The Power of Elimination: Solving Smale's 9^{th} Problem

Jean-Louis Lassez
jlassez@coastal.edu
Coastal Carolina University

Abstract

Based on Tarski's meta theorem we argue that elimination methods such as Fourier and Gauss can be used to provide strikingly simple and elementary proofs of major theorems. Most such examples that others and ourselves provided in the past were of known theorems, here we present such a proof for a long standing conjecture: the strong polynomiality of linear programs, which is listed as the ninth problem in Smale's list of Mathematical Problems for the Next Century.

Introduction

Already in 1918 C.I. Lewis [1] stated that "For purpose of application of algebra to ordinary reasoning, elimination is a process more important than solution, since most processes of reasoning take place through the elimination of middle terms". The fact that the elimination method due to Fourier in 1824 [2], and sometimes known as Fourier-Motzkin elimination, can lead to strikingly simple proofs of major theorems is well known. For instance Dantzig and Eaves [3] state that "From it one can derive easily, by trivial algebraic manipulations, the fundamental theorem of linear programming, Farkas lemma, the various theorems of the alternatives and the well known Motzkin Transportation theorem". This property of Fourier elimination has maybe not received as much attention as it deserved because the process is hopelessly exponential. However, as pointed out initially in a series of papers (see[4] for a review), and more recently and forcibly argued in [5] this "theorem proving" property is not accidental, indeed Fourier elimination as well as Gaussian elimination are particular cases of Tarski's elimination for real closed fields [6]. In his remarkable meta theorem, Tarski stated that all theorems of elementary algebra and geometry can be obtained by a single algorithm: quantifier elimination. This has provided the foundation for a major direction in automated theorem proving, as well as in robotics [7] [8] and geographic databases [9] [10] and other applications [11]. However the curse of combinatorial explosion has severely limited the scope of applications, despite major progress in the understanding of its complexity (Renegar [12]).
But if one is interested in properties that do not require an effective execution of the steps of the algorithm, for instance existential properties that are preserved by these steps, elimination can be used with spectacular results. This approach to "non automated" theorem proving has been labeled "Qualitative Theorem Proving" by Chandru and Lassez [5]. However, if one can use elimination to provide spectacularly simpler proofs of

already known theorems having complex proofs using more sophisticated mathematical machinery, as shown in the aforementioned papers, the challenge remains of providing such a proof for an important conjecture of substantial significance.

The conjecture chosen was the strong polynomiality of linear programs. It was in fact suggested to me by Paris Kanellakis, when I showed some surprise that elimination was not used in a more systematic way. His argument was that solving the strong polynomiality problem of linear constraints would be the best way to validate the methodology. It has indeed been listed as the ninth problem in Smale's list of mathematical problems for the 21st century [13], mirroring Hilbert's list of problems at the previous turn of century, over a hundred years ago. The issue is to find a solution to a finite system of linear inequality constraints in a number of steps polynomial with respect to the number of constraints and the number of variables. The first known algorithm to solve such linear programs is Fourier elimination itself, established in 1824, it was rediscovered by de La Vallée Poussin, Motzkin and others and has historical value as well as being at the heart to the proposed methodology. Considered later by Kantorovich it finally led Dantzig to invent his simplex algorithm, whose importance in solving economic problems cannot be overstated. The simplex is an extraordinary algorithm whose worst case complexity is exponential, but nevertheless has excellent practical behavior. Over the past fifty years many mathematicians, and computer scientists have studied its behavior, including celebrated scholars such as Smale, Karp and Kalai, while others have attempted to find a less complex algorithm at the theoretical level, or one that would outperform it at the practical level. I refer to [14] [15] for a discussion of this work, leading to Kachyian and Karmarkar's results who establish that a set of linear constraints can be solved in polynomial time. However two problems remain: at the theoretical level polynomiality has been established for a measure of complexity which takes into account the size of the coefficients. The still open problem is to find a so called strongly polynomial algorithm, whose complexity depends solely on the number of constraints and the number of variables, so that we can have real numbers as coefficients. At the practical level the problem is to find, if possible, an algorithm that will outperform the simplex in a significant way. I address the first problem here, not the second. Still, because of the historical, theoretical, practical and even political importance of that problem and the considerable efforts that have been devoted to it, particularly in the past sixty years, any claim that the strong polynomiality of linear programming has been settled should be considered with the utmost caution. Some will even think that a claim of the existence of an elementary proof based essentially on Fourier's 1824 result should not be considered at all. So I request that the reader consider the following arguments before dropping the paper:

There is a precedent, Hochbaum and Naor [16] proved , in an important particular case that Fourier's algorithm could indeed have a low complexity. This I will also see as an avatar of Tarski's meta theorem. So this power of Fourier elimination should not a priori be dismissed. Also Yamnitsky and Levin [17] gave a remarkably concise proof of a variant of Dikin-Karmarkar's algorithm, so one cannot a priori rule out that extremely difficult problems in that area might still have "simple" solutions. The challenge is to find the right approach.

Another originality in the approach is that I look for an existential proof, not a constructive one, as opposed to the other known and successful approaches. This is due to the nature of the method, and it will be made clear why this relaxation can help considerably.

Finally an simple extension to Fourier elimination plays a most significant role. This extension, due to Lassez and Maher [18] deals with implicit equalities.

In the next section we review briefly various extensions of Fourier elimination: the detection of implicit equalities , constraint elimination and constant elimination. It should be pointed out that only the first extension [18] is needed for the settling of Smale's 9^{th} problem. We present the other two informally because they can be presented at little cost as they are straightforward, they provide a novel and interesting geometric view of elimination, give an insight on convex hulls of affine spaces and unbounded sets, and finally they are the concepts that ultimately led to the discovery of the proof. Without this section it would look like the proof was found purely by accident, and a major purpose of, and motivation for this paper would be defeated.
After these preliminaries the proof is given.

Elimination: Variables, Constraints and Constants.

Variables: Fourier elimination
Let $S=\{Ax\leq b\}$ be a set of inequalities where A is an (n,d) matrix of real numbers, x is a vector of d real variables, $x=(x_1, x_2,\ldots, x_d)$ and b is a real constant vector in R^d.

To eliminate the variable x_k the inequalities are partitioned into:

1. The subset Sp of S which contains all the inequalities where x_k appears with a strictly positive coefficient.
2. The subset Sq of constraints where x_k appears with a strictly negative coefficient.
3. The subset So of constraints where c_p, c_q does not appear.

From Sp and Sq all the possible pairs (c_p, c_q) where c_p is in Sp and c_q is in Sq are formed. From each of these pairs a new inequality not containing x_k is generated by a positive linear combination of the two constraints c_p and c_q.

The set of all inequalities generated this way together with So defines the result of Fourier elimination of the variable x_k.

Note the important case where Sq is empty: no new constraint can be generated, so So is the result. Similarly if Sp is empty.

Fourier's algorithm consists in repeated elimination of the variables and has three possible outcomes:

1. A contradiction as a constraint $0\leq a$ where a is a strictly negative number is generated at some point in the process. S is found to be unsolvable.

2. A non empty set of tautologies $0\leq t$ where t is a positive number, and no contradiction. S is solvable.

3. The empty set of constraints. As no contradiction is generated S is solvable.

Note: if there is a constraint with a strict inequality <, all combinations with other constraints will give strict inequality constraints.

The third case is interesting as it corresponds to a case called strongly solvable [4] because the solvability does not depend on the values of the right hand side constants. If they are all set to 0 the set of constraints defines a full dimensional cone.

