Using Semidefinite Programming to Minimize Polynomials

Ruchira S. Datta

December 3, 2001

1 Motivation

Optimization problems arise in widely varying contexts. The general optimization problem is to find the minimum value of a certain function, the objective, on a certain set, defined by constraints. To make such problems amenable to analysis, further restrictions must be imposed on the kinds of objectives and constraints that may arise. A priori, it might seem useful to require them to be polynomial. After all, the entire toolbox of algebra and calculus is available to deal with polynomials. They can be represented, evaluated, and manipulated easily, and they are often used to approximate more complicated functions. Partly for these reasons, they arise in many kinds of applications.

As this course has demonstrated, it so happens that this is not the most fruitful restriction for optimization problems. Even problems with quadratic objectives and quadratic constraints may be difficult to solve. Rather, it is the field of *convex* optimization which has developed a body of theory and practice leading to computationally effective solution procedures. However, the aforementioned reasons why polynomial optimization would be desirable are still valid. Thus, this paper attempts to explain and demonstrate how to use the techniques of convex optimization to approximately (often, exactly) solve polynomial optimization problems. For concreteness, the problems will be posed as *minimization* problems. For simplicity, the constraints will be linear, or absent.

2 Why Not Just Calculus?

At first it might not be clear why this problem is nontrivial, particularly since a simple domain has been assumed. After all, everyone learns in calculus that a function attains its minimum (if it has one) either at a critical point or on the boundary of its domain. Since the function in question is polynomial, its partial derivatives are easy to compute. So, just set them all to zero and evaluate the function at each of those points. If it is in doubt whether or not a particular point is a minimum, evaluate the Hessian at that point. This requires computing second derivatives, which is also easy. To find the minimum on the boundary, simply specialize the polynomial and repeat the procedure on this new domain. In the cases under consideration this new domain will be of smaller dimension. Eventually it will be of dimension zero, and will consist of a finite number of points. Thus, the whole problem reduces to evaluating polynomials at certain points, namely, the critical points and the boundary points. Since it is easy to evaluate polynomials, the whole problem is now solved.

Alas, it is not that easy. First the critical points must be found. Even to find the zeroes of a single polynomial in one variable is more difficult than it might seem. As Galois showed, if the polynomial has degree at least five then no expression for the roots in terms of radicals may exist. Nevertheless, effective numerical procedures are known. But the set of critical points is the set of common zeroes of several polynomials in several variables. The study of such sets, called *algebraic sets*, is the main concern of algebraic geometry, a rich, deep, and vast subject. Gröbner bases currently constitute one of the main computational tools in this area. Briefly, a Gröbner basis for an algebraic set is another set of polynomials whose common zeroes form exactly that same algebraic set. But the polynomials in the Gröbner basis satisfy some additional technical properties which facilitate various computational algorithms. See for example [CLO97] or [BW93]. An additional piece of terminology will be required: the *ideal* generated by a set of polynomials consists of all sums of multiples of those polynomials (where the multipliers can be any other polynomials). Clearly all zeroes of a set of polynomials are also zeroes of the ideal generated by those polynomials, and vice versa. So the ideal defines the same algebraic set.

Modern tools are available to carry out Gröbner basis computations. The main difficulty arises from complexity considerations. The Gröbner basis found may be very large (in several different senses) compared to the original

set of polynomials. Another problem is that the set of critical points, called the *critical locus*, may not be zero-dimensional, i.e., there may be an infinite number of them. In this case the polynomial needs to be evaluated at one point in each connected component of the critical locus. A more fundamental problem is that these tools require working over the complex numbers. Minimization, on the other hand, requires a notion of ordering and thus intrinsically involves the real numbers. Thus many complex critical points, which have no relevance to the problem, will be enumerated needlessly.

