Speeding up Cylindrical Algebraic Decomposition by Gröbner Bases

D.J. Wilson, R.J. Bradford & J.H. Davenport

Department of Computer Science, University of Bath
Bath BA2 7AY, U.K.
{D.J.Wilson, R.J.Bradford, J.H.Davenport}@bath.ac.uk

Abstract. Gröbner Bases [Buc70] and Cylindrical Algebraic Decomposition [Col75,CMMXY09] are generally thought of as two, rather different, methods of looking at systems of equations and, in the case of Cylindrical Algebraic Decomposition, inequalities. However, even for a mixed system of equalities and inequalities, it is possible to apply Gröbner bases to the (conjoined) equalities before invoking CAD. We see that this is, quite often but not always, a beneficial preconditioning of the CAD problem.
It is also possible to precondition the (conjoined) inequalities with respect to the equalities, and this can also be useful in many cases.

The examples used in this paper are available in [Wil12]. This work was partially supported by the U.K.'s EPSRC under grant number EP/J003247/1.

1 Introduction

Solving systems of equations, or equations and inequations ($\neq$)/inequalities ($>, <$) is an old subject. Deciding the truth of, or more generally eliminating quantifiers from, quantified Boolean combinations of such statements, is more recent [Tar51]. We can distinguish many families of methods, even if we restrict attention to the real numbers, or possibly the complex numbers.

$=_G$ The method of Gröbner bases. Here the input is a set $S = \{s_1, \ldots, s_k\}$ of polynomials in some polynomial ring $k[x_1, \ldots, x_n]$ equipped with a total order[1] $\prec$ on the monomials, and the output is a set $G = \{p_1, \ldots, p_l\}$ which is equivalent, in the sense that it generates the same ideal, i.e. $(G) = (S)$, and is simpler, or "surprise-free", in that the leading monomial with respect to $\prec$ (denoted $\mathrm{lm}_\prec$) behaviour is explicit, $(\mathrm{lm}_\prec(G)) = (\mathrm{lm}_\prec((G)))$. Then the solutions of G are those of S, i.e.

$$\{\mathbf{x} : p_1(\mathbf{x}) = 0 \wedge p_2(\mathbf{x}) = 0 \wedge \cdots \wedge p_l(\mathbf{x}) = 0\}\,. \tag{1}$$

[1] We have concentrated on purely lexicographical orders, since these seem to be the most useful to us.

$=_{\Delta}$ The method of triangular decomposition via regular chains [ALMM99,MM05]. Here the output is a set of regular chains of polynomials

$$\{(p_{1,1}, p_{1,2}, \ldots), (p_{2,1}, p_{2,2}, \ldots), \ldots\}, \tag{2}$$

and the solution is the union of the set of regular zeros of these regular chains:

$$\left\{\mathbf{x} : p_{1,1}(\mathbf{x}) = p_{1,2}(\mathbf{x}) = \cdots = 0 \wedge \left(\prod_i \mathrm{init}(p_{1,i})\right)(\mathbf{x}) \neq 0\right\} \cup \cdots . \tag{3}$$

$<_{\mathrm{Col}}$ The method of Cylindrical (semi-)Algebraic Decomposition for real closed fields, computed via repeated projection to $\mathbf{R}^1$ and repeated lifting [Col75, and many improvements].

$=_{\mathrm{Col}}$ The previous case restricted to equality.

$<_{\Delta\mathbf{R}}$ The method of Cylindrical (semi-)Algebraic Decomposition for real closed fields via triangular decomposition [CMMXY09].

$\neq_{\Delta\mathbf{C}}$ The method of Cylindrical Decomposition over the complexes via triangular decomposition, which was introduced in [CMMXY09] as a stepping-stone to the previous method, but which probably has independent interest.

$\nexists_{\mathrm{CH}}$ Quantifier Elimination by partial (i.e. taking account of the Boolean structure and quantifier structure) Cylindrical Algebraic Decomposition [CH91].

Others such as Weispfenning's Virtual Term Substitution [Bro05, is a readable introduction], or Tarski's original method [Tar51].

Conversely instead of asking for solutions $\mathbf{x}$ to $\exists\mathbf{x} f_1(\mathbf{x}) \geq 0 \wedge \cdots$, we may use a Positivstellensatz to show that no such $\mathbf{x}$ exist, as in [PQR09]. We do not discuss this direction further here.