The second case is also particularly interesting as shown in [18] because it contains a hidden property of Fourier's algorithm. A linear inequality $C\leq a$ in a set S of inequalities defining a polyhedral set P is called an *implicit equality* if and only if replacing the inequality by an equality in that constraint does not affect the definition of P. The hidden property of Fourier's algorithm is that *S contains implicit equalities if and only if Fourier's algorithm generates tautologies $0\leq 0$.* In that case *all the constraints in S that were used to generate such a tautology are implicit equalities*. One should note that the proof of this result comes partly from a basic property called independence of negative constraints, reported in Lassez and McAloon [19]. This has been used to prove in again a very simple way, non trivial properties of implicit equalities and redundancy [18] [20].

Another simple characterization of implicit equalities established in [18] and that we will use here is the following corollary to Fourier's algorithm hidden property.

Proposition:
A set $\{Ci\leq r_i\}$ i=1,…,n of inequalities is a set of implicit equalities if and only if there is a strictly positive linear combination of the constraints whose coefficients are all 0:

$$\sum \lambda i Ci = [0] \text{ and } \sum \lambda i\, r_i = 0,\ \lambda i > 0 \quad i=1,\ldots,n$$

Note: [0] denotes the linear expression where all coefficients are 0 $\sum \lambda i Ci = 0$ would denote an hyperplane.

Eliminating Constraints:
It is known that variable elimination corresponds to the operation of projection geometrically. Let us make a remark that will initially appear to be absurdly pedantic: Yes elimination of variables can be viewed as a projection, but for the last step.
Indeed when we eliminate the last variable of a non empty polyhedral set we are left with either the empty set or tautologies such as $0\leq 3$. An empty set of constraints or such tautologies define the whole space, so suddenly the dimension goes up, indeed $0x+0y\leq 3$ is satisfied for all real numbers x, y, such a constraint defines R^2. The explanation is simple if we consider projections on an arbitrary hyperplane. The operation of projection of a polyhedral set P consists in fact of two parts: a *cylindrification* , where we construct the set of points which project on P along the direction of the normal vector of the hyperplane, and a *section* where the projection is obtained as the intersection of this cylinder with the hyperplane. This can be achieved easily as a variant of Fourier elimination:

Cylindrification: Let h be a normal vector of the hyperplane H. For each constraint $C\leq r$ in the set defining the polyhedral set, let σ denote the valuation of the left hand side of the constraint, that is the dot product of h with the vector of coefficients of C. Now we add to each constraint's left hand side the monomial σz where z is a new variable. A straightforward adaptation of the proof of correctness of Fourier elimination shows us that we obtain the smallest cylinder containing the polyhedral set P when we eliminate the variable z. Intersecting with H gives us the required projection. Because we now have an equality we can eliminate a variable.
So we can define an arbitrary set of independent vectors spanning the whole space and project along these instead of the initial set of axes, generalizing Fourier elimination to arbitrary sets of coordinates (not necessarily orthogonal).
Therefore strictly speaking Fourier elimination represents a projection only when we set a variable to 0 after we eliminate it from the set: $0x \leq 3$ represents a one dimensional space space, if we add $x=0$ we have the origin.

But what happens if we do not intersect? We will not eliminate a variable at each step, apart from the extra one that we introduced. Consider a single point, it is an affine space of dimension 0. When we cylindrify along the first axis we obtain an affine set of dimension one, a line. As we proceed with cylindrification we generate affine spaces of increasing dimension, leading ultimately to a hyperplane of dimension d-1 and then to the whole space. We draw two conclusions from this: first the variables are not eliminated, it is the constraints that are eliminated, because the whole space is represented by an empty set of constraints (tautologies been redundant). The other conclusion provides an interesting and more intuitive geometrical explanation of Fourier's algorithm hidden property which is a key to the proof we will present in the next section. A set of implicit equalities defines an affine set, so cylindrification will lead to a hyperplane defined by constraints $C\leq r$ and $-C\leq -r$, and then to the tautology $0\leq 0$.

Constant elimination and the construction of convex hulls

Once we have considered variable and constraint elimination, we should indeed consider *constant elimination*. It is in fact even more straightforward than the two others.