3 Introducing The Examples

To demonstrate the various techniques, it is time to introduce the three running examples which will be used throughout this paper. The first example is the following polynomial of degree 6 in 3 variables:

$$\begin{aligned} p_1(x, y, z) &= 9x^2y^4 + 9x^2z^4 + 36x^2y^3 \\ &\quad +36x^2y^2 - 48xyz^2 \\ &\quad +4y^4 + 4z^4 - 16y^3 + 16y^2. \end{aligned}$$

Only the unconstrained case will be considered.

The second example is the following polynomial of degree 8 in 3 variables:

$$\begin{aligned} p_2(x, y, z) &= x^4y^2z^2 + x^2y^4z^2 + x^2y^2z^4 \\ &\quad -4x^2y^2z^2 + 1. \end{aligned}$$

This is the dehomogenization of the second Motzkin form; see for example [Rez00]. Only the unconstrained case will be considered.

The third example is the following polynomial of degree 10 in 4 variables:

$$\begin{aligned} p_3(a, b, c, d) &= (1 - a^2b^2)(1 - cd)(ad - bc)^2 \\ &\quad +2ab(cd - ab)(1 - ab)(c - d)^2 \\ &\quad +(a^2b^2 - c^2d^2)(1 - cd)(a - b)^2. \end{aligned}$$

Ronen Peretz wished to study this polynomial in connection with circle packings and the discrete Riemann mapping. Here only the constrained case $0 \leq a, b, c, d \leq 1$ will be considered. This polynomial has a fairly compact representation, yet it is already enough to strain the capabilities of various approaches to polynomial minimization.

4 Critical Points

This section demonstrates the use of calculus for polynomial optimization. Due to length considerations, only the first example, p_1, is described in detail; the others are sketched. Those interested primarily in the application of convex optimization techniques may wish to skip this section, which appears simply for purposes of comparison.

The partial derivatives of p_1 are $\partial p_1/\partial x = 18xy^4+18xz^4+72xy^3+72xy^2-48yz^2$, $\partial p_1/\partial y = 36x^2y^3+108x^2y^2+72x^2y-48xz^2+16y^3-48y^2+32y$, and $\partial p_1/\partial z = 36x^2z^3-96xyz+16z^3$. The computer algebra tool SINGULAR [GPS01] can be used to compute a Gröbner basis of the ideal generated by these polynomials, called the *Jacobian ideal* of p_1. This yields the following:

1. $9x^2y^2+4y^4-12xyz^2+4z^4+18x^2y-8y^3-12xz^2-4y^2+8y$
2. $9x^2z^3-24xyz+4z^3$
3. $9xy^2z^2+4y^4-2z^4-12y^3+8y^2$
4. $27x^2yz^2-18xz^4-12xy^3-12y^2z^2-96xy^2+44yz^2$
5. $y^5-3xy^2z^2+yz^4-y^4-6xyz^2+z^4-4y^3+4y^2$
6. $3xy^4+3xz^4+12xy^3+12xy^2-8yz^2$
7. $9xz^5+6xy^3z+6y^2z^3+12xy^2z-16yz^3$
8. $9xyz^4+6y^3z^2-15xz^4-18xy^3-22y^2z^2-36xy^2+32yz^2$
9. $2y^4z^2-6xyz^4+2z^6-4y^3z^2+3xz^4+18xy^3+2y^2z^2+36xy^2-16yz^2$

SINGULAR states that this ideal has codimension 2. The maximum possible codimension would be 3, since 3 variables are involved. Roughly, this would correspond to a 0-dimensional algebraic set, containing a finite number of points. For example, the ideal generated by 3 polynomials x, y, and z would define an algebraic set consisting of a single point, namely the origin. In the present case, since the codimension is not 3, there are an infinite number of critical points. So further work is needed to find the minimum.

It would be desirable to find the connected components of the critical locus. However, these cannot be computed directly. Instead, a useful available tool is the *primary decomposition*. A short explanation of the geometric

interpretation of primary decomposition appears as Section 3.8 of [Eis95]. For the present purposes it is sufficient to note that it yields a collection of algebraic sets, each of which is connected and has a well-defined dimension. The union of these sets is the original algebraic set. Unfortunately the sets may intersect, so picking a critical point from each of these sets may result in needless work; but this seems unavoidable.