It should be noted that both $<_{\mathrm{Col}}$ and $<_{\Delta\mathbf{R}}/\neq_{\Delta\mathbf{C}}$ (but not $\nexists_{\mathrm{CH}}$) have the drawback that the Cylindrical Algebraic Decomposition produces decompositions for, not only the question posed, e.g. $\forall y \exists z p(x,y,z) = 0 \wedge q(x,y,z) = 0 \wedge r(x,y,z) > 0$, but also *all* other questions involving the same polynomials, provided the quantifiers are over variables in the same order, e.g. $\exists y \forall z p(x,y,z) < 0 \vee (q(x,y,z > 0 \wedge r(x,y,z) = 0)$.

This paper asks the question: "can these methods usefully be combined?" The combinations we are thinking about are those of conjunction: Can the fact that B is in the context of $a_1 = 0 \wedge \cdots \wedge a_k = 0 \wedge B$ be used to simplify B? In particular, we look at the use of Gröbner base methods to simplify the equalities in the conjunction *and* to simplify the inequalities in the light of the equalities.

Technical Note: all computations ($=_G$, $<_{\Delta\mathbf{R}}$ and $\neq_{\Delta\mathbf{C}}$) were performed in Maple 16β on a 3.1GHz Intel processor, except for the $<_{\mathrm{Col}}$, $=_{\mathrm{Col}}$ and $\nexists_{\mathrm{CH}}$ ones, which were performed on a 2.83GHz Intel processor with QEPCAD B version 1.65 [Bro03]. Times for a hybrid calculation, e.g. $=_G/<_{\mathrm{Col}}$, are either quoted as the total time or a decomposition $a + b = c$ where a is the time (in milliseconds) for $=_G$, b for $<_{\mathrm{Col}}$, and c is the sum. We have run QEPCAD in three modes:

1. on the problem as given in [BH91], implementing $\nexists_{\mathrm{CH}}$;

2. as above but with the `full-cad` option to ignore the Boolean structure of the expression;
3. with no quantifiers stated, and the `full-cad` option, implementing $<_{\text{Col}}$.

2 Examples in this paper

2.1 [BH91]

This paper has a variety of examples for $\nexists_{\text{CH}}$, all of a form to which $=_G$ is applicable.

2.2 [CMMXY09]

This paper has a variety of examples for $<_{\mathbf{\Delta R}}$. We chose some of those to which $=_G$ is applicable.

2.3 Two Spheres and A Cylinder

Let the following be spheres in $\mathbf{R}^3$:

$$\begin{aligned} S_1 &: \quad (x-1)^2 + y^2 + z^2 - 3; \\ S_2 &: \quad (x+1)^2 + y^2 + z^2 - 3; \\ S_3 &: \quad (x-1)^2 + \left(y - \tfrac{1}{2}\right)^2 + z^2 - 3; \\ S_4 &: \quad (x+1)^2 + \left(y + \tfrac{2}{3}\right)^2 + \left(z + \tfrac{3}{4}\right)^2 - 3. \end{aligned}$$

Denote the infinite cylinder centred on the z-axis with radius 1 by C, so that the equation defining the cylinder is:

$$C : \quad x^2 + y^2 - 1.$$

Now we investigate intersecting pairs of spheres (roughly increasing in CAD 'difficulty') under conditions based on the cylinder. We assume the spheres' equation will always be required to equal 0 but make no assumptions on the condition on the cylinder. That is, we wish to solve the problem:

$$S_i = 0 \ \wedge \ S_{i+1} = 0 \ \wedge \ C * 0 \qquad * \in \{=, \neq, <, >, \leq, \geq\}, i = 1, 2, 3. \tag{4}$$

We use the underlying variable ordering[2] (z, y, x).

[2] This is the QEPCAD notation, meaning that we will project from (z, y, x)–space to (z, y)–space to (z)–space. We therefore end up with polynomials in z alone, so this is equivalent to a purely lexicographical Gröbner base with $z \prec y \prec x$, i.e. `plex([z,y,x])` in Maple: $=_{GC}$ is used to indicate Gröbner bases with this (compatible) ordering. The CAD package in Maple [CMMXY09] requires `PolynomialRing([x,y,z])` to achieve the same effect as QEPCAD's (z, y, x). $=_{GR}$ denotes the reverse `plex` order.

3 Prior Art

Needless to say, we are not the first to have had this idea.