Let the conic hull of a polyhedral set P be the set of points on half lines issued from the origin which intersect P.

The construction of the conic hull is trivial using Fourier elimination:
Multiply each right hand side of the constraints defining P by a new variable t, and add the constraint $t>0$ to the set. Indeed a point belongs to the conic hull if and only if its coordinates are a multiple of the coordinates of a point in P. This gives a parametric definition of the conic hull of P. Eliminating the variable t by Fourier elimination gives us a non parametric definition. (but for the origin which is taken away by the < inequality).
Eliminating the variable t is in essence eliminating the right hand side constants, so we call this process *elimination of constants*.

It is straightforward to verify that the convex hull of P with the origin is obtained by adding to the constraints defining the conic hull the constraints from P that have positive constants as their right hand side. Again one has to take into account the fact that the origin itself will be excluded because of the strict inequalities arising from Fourier elimination and the requirement that $t>0$. This is only a minor correction, the main advantage is that P does not need to be bounded. In fact if we combine constant elimination and cylindrification we can construct convex hulls of affine sets and other unbounded polyhedral sets. These remarks were obtained over ten years ago before I retired from the IBM Research Center, I was unable to exploit them in any significant way, so they remained as elementary unpublishable remarks. But the work with Chandru gave me hope that they could lead to elegant proofs of new results.

Solvability, constant elimination and implicit equalities:

It is known that the problem of solvability of a set of linear constraints is strongly polynomially equivalent to the problem of deciding if a cone is reduced to the origin. We will nevertheless outline a method using constant elimination, as it is from this construction that the proof was found, as a side effect. This proof is existential, but one can easily derive efficient methods to construct solutions from such a decision procedure. Our purpose here is to come up with the simplest possible proof of strong polynomiality, not to come up with the lowest possible complexity bound.

A problem with variable elimination is that we cannot control the combinatorial explosion. When we restrict ourselves to strongly solvable sets however we can vary the right hand side of the constraints as we wish since strong solvability is preserved by translation. Deciding strong solvability is linked to the existence of implicit equalities. So we can play with the right hand side, making only one constant negative and all others positive, then constant elimination looses its complexity. We loose information, bounded sets become unbounded, solvable sets become unsolvable, but still a relationship remains regarding the implicit equalities. It is this Achille's heel that will exploit in the next section.

Consider the set S of n constraints
S= {Ci≤ 0} i=1,….,n
We want to relate the existence of implicit equalities in S to the concept of solvability. Let x be a distinguished variable, and without loss of generality we can assume that Cn has been chosen so that x has a positive coefficient in Cn, and that at least one of the other constraints is such that x appears with a negative coefficient. Otherwise we can reduce S because all coefficients of x have the same sign, and start over with another variable.
Let T be the set of n constraints obtained from S in the following way:
If Ci , i<n, has x with a negative coefficient, or if x does not appear in Ci, Ci ≤ 0 is in T.
If Ci has x with a strictly positive coefficient, Ci-Cn ≤ 1 is in T. Finally Cn≤ -1 is in T.

Clearly S, when Cn≤0 is replaced by Cn<0 is the conic hull of T. T is unsolvable if and only if its conic hull is empty, that is if and only if Cn ≤ 0 is an implicit equality in S.

Now if we assume that Cn has been scaled so that the coefficient of x is larger in absolute value than the negative coefficients in the other constraints of S, we can eliminate x in T by Fourier elimination and obtain a new set U of n-1 constraints in a smaller dimension that is solvable if and only if T is solvable. We can take an arbitrary set of constraints as being U and perform the reverse construction by adding a new variable. This gives us a method to tie finding implicit equalities and solvability. There are many others, we will not go into this as a side effect of this construction is what gives us the key to the proof. The side effect comes when we look at which properties are preserved when we set the right hand side of all constraints to 0.
The only prerequisites for that proof are Fourier's algorithm extended by Lassez and Maher [18], and the associated proposition which states that implicit equalities are characterized by positive linear combinations whose coefficient are all nil., and the well known fact that *a constraint is redundant in a set if and only if it is a positive linear combination of the other constraints in the set,* with eventually the addition of an arbitrary positive constant to the right hand side. (We will not consider this addition here as we will deal strictly with cones, which are represented by constraints having a right hand side constant equal to 0). We also note that finding implicit equalities in a cone is (strongly) polynomially equivalent to solvability. As we mentioned before this is well known and can be done a number of ways, including the one we sketched using constant elimination. We also can add one dimension and build the convex hull of the polyhedral