The primary decomposition library [PDS01] of SINGULAR is used to calculate the primary decomposition of the Jacobian ideal:

1. the ideal generated by $y^4 + z^4 - 4y^2$, $3xz^2 + 2y^2 - 4y$, $3xy^2 + 6xy - 2z^2$, and $9x^2y + 18x^2 + 4y - 8$ is primary—indeed it is prime.

2. the ideal generated by z^3, yz, y^2, and $9x^2y - 6xz^2 + 4y$ is primary, with prime ideal generated by y and z.

3. the ideal generated by $81z^4 + 448$, $3y + 4$, and $-3z^2 + 4x$ is primary—indeed it is prime.

4. the ideal generated by z^3, $y - 1$, and $-8z^2 + 27x$ is primary—indeed it is prime.

5. the ideal generated by z^6, $-z^2 + 2y$, and $3x - 2$ is primary, with prime ideal generated by $3x - 2$, y, and z.

6. the ideal generated by z^6, $z^2 + 2y$, and $3x + 2$ is primary, with prime ideal generated by $3x + 2$, y, and z.

The abovementioned algebraic sets are the ones defined by the *prime* ideals above; it suffices to pick one point from each of them. Except for (1), this can be done by inspection. Note that (2) shows that the entire x-axis is contained in the critical locus; this could also be seen by inspecting the partial derivatives. Note also that (5) and (6) each define a single point on the x-axis, illustrating that the algebraic sets given by this decomposition may intersect or even lie within one another. Note further that $81z^4 + 448 = 0$ has no real solution, so (3) need not be considered.

Elementary techniques can also take care of (1). Just note that the equations $y^4 + z^4 - 4y^2 = 0$, $3xz^2 + 2y^2 - 4y = 0$, and $3xy^2 + 6xy - 2z^2 = 0$ hold on the entire x-axis. The last equation reduces on the x-axis to $18x^2 - 8 = 0$, or $x = \pm 2/3$. Thus the algebraic set defined by (1) intersects the x-axis and need not be considered separately.

Now p_1 must be evaluated at two points.

- $p_1(0, 0, 0) = 0$; this corresponds to (2)
- $p_1(0, 1, 0) = 4$; this corresponds to (4)

It is also necessary to consider what happens to p_1 asymptotically as its arguments approach ∞ along some line. The highest degree terms of p_1 are $9x^2y^4 + 9x^2z^4$. When $y = z = 0$, p_1 specializes to 0. When $x = 0$, p_1 specializes to $4y^4 + 4z^4 - 16y^3 + 16y^2$. The conclusion is that the global minimum of p_1 is zero.

Analysis of the second Motzkin polynomial p_2 proceeds similarly, although its high degree of symmetry leads to some simplifications. The Jacobian ideal has dimension 2 and codimension 1; SINGULAR computes a Gröbner basis with 7 elements. The primary decomposition consists of 17 ideals, but in fact it suffices (considering symmetry) to evaluate the polynomial at two points: the origin, where it takes the value 1, and the point $(1, 1, 1)$, where it vanishes. The critical locus of p_2 consists of the xy-plane, the yz-plane, the zx-plane, and the eight points $(\pm 1, \pm 1, \pm 1)$. The highest degree terms of p_2 are $(x^2 + y^2 + z^2)x^2y^2z^2$. If one of the variables is zero the polynomial reduces to 1. Thus the global minimum of p_2 is zero.

Since Peretz's polynomial p_3 is of degree 3 in c and d, it clearly could not have a global minimum or maximum in the unconstrained case. SINGULAR does succeed in computing a Gröbner basis of the Jacobian ideal of p_3. Unfortunately, this Gröbner basis consists of 103 polynomials. Some of them are of degree 10. The leading term of one of them is

$$1151118930719804789592240352920 0a^5bc^3d.$$

Attempts to make any further computations apparently cause SINGULAR to hang.