3.1 Buchberger–Hong

[BH91] considers the case of $=_G$ ([BGK85] re-implemented in C) applied to $<_{\mathrm{Col}}$ (an early version of [CH91] re-implemented in C), i.e., rather than computing a CAD for the zeros of a system of equations E (i.e. $e_1 = 0 \wedge e_2 = 0 \wedge \cdots$) and inequalities F, compute it for G, a (purely lexicographical) Gröbner base for E, and F. They generally found a very substantial speed-up in the total computation time, e.g. "Solotareff A"[3]

$$\exists x \exists y \;\; 3x^2 - 2x - a = x^3 - x^2 - ax - 2b + a - 2 = \tag{5}$$

$$3y^2 - 2y - a = y^3 - y^2 - ay - 2b + a - 2 = 0 \wedge \tag{6}$$

$$4a \in [1, 7] \wedge 4b \in [-3, 3] \wedge x \in [-1, 0] \wedge y \in [0, 1] \tag{7}$$

(with the variable ordering (b, a, x, y)) took them 11500 ms for $\nexists_{\mathrm{CH}}$, but 717 for $=_G$, and 117 for $\nexists_{\mathrm{CH}}$ applied to the result, a total of 834 ms, or a 13-fold speed-up. "Solotareff B" is the same problem but with (a, b, x, y) as the variable ordering, and here the $\nexists_{\mathrm{CH}}$ time was again greatly reduced, but the $=_G$ time was excessive. Of course, there have been substantial improvements in the implementation of all these algorithms since [BH91] was published, and Table 2 shows that the $=_G$ time is now less than 1/3 of the $\nexists_{\mathrm{CH}}$ time. We choose rather to focus on the number of cells generated, which is closely connected to the $=_{\mathrm{Col}}$ time, and also affects the time taken to make use of the output. The cell counts are shown in Table 1.

Table 1. Cell counts for Solotareff

	Ordering A		Ordering B	
	$<_{\mathrm{Col}}$	$=_G/<_{\mathrm{Col}}$	$<_{\mathrm{Col}}$	$=_G/<_{\mathrm{Col}}$
(5–7) Partial	153	63	375	41
Full	349	625	1063	237
(5–6) Partial	29	15	97	17
Full	29	33	97	17

More reruns of [BH91] are given in Table 2. We see that, with today's technology, the conclusion of [BH91], viz. that $=_G$ generally improves $\nexists_{\mathrm{CH}}$ for the class of problems to which it is applicable, is still generally valid, but the details differ: notably the Gröbner base time is generally insignificant today.

[3] There are various problems labelled "Solotareff": for a description of this class see [Wil12] and the links therein.

Table 2. [BH91] with today's technology

	$\nexists_{\mathrm{CH}}$		$=_G/\nexists_{\mathrm{CH}}$		$\nexists_{\mathrm{CH}}$/`full-cad`		$=_G/\nexists_{\mathrm{CH}}$/`full-cad`	
	Time	**Cells**	**Time**	**Cells**	**Time**	**Cells**	**Time**	**Cells**
I A	190	503	22+72=	94 23	188	503	22+73=	95 51
I B	199	369	21+74=	95 17	191	369	21+75=	96 33
R A	85	1	24+73=	97 1	86	1	24+71=	95 1
R B	129	1	24+72=	96 1	125	1	24+72=	96 1
E A	297	621	25+134=	159 621	576	11139	25+394=	419 11139
E B	Error	?	50+?=	Error ?	Error	?	50+?=	Error ?
S A	89	153	22+72=	94 63	199	349	22+185=	207 625
S B	113	375	23+75=	98 41	228	1063	23+180=	203 237
C A	133	19	42+?=	Error ?	235	19	42+?=	Error ?
C B	Error	?	132+?=	Error ?	Error	?	132+?=	Error ?

Table 3. [BH91] Examples for full CADs

	$=_{\mathrm{Col}}$		$=_G/=_{\mathrm{Col}}$		$<_{\Delta\mathbf{R}}$		$=_G/<_{\Delta\mathbf{R}}$	
	Time	**Cells**	**Time**	**Cells**	**Time**	**Cells**	**Time**	**Cells**
I A	236	3723	22+77=	99 273	29426	3763	2470	273
I B	212	3001	21+76=	97 189	36262	2795	1482	189
R A	150	2101	24+86=	110 105	17355	1267	570	165
R B	21091	7119	24+80=	104 141	356670	7119	470	141
E A*	7390	114541	25+3189=	3214 53559	262623	28557	62496	14439
E B*	Error	?	50+?=	Error ?	$> 1000s$	?	$> 1000s$	?
S A*	115	1751	22+82=	104 297	16014	1751	2025	297
S B*	253	6091	23+82=	105 243	43439	6091	1647	243
C A*	820	8387	42+?=	Error ?	216028	7895	$> 1000s$	?
C B*	Error	?	132+?=	Error ?	$> 1000s$	?	$> 1000s$	?