set with a given point in that new dimension, it is less expensive than constant elimination, or we can go to the dual. This problem will not be addressed here.

Strong Polynomiality of linear programs: some inequalities are more equal than others

Theorem
Solving a linear program is strongly polynomial

Proof

We want to find implicit equalities in a cone $S=\{Ci\leq 0\}$ $i=1,....,n$

We build $T=\{Ci-Cn\leq 0, Cn\leq 0\}$ for $i=1,...,n-1$.

Now if a subset of S has all implicit equalities, we have for that subset a positive linear combination:

$\sum\lambda iCi=[0\ ]$ by the proposition above, and then $\sum\lambda i\ (Ci-Cn)+(\sum\lambda i)Cn=[0]$, so T has implicit equalities and $Cn\leq 0$ is an implicit equality in T.

Conversely if there are implicit equalities in T, we have a positive linear combination:
$\sum\mu i\ (Ci-Cn)+\mu Cn=[0]$

If $\mu=\sum\mu i$ then S has implicit equalities, if $\mu<\sum\mu i$ then $Cn\leq 0$ is redundant in S, as Cn is then equal to a positive linear combination of constraints in S.
If $\mu>\sum\mu i$ then S has implicit equalities and Cn is one of them.

Now, we can scale Cn by multiplying by an arbitrary positive number, so that the coefficient of a chosen variable x in Cn is in absolute value strictly greater than any of the coefficients of x in the other constraints in S. As a consequence T will be such that the coefficient of x in Cn is of opposite sign to the coefficient of x in all other constraints of T. We can therefore eliminate the variable x, resulting in a cone U defined by n-1 constraints.
In that process, if there are implicit equalities in T then there will be implicit equalities in U and/or the constraint $0\leq 0$ is generated [18].

We can repeat the process by starting again with U. The implicit equalities propagate until constraints 0≤0 are generated.

As the number of constraints and the number of variables decreases by 1 when going from S to T to U, the set of constraints becomes smaller and, if there are implicit equalities in S, the constraint 0≤0 will be generated at some stage when constructing U. Each time it is generated we know that the preceding T has implicit equalities, and that the preceding S has either implicit equalities or has its last constraint redundant.

Complexity (rough bound)
We initially build at most n blocks of constraints S-U-T each one obtained from the preceding. It means that we have at most k of order $0(n^2)$ constraints altogether in these blocks (we assume without loss of generality that the number of variables is less than the number of constraints). When this operation is completed, the algorithm terminates as no implicit equality has been detected, S will not contain any. If we have found implicit equalities in a block, we can propagate them backwards by keeping track of the coefficients in the successive linear combinations [18] and the propagation will either reach S, and we will detect implicit equalities or find a redundancy in S, or stop in one of the intermediate blocks, telling us that a constraint is redundant in that block. We can then repeat the process from there with at most k-1 constraints altogether. Consequently we will have generated at most $0(n^4)$ constraints by simple operations of variable elimination.
This cost is more than the cost of transformation from a solvability problem to the problem of detecting implicit equalities in a cone. This process gives us an existential proof, not a constructive one. To effectively obtain a solution there are many ways, that are strongly polynomial: we can test if a constraint is a facet for instance, it will be simpler if we assume boundedness or full dimensionality. In that case it might be more practical to go directly to the computation of the extreme point optimizing the objective function by using binary type searches in directions close to the direction defined by the objective function. There is also the issue of what can be computed in parallel. In other words a full investigation is needed in order to find what is the best complexity that this method can lead to. It is beyond the scope of that paper.