To conclude this section, finding the critical points worked for the polynomials of smaller degree. Some effort was saved through ad hoc tricks (for example, noticing which parts of the primary decomposition were unnecessary, which might not always be obvious upon inspection), but purely algorithmic methods would eventually have succeeded in finding the minimum. However, these exact methods quickly became too computationally intensive on a polynomial of moderate size. Thus the usefulness of approximate techniques becomes clear. These are the subject of the following sections.

5 Semidefinite Programming

5.1 Overview

The problem of minimizing a polynomial is related to the problem of determining whether the polynomial is *positive semidefinite* (or *p.s.d.*), i.e., is always nonnegative. It is clear that a solution to the former answers the latter question. Conversely, if the latter question can be answered for a family of polynomials obtained by adding a constant term to the original one, then each affirmative answer gives a lower bound on the original polynomial, and each negative answer gives an upper bound on the minimum.

Clearly a sufficient condition for a polynomial to be p.s.d. is that it be a sum of squares of other polynomials. (In what follows, these other polynomials will be called "subpolynomials".) However, this is not a necessary condition. Thus, the problem of determining whether a particular polynomial is a sum of squares, or *s.o.s.*, is a relaxation of the problem of determining whether it is p.s.d. The former problem can be solved by semidefinite programming, as discussed extensively by Pablo Parrilo in his thesis [Par00] and in [Par01].

The so-called "Gram matrix" is the fundamental construction. A polynomial cannot possibly be s.o.s. unless it has even degree, say $2d$. In that case it may be a sum of squares of subpolynomials, each of which has degree at most d. In fact, every monomial occurring in the original polynomial has degree at most $2d$ and can be expressed, possibly in more than one way, as a product of two monomials each of which has degree at most d. In what follows, the monomials of degree at most $2d$ will be called "supermonomials" and the monomials of degree at most d will be called "submonomials". Viewing each supermonomial as a product of submonomials, the whole polynomial is a sum of real numbers times products of two submonomials. That is, it is a quadratic form in the submonomials.

An example should make this clear. Consider the polynomial $p(x, y) = 2x^3y + 3x^2y^2 - xy^3$. Here the supermonomials, of degree 4, are x^3y, x^2y^2, and xy^3. Each can be written as a product of two submonomials of degree 2. For example, $x^3y = (x^2)(xy)$, $x^2y^2 = (xy)^2$, and $xy^3 = (xy)(y^2)$. This yields the following expression for p:

$$\begin{pmatrix} x^2 & xy & y^2 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 \\ 1 & 3 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & 0 \end{pmatrix} \begin{pmatrix} x^2 \\ xy \\ y^2 \end{pmatrix}$$

Of course, this expression is not unique. For example, $x^2y^2 = (x^2)(y^2)$ as well. In fact, in order that a particular quadratic form represent the polynomial, all that is required is that the sum of the coefficients (in the quadratic form) of all the products of two submonomials yielding the same supermonomial should equal the coefficient (in the polynomial) of that supermonomial. So the set of quadratic forms representing a given polynomial forms an affine subspace of the set of symmetric matrices, defined by the aforementioned constraints.

The key fact is that the polynomial is s.o.s. if and only if it can be represented by a p.s.d. quadratic form. Indeed, a subpolynomial can be considered as a linear form on the submonomials. Writing the sum of squares expression corresponds to diagonalizing the quadratic form.

The problem has now been reduced to an SDP feasibility problem. The question is whether there exists a p.s.d. matrix in the affine subspace of quadratic forms corresponding to this polynomial.

As explained above, to gain more information about the actual minimum of the polynomial, one could add various constants to the original polynomial and repeatedly solve the above problem. However, Pablo Parrilo and Bernd Sturmfels describe a much more direct method in [PS01]. Specifically, the affine subspace of quadratic forms is described parametrically, and an additional parameter λ, the constant term, is introduced. Assuming the first submonomial is 1, λ is the entry in the upper left corner of the symmetric matrix. The problem is now reduced to an SDP optimization problem: to find a p.s.d. matrix in the affine subspace which minimizes λ. If the original constant term of the polynomial is c, then the polynomial plus $\lambda - c$ is always nonnegative, i.e., the polynomial is never less than $c - \lambda$. So $c - \lambda$ is a lower bound for the minimum of the polynomial.