* indicates that the linear inequalities have been omitted in this version.

There is one point which is not explicit in [BH91]. As the computation of Gröbner bases in one variable is just equivalent to Euclid's algorithm, i.e. Gaussian elimination in Sylvester's matrix, Gröbner base computations which are not genuinely multi-variate do not affect the set of resultants etc. generated in $<_{\mathrm{Col}}$, and hence are of limited use in the projection phase. They might still reduce the work done in the lifting phase, of course.

Table 3 re-runs the examples of [BH91], but asking for complete cylindrical algebraic decompositions, and hence we can compare $<_{\mathrm{Col}}$ with $<_{\Delta\mathbf{R}}$ legitimately. Given that the algorithms are fundamentally different, the similarities in cell counts are striking. The differences in cell counts (where present) reflect differences in the cylindrical algebraic decompositions for the same input problem.

3.2 Phisanbut

Phisanbut [Phi11], considering branch cuts in the complex plane, observed that $g = 0 \wedge f > 0$ could be reduced to $g = 0 \wedge \text{prem}(f, g) > 0$ under suitable conditions, where prem denotes the pseudo-remainder operation. More precisely, if f and g are regarded as polynomials in the main variable x, of degrees d and e respectively, then $\text{prem}(f, g) = \text{rem}(c^{d-e+1} f, g)$, where c is the leading coefficients of g. When $g = 0$ and $c > 0$, or when $d - e + 1$ is even, $\text{prem}(f, g)$ has the same sign as f. Unfortunately c might have variable sign, and $d - e + 1$ might be odd, so define $\text{pprecond}(f, g) = \text{rem}(c^{(d-e+1)^*} f, g)$, where n^* is n if n is even and $n + 1$ if n is odd. Maple also defines $\text{sprem}(f, g) = \text{rem}(c^m f, g)$, where m is the smallest integer such that the division is exact, and by analogy we have $\text{sprecond}(f, g) = \text{rem}(c^{m^*} f, g)$. Note that $\text{sprecond}(f, g) = \text{pprecond}(f, g)$ or a strict divisor of it, i.e. sprecond is never worse. She generally, but not always, saw [Phi11, Tables 8.13, 8.14] a significant decrease in the number of cells, and the time taken to compute sprecond was minimal.

4 Further developments

4.1 $=_G$ with $<_{\Delta \mathbf{R}}$

It would seem natural to apply $=_G$ to $<_{\Delta \mathbf{R}}$, as [BH91] did to $\not\exists_{\text{CH}}$. The results are in Table 3, and show a speed-up in all instances except the Collision problems. We also note the substantial speed advantage enjoyed by $<_{\text{Col}}$, and this is a subject for further study.

4.2 $=_G$ with $\neq_{\Delta \mathbf{C}}$

We can also mix $=_G$ with $\neq_{\Delta \mathbf{C}}$, and these results are shown in Table 4, which also compares $\neq_{\Delta \mathbf{C}}$ with $<_{\Delta \mathbf{R}}$. $<_{\Delta \mathbf{R}}$ involves doing $\neq_{\Delta \mathbf{C}}$ first, and then running the `MakeSemiAlgebraic` algorithm from [CMMXY09]. For these examples, the `MakeSemiAlgebraic` step is the most expensive initially, but often not after we apply $=_G$.

4.3 $=_G$ with inequalities in $<_{\Delta \mathbf{R}}$

Having reduced the equalities to a Gröbner base G, it is now possible to reduce the inequalities by G, since adding/subtracting a multiple of an element of G is adding/subtracting 0. We can reduce with respect to the main variable, denoted $=_G/\rightarrow_x^G$, with respect to secondary variables, denoted $=_G/\rightarrow_y^G$, or with respect to all variables (Maple's `NormalForm`), denoted $=_G/\overset{*}{\rightarrow}^G$. If we compare tables 6 and 7 we see that the number of cells produced is the same across the two methods.