Acknowledgments: many, still in progress

References

1. Lewis C.I. (1918), A Survey of Symbolic Logic. Berkeley *University of California Press.*

2. Fourier, J-B.(1824), reported in: Analyse des travaux de l'Académie Royale des Sciences, pendant l'année 1824, Partie Mathématique.

Histoire de l'Académie Royale des Sciences de l'Institut de France 7 (1827) xlvii-lv (Partial English translation in : D.A. Kohler, Translation of a Report by Fourier on his work on Linear Inequalities, *Opsearch* 10).

3. Dantzig, G. B. and Eaves B.C. (1973), Fourier-Motzkin Elimination and its Dual, *Journal of Combinatorial Theory* Ser. A, 14

4. Lassez, J-L. (1991), "From LP to LP: Programming with Constraints," in *Database Programming Languages*, P. Kanellakis and J.W. Schmidt Ed., *Morgan-Kaufmann.*

5. Chandru V., Lassez J-L. (February 2004), Qualitative Theorem Proving in Linear Constraints. In *Verification, Theory and Practice*, *Springer Verlag Lecture Notes in Computer Science.* Volume in Honor of Zohar Manna, N. Dershowitz Ed.

6. Van den Vries, L. (March 1988), Alfred Tarski's *Elimination Theory for Closed Fields*, The Journal of Symbolic Logic, vol 53 n.1.

7. Latombe J-C.(1991), Robot Motion Planning ,*Kluwer Academic Publishers.*

8. Schwartz, J.T. and Sharir M. (1989), A Survey of Motion Planning and Related Geometric Reasoning, in *Geometric Reasoning* , D. Kapur and J.L. Mundy ed., *MIT Press.*

9. Brodsky, A., Lassez, C., Lassez, J-L., Maher, M.(july 1999), "Separability of Polyhedra for Optimal Filtering of Spatial and Constraint Data", *Journal of Automated Reasoning*, *Kluwer.* (preliminary version in the Proceedings of the ACM PODS Conference 1995)

10. Revesz, P. (2002) *Introduction to Constraint Databases* Springer.

11. Huynh, T., Joskowicz, L., Lassez, C., Lassez, J-L.(December1991), "Practical Tools to Reason about Linear Constraints," *Fundamenta Informaticae*, vol. 15, no. 3-4.

12. Renegar J. (1991), Recent Progress on the Complexity of the Decision Problem for the Reals. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science* 6.

13. Smale, S..(1998) Mathematical problems for the next century. *Mathematical Intelligencer,* 20:7-15.

14. Papadimitriou C. H., Steiglitz K. (1998), Combinatorial Optimization: Algorithms and Complexity *Dover.*

15. Bertsimas D., Tsitsiklis(1997), J.N., Introduction to linear optimization *Athena Scientific*

16. D. S. Hochbaum and J. Naor (December 1994). Simple and fast algorithms for linear and integer programs with two variables per inequality. *SIAM Journal on Computing*, 23(6):1179-1192.

17. Yamnitsky, B. and Levin, L. (1982) An Old Linear Programming Algorithm runs in Polynomial Time. *IEEE Symposium on the Foundations of Computer Science*

18. Lassez, J-L., Maher, M.(1992), “On Fourier's Algorithm for Linear Constraints,” vol. 9, *Journal of Automated Reasoning*.

19. Lassez, J-L., McAloon, K. (January 1992), “A Canonical Form for Generalized Linear Constraints,” *Journal of Symbolic Computation*.

20. Lassez, J-L., Huynh, T., McAloon, K., (1993) “Simplification and Elimination of Redundant Linear Arithmetic Constraints,” in *Constraint Logic Programming, Selected Research*, Benhamou F. and Colmerauer A. Eds. MIT Press.