Some special-purpose auxiliary code was written to help carry out the above procedures. This Ocaml code [Ocaml] resides in a file `minpoly.ml`. To follow along with the examples, start up Ocaml by typing

```
ocaml -I `camlp4 -where`
```

Load the necessary Camlp4 libraries:

```
# #load "camlp4o.cma";;
# #load "pa_extend.cmo";;
```

(The first # is the Ocaml prompt and should not be typed. The second #

is part of the toplevel directive `#load` and should be typed.) Finally, load `minpoly.ml`:

```
# #use "minpoly.ml";;
```

The Ocaml prompt will no longer be shown.

5.2 First Example

SDP methods will now be applied to minimizing p_1. Type

```
let p1 = polynomial_of_stream [| "x"; "y"; "z" |]
    ( Stream.of_string "9*x^2*y^4+9*x^2*z^4+36*x^2*y^3
    +36*x^2*y^2-48*x*y*z^2+4*y^4+4*z^4-16*y^3+16*y^2" );;
```

The first thing to do is to enumerate all the submonomials of degree at most 3 in 3 variables. Typing

```
List.map (string_of_monom [| "x"; "y"; "z" |])
  (monomials_up_to_deg 3 3);;
```

generates the following list of $20 = \binom{5}{2}$ submonomials: 1, x, y, z, xy, xz, yz, x^2, y^2, z^2, xyz, xy^2, xz^2, x^2y, yz^2, x^2z, y^2z, x^3, y^3, z^3.

Are all these submonomials really necessary? After all, p_1 itself has only 8 terms. It seems that many of the constraints defining the affine space would be homogeneous and could be trivially satisfied by setting the corresponding coefficients to zero.

It is easy to see that the submonomials such as y^2z which don't divide any of the supermonomials can be left out. But in fact a much stronger result was proved by Bruce Reznick in [Rez78]. Monomials correspond to vectors of nonnegative integers (the exponents of each variable). The convex hull of the vectors corresponding to monomials occurring in the polynomial is called the *Newton polytope* of the polynomial. Reznick showed that the only submonomials needed are those whose corresponding vectors lie within the polytope which is *half* the Newton polytope.

Of course, it is possible to determine these vectors through linear programming, but for the case at hand it is easy to do so by inspection. In this case, the vectors are $(2, 4, 0)$, $(2, 0, 4)$, $(2, 3, 0)$, $(2, 2, 0)$, $(1, 1, 2)$, $(0, 4, 0)$, $(0, 0, 4)$, $(0, 3, 0)$, and $(0, 2, 0)$. The point $(1, 1, 2)$ can be expressed as $1/4\,((2, 2, 0) + (0, 2, 0) + (0, 0, 4) + (2, 0, 4))$. Thus the Newton

polytope is the convex hull of $(0, 2, 0)$, $(0, 4, 0)$, $(2, 2, 0)$, $(2, 4, 0)$, $(0, 0, 4)$ and $(2, 0, 4)$. Half of this polytope is the convex hull of $(0, 1, 0)$, $(0, 2, 0)$, $(1, 1, 0)$, $(1, 2, 0)$, $(0, 0, 2)$, and $(1, 0, 2)$. The only points to check are $(0, 1, 1)$ and $(1, 1, 1)$. Since $(0, 1, 1) = {}^1/_2\,((0, 2, 0) + (0, 0, 2))$, and similarly for $(1, 1, 1)$, these eight points are the only ones needed. They correspond to the submonomials y, y^2, xy, xy^2, z^2, xz^2, yz, and xyz.