Table 4. Timings for [BH91] Examples: $<_{\Delta\mathbf{R}}/\neq_{\Delta\mathbf{C}}$

	$\neq_{\Delta\mathbf{C}}$ **Time**	$<_{\Delta\mathbf{R}}$ **Time**	**Ratio**	$=_G/\neq_{\Delta\mathbf{C}}$ **Time**	$=_G/<_{\Delta\mathbf{R}}$ **Time**	**Ratio**
Intersection A	5691	29426	4.17	1168	2470	1.11
Intersection B	5584	36262	5.49	886	1482	0.67
Random A	4614	17355	2.76	310	570	0.84
Random B	67343	356670	4.30	318	470	0.48
Ellipse A*	85425	262623	2.07	27916	62496	1.24
Ellipse B*	441245	$> 1000s$	-	$> 1000s$	$> 1000s$	-
Solotareff A*	6666	16014	1.40	1760	2025	0.15
Solotareff B*	9536	43439	3.56	1404	1647	0.17
Collision A*	41085	216028	4.26	$> 1000s$	$> 1000s$	-
Collision B*	$> 1000s$	$> 1000s$	-	$> 1000s$	$> 1000s$	-

"Ratio" = $(<_{\Delta\mathbf{R}} - \neq_{\Delta\mathbf{C}})/\neq_{\Delta\mathbf{C}}$, i.e. the relative cost of `MakeSemiAlgebraic`.

Table 5. Examples from [CMMXY09]

	$<_{\Delta\mathbf{R}}$		$=_G/<_{\Delta\mathbf{R}}$		Ratio	
	Time	**Cells**	**Time**	**Cells**	**Time**	**Cells**
Cyclic–3	3136	381	20 + 245 = 265	21	11.83	18.14
Cyclic–4	$> 1000s$	?	64 + 5813 = 5877	621	?	?
2	2249	895	22 + 1845 = 1867	579	1.20	1.55
4	3225	421	24 + 19738 = 19762	1481	0.16	0.28
6	363	41	20 + 918 = 938	89	0.39	0.46
7	3667	895	26 + 6537 = 6563	1211	0.56	0.74
8	3216	365	21 + 174 = 195	51	16.49	7.16
13	14342	4949	18 + 220 = 238	81	60.26	61.10
14	334860	27551	21 + 971 = 992	423	337.56	65.13

Table 6. Spheres and Cylinders: $<_{\Delta\mathbf{R}}$

	$<_{\Delta\mathbf{R}}$		$=_G/<_{\Delta\mathbf{R}}$		$=_G/\rightarrow_y^G/<_{\Delta\mathbf{R}}$		$=_G/\rightarrow_x^G/<_{\Delta\mathbf{R}}$		$=_G/\overset{*}{\rightarrow}^G/<_{\Delta\mathbf{R}}$	
	Time	**Cells**	**Time**	**Cells**	**Time**	**Cells**	**Time**	**Cells**	**Time**	**Cells**
S_1, S_2, C	9830	1073	1057	267	394	91	528	183	298	99
S_2, S_3, C	187048	12097	5880	1299	3171	627	2149	517	506	213
S_3, S_4, C	247458	11957	8164	1359	9177	1123	5476	881	590	213

Table 7. Spheres and Cylinders: $<_{\text{Col}}$

	$<_{\text{Col}}$		$=_G/<_{\text{Col}}$		$=_G/\overset{*}{\rightarrow}^G/<_{\text{Col}}$	
	Time	**Cells**	**Time**	**Cells**	**Time**	**Cells**
S_1, S_2, C	30	1073	23 + 8 = 31	267	24 + 4 = 28	99
S_2, S_3, C	763	12097	27 + 36 = 63	1299	28 + 13 = 41	213
S_3, S_4, C	1760	11957	28 + 37 = 65	1359	29 + 14 = 43	213

5 Choice of Method

Suppose we are given a problem, which we may formulate as

$$\text{quantified variables } e_1 = 0 \wedge \cdots \wedge e_k = 0 \wedge B(f_1, \ldots, f_l), \tag{8}$$

where B is a Boolean combination of conditions $= 0, \neq 0, < 0$ etc. on some polynomials f_j, then we may be able, by applying Gröbner techniques to the e_j , producing $e_j^{(i)}$, and then reducing the f_j, to produce various alternative formulations

$$\text{quantified variables } e_1^{(i)} = 0 \wedge \cdots \wedge e_{k^{(i)}}^{(i)} = 0 \wedge B(f_1^{(i)}, \ldots, f_l^{(i)}), \tag{$8^{(i)}$}$$

and each of these may have several variable orderings compatible with the constraints implied by the quantification (if any). Which should we choose? Of course, in the presence of arbitrary parallelism, we can start them all, and accept the first to finish, but we may wish to be less extravagant.

