3)$ arithmetic operations, thus a cost $\Theta(n^4)$. This is however unneeded: instead, we compute a Gaussian reduction of the quadratic form:

```
for i:=1 to n do
begin
  if m[i, i] <= 0 then goto non_positive_definite
  v := m.row(i) / m[i, i]
  m := m - m[i, i] * v.transpose() * v
end
```

If the initial m is positive definite, then so it will be at each iteration the restriction of m to the lines and rows $i \ldots n$; conversely, if it is not, then the program will terminate.

At the beginning of an iteration, the matrix formed from the rows and columns $i \ldots n$, noted $m[i \ldots n, i \ldots n]$ is positive semidefinite, by induction hypothesis; by Th. 11 its leading principal minors are positive.

The m := m − m[i, i] ∗ v.transpose() ∗ v line can be decomposed into: it subtracts the first line, scaled, from the remaining lines (thus changing none of the leading principal minors), then it zeroes the first line. Before zeroeing, the rows and columns $i \ldots n$ look like:

$$\begin{pmatrix} m_{i,i} & m_{i,i+1} & \ldots \\ 0 & m_{i+1,i+1} & \ldots \\ \vdots & \vdots & \ddots \\ 0 & m_{n,i+1} & \ldots \end{pmatrix}$$

The leading principal minors of $m[i+1 \ldots n, i+1 \ldots n]$ are the same as those of the above matrix, times $m_{i,i}$. They are also positive, thus, by Th. 11, this matrix is also positive definite. Note that since the final result is not affected by the columns and rows above i, we could have used fewer operations.

A minor modification of this algorithm decides positive semidefiniteness:

```
for i:=1 to n do
begin
  if m[i, i] < 0 then
    goto non_positive_demidefinite
  if m[i, i] = 0 then
    if m.row(i) <> 0
      goto non_positive_demidefinite
  else
    begin
      v := m.row(i) / m[i, i]
      m := m - m[i, i] * v.transpose() * v
    end
end
```

Suppose that the entrance of iteration i, $m[i \ldots n, i \ldots n]$ is positive semidefinite. If $m_{i,i} = 0$, then the ith base vector is in the isotropic cone of the matrix, thus of its kernel, and the row i must be zero. Otherwise, $m_{i,i} > 0$. By adding ϵ to the diagonal of the matrix, we would have a positive definite matrix and thus the output of the loop iteration would also be positive definite, as above. By $\epsilon \to 0$ and the fact that the set of positive semidefinite matrices is topologically closed, then the output of the loop iteration is also positive semidefinite.

Finally, the same algorithm can also be used for transforming a semidefinite positive matrix into a "sum of squares" form, also known as Gaussian reduction:

```
for i:=1 to n do
begin
  if m[i, i] < 0 then
    goto non_positive_demidefinite
  if m[i, i] = 0 then
    if m.row(i) <> 0
      goto non_positive_demidefinite
  else
    begin
      v := m.row(i) / m[i, i]
      output.append(m[i, i], v)
      m := m - m[i, i] * v.transpose() * v
    end
end
```

The output variable is then a list of couples (c_i, v_i) such that the original matrix m is equal to $\sum_i c_i v_i^T v_i$ (with v_i row vectors). Otherwise said, for any row vector u, $umu^T = \sum_i c_i \langle u, v_i \rangle^2$.