So type the following:

```
let ( ds, ps ) = poly_prods 3 p1
[ [| 0; 1; 0 |]; [| 0; 2; 0 |];
  [| 1; 1; 0 |]; [| 1; 2; 0 |];
  [| 0; 1; 1 |]; [| 1; 1; 1 |];
  [| 0; 0; 2 |]; [| 1; 0; 2 |] ];;
let ch = open_out "p1_interp.txt";;
pr_interp ch [| "x"; "y"; "z" |] ( ds, ps );;
close_out ch;;
let ch = open_out "p1.dat-s";;
pr_sdpa ch ( ds, ps );;
close_out ch;;
```

Two files have been created. The file `p1_interp.txt` prints the output of `poly_prods` in human-readable format, in two parts. The first part begins with the line

```
There are 9 possible monomial divisors:
```

and proceeds to list them. These are the constant submonomial, plus the same eight submonomials passed as an argument to `poly_prods`, reordered in a canonical way. The constant submonomial is always included even if it is not in the Newton polytope, for reasons that will be explained below. The second part begins with the line

```
There are 34 possible monomial products:
```

and proceeds to list the supermonomials, along with their coefficients in the polynomial `p1` passed as another argument to `poly_prods`, and the various ways they can be factored into submonomials. (The other argument passed to `poly_prods`, 3, is simply the number of variables.) A typical line is

```
-48 of (13) x*y*z^2= (8) z^2* (4) x*y= (9) x*z^2* (2) y
```

which indicates that the thirteenth supermonomial, xyz^2, occurs in `p1` with coefficient -48, and can be expressed in two ways as the product of submonomials.

The file `p1.dat-s` prints the information generated by `poly_prods` in sparse SDPA format. This format is documented at [SDPA]. The problem to solve is of the form

$$\text{maximize } \mathrm{Tr}(F_0 Y) \text{ subject to } \mathrm{Tr}(F_i Y) = c_i \text{ and } Y \text{ is p.s.d.}$$

The first three lines indicate that there are 34 constraints on 9×9 matrices with 1 block each (i.e., no block structure). The next line contains the coefficients c_i of p_1 in the basis of supermonomials. Subsequent lines express the constraints defining the affine space. Since this is a feasibility problem, the matrix F_0 is 0. There are two lines starting with 6:

```
13 1 8 4 1.0
13 1 9 2 1.0
```

This indicates that the $(8, 4)$ entry and the $(9, 2)$ entry of F_6 are each 1. (The F_i's are automatically symmetrized.)

The file `p1.dat-s` can now be submitted to an SDP solver, for example one at NEOS [NEOS]. A feasible point is found, such as

$$M = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 16 & -8 & 0 & 0 & 0 & 0 & 0 & -12 \\
0 & -8 & 4 & 0 & 0 & 0 & 0 & 0 & 6 \\
0 & 0 & 0 & 36 & 18 & 0 & 0 & -12 & 0 \\
0 & 0 & 0 & 18 & 9 & 0 & 0 & -6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -12 & -6 & 0 & 0 & 4 & 0 \\
0 & -12 & 6 & 0 & 0 & 0 & 0 & 0 & 9
\end{pmatrix}$$

Taking the singular value decomposition of this matrix yields the expression

$$\begin{pmatrix} 0 & 0 \\ 0 & -4 \\ 0 & 2 \\ -6 & 0 \\ -3 & 0 \\ 0 & 0 \\ 0 & 0 \\ 2 & 0 \\ 0 & 3 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & -6 & -3 & 0 & 0 & 2 & 0 \\ 0 & -4 & 2 & 0 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$

for M. From this the expression of p_1 as a sum of squares can be read off (recall that the position of each submonomial was printed in `pl-interp.txt`):

$$p_1 = (-6xy - 3xy^2 + 2z^2)^2 + (-4y + 2y^2 + 3xz^2)^2.$$

This demonstrates that 0 is a lower bound for p_1. However, it does not show that it is actually the best lower bound that could be obtained by this method. So type the following:

```
let ch = open_out "plmin.dat-s";;
pr_sdpa_for_min ch ( ds, ps );;
close_out ch;;
```