In the contexts of $\nexists_{\mathrm{CH}}$ (strictly speaking, the REDLOG implementation), and where the only choice was in the variable order, this question was considered by [DSS04]. Retrospectively, there are two measures for the difficulty of a CAD computation: the time taken and the number of cells produced. For a given $\nexists_{\mathrm{CH}}$ problem, they observed that two are usually correlated for different formulations, and we observe the same here for $<_{\Delta\mathbf{R}}$ — see our tables. However, we would like a measure that could be calculated in advance, rather than retrospectively.

The processes of [Col75,CH91] starts with a set A_n of polynomials in n (ordered) variables $x_1, \ldots, x_n$, and

1. repeatedly project A_i into A_{i-1} in one fewer variable, until A_1 has only one variable,
* (denote the set $\{A_n, \ldots, A_1\}$ by $A(x_1, \ldots, x_n)$)
2. isolate the roots of these polynomials to get a decomposition of $\mathbf{R}^1$,
3. repeatedly lift the decomposition until we get a (partial for [CH91]) cylindrical algebraic decomposition of $\mathbf{R}^n$.

The third step is, both theoretically and practically, by far the most expensive. Hence the question arises: what can we measure at the end of step 1, i.e. depending on A only, which is well-correlated with the final cost? Three things come to mind.

$\texttt{card}(A(x_1, \ldots, x_n)) = \sum_{i=1}^{n} |A_i|$.
$\texttt{td}(A(x_1, \ldots, x_n)) = \sum_{i=1}^{n} \sum_{p_{i,j} \in A_i} \texttt{td}(p_{i,j})$ where `td` denotes total degree.
$\texttt{sotd}(A(x_1, \ldots, x_n)) = \sum_{i=1}^{n} \sum_{p_{i,j} \in A_i} \sum_{\text{monomials } m \text{ of } p_{i,j}} \texttt{td}(m)$.

[DSS04] discard `td`, observing that `td` and `sotd` are highly correlated and `sotd` "has the advantage of favouring sparse polynomials". They then observe that $\texttt{sotd}(A(x_1, \ldots, x_n))$ is significantly more correlated with the retrospective measures for any given problem than `card`. This gives a first algorithm for deciding how to project: for all admissible (i.e. compatible with the quantifier structure, if

any) permutations π of $(x_1, \ldots, x_n)$, compute $A(x_{\pi(1)}, \ldots, x_{\pi(n)})$, and choose the one with the least `sotd` value. The drawback of this is that it requires potentially $(n-1)n!$ projection operations. They show that (at least on their examples) this always produces a good projection order, and frequently the optimal.

Table 8. Spheres and Cylinders: $<_{\Delta\mathbf{R}}$ — choice of orderings

		$<_{\Delta\mathbf{R}}$		$=_G/<_{\Delta\mathbf{R}}$		$=_G/\overset{*}{\to}^G/<_{\Delta\mathbf{R}}$	
		Time	**Cells**	**Time**	**Cells**	**Time**	**Cells**
S_1, S_2, C	C	8654	1073	905	267	270	99
	R			902	267	453	183
S_2, S_3, C	C	189202	12097	5911	1299	499	213
	R			18941	2639	5307	859
S_3, S_4, C	C	248340	11957	8159	1359	580	213
	R			160171	9091	196714	11203

They therefore propose a *greedy algorithm*, where for all permissible choices of the first variable to be projected, we compute $\texttt{sotd}(A_{n-1})$, and choose the variable which gives the least value. Having fixed this as the first variable to project, for all permissible choices of the second variable to be projected, we compute $\texttt{sotd}(A_{n-2})$, and choose the variable which gives the least value, and so on. Hence, assuming all projection orders are possible, the **number** of projections done is $n + (n-1) + \cdots = O(n^2)$ rather than $n!$. It is currently an open question whether the **cost** of projections behaves similarly.

We proposed taking this idea still further, and suggested that, for several different formulations $A_n, B_n, \ldots$ of a problem, we should compute $\texttt{sotd}(A_n)$, $\texttt{sotd}(B_n), \ldots$ and take the formulation that yields the lowest `sotd`. We observed, however, that neither `td` nor `sotd` are good predictors in Table 11, despite seeming useful in Table 10.

6 The metric `TNoI`

When we apply Gröbner techniques to a set of equations (either by calculating a basis or a normal form) we are, in some sense, trying to simplify the set of equations. In a zero-dimensional ideal, as shown in the Gianni-Kalkbrener Theorem [Gia89,Kal89], a purely lexicographic Gröbner basis has a very distinct, triangular structure.

With this in mind we thought it may be of some use to consider the number of variables present in a certain problem and so defined the following quantity, `TNoI`, which stand for "Total Number of Indeterminates":

$$\texttt{TNoI}(F) = \sum_{f \in F} \texttt{NoI}(f), \tag{9}$$

where $\texttt{NoI}(f)$ is the number of indeterminates present in a polynomial f.

Table 9. [BH91]: effect of orderings $=_{GC}$ versus $=_{GR}$

		$<_{\Delta\mathbf{R}}$ **Time**	**Cells**	$=_G/<_{\Delta\mathbf{R}}$ **Time**	**Cells**
Intersection A	C	29426	3763	2470	273
	R			$> 1000s$	?
Intersection B	C	36262	2795	1482	189
	R			$> 1000s$	?
Random A	C	17355	1219	570	165
	R			$> 1000s$	?
Random B	C	356670	7119	470	141
	R			$> 1000s$	?
Ellipse A*	C	262623	28557	62496	14439
	R			271726	29939
Ellipse B*	C	$> 1000s$	?	$> 1000s$	?
	R			$> 1000s$	?
Solotareff A*	C	16014	1751	2025	297
	R			$> 1000s$	?
Solotareff B*	C	43439	6091	1647	243
	R			$> 1000s$	?
Collision A*	C	216028	7895	$> 1000s$	?
	R			$> 1000s$	?
Collision B*	C	$> 1000s$	?	$> 1000s$	?
	R			$> 1000s$	?

We note that $=_{GR}$ is definitely worse than $=_{GC}$.

Table 10. Spheres and Cylinders: $<_{\Delta\mathbf{R}}$ —degrees

	$<_{\Delta\mathbf{R}}$ degrees	**Time**	**Cells**	$=_G/<_{\Delta\mathbf{R}}$ degrees	**Time**	**Cells**	$=_G/\overset{*}{\rightarrow}^G/<_{\Delta\mathbf{R}}$ degrees	**Time**	**Cells**
S_1, S_2, C	6 / 18	8654	1073	5 / 9	905	267	5 / 7	270	99
S_2, S_3, C	6 / 19	189202	12097	5 / 11	5911	1299	5 / 10	499	213
S_3, S_4, C	6 / 21	248340	11957	5 / 15	8159	1359	5 / 15	580	213

'degrees' is $\texttt{td}(A_n)/\texttt{sotd}(A_n)$.

Table 11. [BH91]: degrees

	$<_{\Delta\mathbf{R}}$ degrees	**Time**	**Cells**	$=_G/<_{\Delta\mathbf{R}}$ degrees	**Time**	**Cells**
Intersection A	6 / 14	29426	3763	17 / 50	2470	273
Intersection B	6 / 14	36262	2795	15 / 41	1482	189
Random A	9 / 16	17355	1219	19 / 68	570	165
Random B	9 / 16	356670	7119	19 / 73	470	141
Ellipse A*	6 / 24	262623	28557	6 / 26	62496	14439
Ellipse B*	6 / 24	$> 1000s$	?	25 / 253	$> 1000s$	?
Solotareff A*	10 / 25	16014	1751	10 / 28	2025	297
Solotareff B*	10 / 25	43439	6091	21 / 69	1647	243
Collision A*	6 / 23	216028	7895	27 / 251	$> 1000s$	?
Collision B*	6 / 23	$> 1000s$	?	36 / 875	$> 1000s$	?

6.1 `TNoI` data

The results of calculating such a quantity are given in Table 8, Table 9 and Table 10, showing a promising correlation to whether our preconditioning (with compatible ordering) is beneficial or not. In particular we note the following points:

- In every example where preconditioning reduces `TNoI` (15 cases) there is a significant reduction in timing (a decrease factor ranging from 4.20 to 757.26) and number of cells produced (a decrease factor ranging from 1.98 to 65.13).
- When preconditioning increases `TNoI` (7 cases) then generally there is an increase in time (an increase factor ranging from 1.79 to 6.13) and the number of cells created (an increase factor ranging from 1.35 to 3.52) or the problem remains infeasible. There is one 'false positive' result ([CMMXY09, Example 2]) where there is an increase in `TNoI` but a slight improvement in the time (a decrease factor of 1.20) and cells produced (a decrease factor of 1.55).
- `TNoI` alone does not measure the abstract difficulty of the calculations: Intersection A has a higher `TNoI` than Ellipse A yet the latter takes 25 times longer and produces over 50 times as many cells. We have only shown how to use it to compare variants of the same problem.