A new SDPA file, `plmin.dat-s`, has been generated. This time there are only 12 constraints. The matrices are again 9×9. This time, the problem to be solved is of the form

$$\text{minimize } c_1x_1 + c_2x_2 + \cdots + c_mx_m \text{ s.t. } x_1F_1 + x_2F_2 + \cdots + x_nF_m - F_0 \text{ is p.s.d}$$

The affine space is now represented parametrically, by an offset $-F_0$ and basis vectors $F_1, \ldots, F_m$. The goal is to minimize the first parameter, x_1, which corresponds to λ. Thus, the objective function is given by $c_1 = 1$ and $c_i = 0$ for $i > 1$.

The matrices $F_0, \ldots, F_m$ were calculated as follows. As explained above, to express the polynomial as a quadratic form, each term consisting of a real number times a supermonomial must be expressed as a weighted sum of products of submonomials. $-F_0$ is the quadratic form in which all the weight is placed on the first product among the list of possible products. Each F_i

takes the weight away from the first product and places it on one of the other products that could make up a particular supermonomial. For example, in this instance the $(8, 4)$ entry of F_0 is 24.0. Thus, $-48xyz^2$ is expressed as $-2(24(z^2)(xy))$. But the $(8, 4)$ entry of F_4 is -1, and the $(9, 2)$ entry of F_2 is $+1$. Thus adding x_2F_4 corresponds to adding x_2 worth of $(xz^2)y - (z^2)(xy)$, which still leaves a valid expression for the same polynomial.

Now this file can also be submitted to an SDP solver, which yields a minimum value of 0 for λ. This means both that p_1 is a sum of squares, and that $p_1 - \varepsilon$ is *not* a sum of squares for any positive ε.

5.3 Second Motzkin Polynomial

The Newton polytope of the second Motzkin polynomial p_2 is the convex hull of $(4, 2, 2)$, $(2, 4, 2)$, $(2, 2, 4)$, $(2, 2, 2)$, and $(0, 0, 0)$. Now $(2, 2, 2) = {}^1/_4\,((0, 0, 0) + (4, 2, 2) + (2, 4, 2) + (2, 2, 4))$. So the Newton polytope is the convex hull of $(0, 0, 0)$, $(4, 2, 2)$, $(2, 4, 2)$, and $(2, 2, 4)$, and half the Newton polytope is the convex hull of $(0, 0, 0)$, $(2, 1, 1)$, $(1, 2, 1)$, and $(1, 1, 2)$.

In this case it is easy to express the polytope in terms of inequalities. For example, the face not containing the origin is orthogonal to the vector $({}^4/_3, {}^4/_3, {}^4/_3)$, which is contained in it. So we must have $x + y + z \leq 4$. The cross product of $(2, 1, 1)$ and $(1, 2, 1)$ is $(-1, -1, 3)$. The point $(1, 1, 2)$ satisfies $-x - y + 3z = 4$, so we must have $-x - y + 3z \geq 0$. Similarly $-x + 3y - z \geq 0$ and $3x - y - z \geq 0$. Clearly the entire polytope lies in the nonnegative octant. But if, say, $z = 0$, then $-x - y \geq 0$ with $x \geq 0$ and $y \geq 0$. So other than the origin, no lattice points with zero components are present. That $(1, 1, 1)$ is present was already demonstrated. But then the next lattice points are already the vertices of the polytope, so these are all the points to consider. They correspond to the monomials 1, xyz, xyz^2, xy^2z, and x^2yz.

Creating the file `p2.dat-s` as before and submitting it to an SDP solver yields that the problem is infeasible. Submitting the file `p2min.dat-s` also yields that the problem is infeasible. Thus, adding an arbitrary constant to this polynomial still will not make it s.o.s. Indeed, this is a famous example of a polynomial which is p.s.d. but *not* s.o.s. Both facts are proved, for example, in [Rez00].

5.4 Peretz's Polynomial

Finally, it is time to try to minimize Peretz's polynomial p_3. As mentioned, Peretz's polynomial is constrained to lie on the hypercube $0 \leq a, b, c, d \leq 1$. To put this into the unconstrained framework, Peretz suggested making the substitutions $a = x^2/(1+x^2)$, $b = y^2/(1+y^2)$, $c = z^2/(1+z^2)$, $d = w^2/(1+w^2)$. Unfortunately, `minpoly.ml` doesn't yet know how to expand polynomial expressions or make substitutions into them. Instead, this can be done in a computer algebra system and pasted into a file, say `peretz-poly3.txt`. To use this file, type

```
let ch = open_in "peretz-poly3.txt";;
let p3 = polynomial_of_stream [| "x"; "y"; "z"; "w" |]
  (Stream.of_channel ch);;
close_in ch;;
```

The resulting polynomial, of degree 20, has 123 terms. Type

```
#print_length 1024;;
unziprt p3;;
```

to see which monomials occur. The Newton polytope has 108 vertices and 18 facets.

Without considering the Newton polytope, but only the monomial divisibility criterion to reduce the number of submonomials, SDP's were generated to minimize this polynomial. Unfortunately they were beyond the capability of current solvers. The SDP's involved 892×892 matrices. The s.o.s. SDP had 9689 constraints and was killed by the system administrator before it could complete. The minimizing SDP had 388590 constraints and could not even be submitted to a solver.

However, Pablo Parrilo and Karin Gatermann have developed techniques for reducing the size of the SDP which exploit the symmetries of the problem [GP01]. Applying these techniques, Parrilo was able to create an SDP with small blocks, which he solved, finding that p_3 is indeed nonnegative.

Thus the utility of SDP for minimizing polynomials is clear. There is great scope for further research in this area.

6 Acknowledgements

The author would like to express deep gratitude to Professor Bernd Sturmfels for supervising this term project. His continuing guidance, discussions, and encouragement were deeply appreciated. The author also wishes to express heartfelt thanks to Professor Pablo Parrilo for his advice and encouragement, and for sending an unfinished draft of [GP01].

7 References

[And76] George Andrews, *The Theory of Partitions*, 1976.

[BW93] Thomas Becker and Volker Weispfenning, *Gröbner Bases: A Computational Approach to Commutative Algebra*, 1993.

[CLO97] David Cox, John Little, and Donal O'Shea, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, 1997.

[Eis95] David Eisenbud, *Commutative Algebra with a View Toward Algebraic Geometry*, 1995.

[GP01] Karin Gatermann and Pablo Parrilo, "Symmetry groups, semidefinite programs, and sums of squares". Preprint, to appear.

[GPS01] G. M. Greuel, G. Pfister, and H. Schönemann, "SINGULAR 2-0-0. A Computer Algebra System for Polynomial Computations." Centre for Computer Algebra, University of Kaiserslautern (2001). `http://www.singular.uni-kl.de`.

[NEOS] `http://www-neos.mcs.anl.gov`

[Ocaml] `http://www.ocaml.org`

[Par00] Pablo Parrilo, *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*, California Institute of Technology Ph.D thesis, 2000.

[Par01] Pablo Parrilo, "Semidefinite programming relaxations for semialgebraic problems", Preprint, 2001.

[PDS01] Gerhard Pfister, Wolfram Decker, and Hans Schoenemann, `primdec.lib`. A SINGULAR library for computing primary decomposition and radical of ideals (2001).

[PS01] Pablo Parrilo and Bernd Sturmfels, "Minimizing Polynomial Functions. Preprint, 2001. `math.OC/0103170`

[Rez78] Bruce Reznick, "Extremal psd forms with few terms," Duke Math. J. **45** (1978), 363–374.

[Rez00] Bruce Reznick, "Some Concrete Aspects of Hilbert's 17th Problem," *Real Algebraic Geometry and Ordered Structures*, 2000, 251–272.

[SDPA] `ftp://plato.la.asu.edu/pub/sdpa_format.txt`