As mentioned above, calculating `TNoI` alone is not of a huge use, and even considering the difference or ratio does little to predict the degree of improvement to expect. However, if we take the logarithm of the ratio (equivalently the difference of the logarithms) of `TNoI` and compare to the time or number of cells we get some interesting results.

Plotting these quantities against each other certainly suggested there was a positive correlation. Recall that the sample correlation coefficient is defined as

$$r_{X,Y} = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}} \tag{10}$$

and is a number between -1 and 1 that indicates how correlated data is. A sample coefficient of 1 indicates perfect positive correlation and a coefficient of -1 indicates perfect negative correlation. Although we are only working with a small bank of data (22 examples) and partially incomplete data (timings of > 1000s were replaced by 10000 seconds and unknown cell numbers were replaced by 100000 to allow for coefficient calculation) there were still promising results.

Let S be the polynomial input, $\mathcal{D}_S$ its corresponding CAD, t_S the time taken to calculate $\mathcal{D}_S$ and c_S the number of cells in $\mathcal{D}_S$. Let G be the Gröbner basis calculated with respect to the compatible ordering and define $\mathcal{D}_G$, t_G and c_G in a similar fashion. With the data set we obtained the sample correlation coefficients were as follows:

- comparing $\log(\texttt{TNoI}(S)) - \log(\texttt{TNoI}(G))$ with $\log(t_S) - \log(t_G)$ gives a sample coefficient $r = 0.821$ which indicates strong correlation (for our limited sample set).

– comparing $\log(\texttt{TNoI}(S)) - \log(\texttt{TNoI}(G))$ with $\log(c_S) - \log(c_G)$ gives a sample coefficient $r = 0.829$ which again indicate a strong correlation (for our limited sample set).

Of course correlation does not imply causation, especially with a relatively small data set, so let us look more deeply at what `TNoI` is measuring.

Table 12. TNoI for Spheres

	$<_{\Delta\mathbf{R}}$			$=_G/<_{\Delta\mathbf{R}}$			$=_G/\overset{*}{\rightarrow}^G/<_{\Delta\mathbf{R}}$		
	`TNoI`	**Time**	**Cells**	`TNoI`	**Time**	**Cells**	`TNoI`	**Time**	**Cells**
S_1, S_2, C	8	8654	1073	5	905	267	4	270	99
S_2, S_3, C	8	189202	12097	6	5911	1299	6	499	213
S_3, S_4, C	8	248340	11957	7	8159	1359	7	580	213

Table 13. TNoI for [BH91]

	$<_{\Delta\mathbf{R}}$			$=_G/<_{\Delta\mathbf{R}}$		
	`TNoI`	**Time**	**Cells**	`TNoI`	**Time**	**Cells**
Intersection A	8	29426	3763	7	2470	273
Intersection B	8	36262	2795	7	1482	189
Random A	9	17355	1219	5	570	165
Random B	9	356670	7119	5	471	141
Ellipse A*	7	262623	28557	6	62496	14439
Ellipse B*	$7 > 1000s$		?	$21 > 1000s$		?
Solotareff A*	9	16014	1751	8	2025	297
Solotareff B*	9	43439	6091	7	1647	243
Collision A*	7	216028	7895	$18 > 1000s$		?
Collision B*	$7 > 1000s$		?	$22 > 1000s$		?

6.2 What is `TNoI` measuring?

Consider what causes `TNoI` to decrease. Let S be a set of polynomials in variables $x_1, \ldots, x_n$ ordered $x_1 < x_2 < \cdots < x_n$. The following are three possible reasons for a decrease in `TNoI`:

1. The number of polynomials in a specific set of variables, $\{x_{i_1}, \ldots, x_{i_l}\}$, is decreased. If x_k is the most important variable then reducing the number of these polynomials will simplify the decomposition in the $(x_1, \ldots, x_k)$-plane. This will simplify the overall CAD, reducing the number of cells produced and hence the time taken to calculate the decomposition.
2. At least one variable is eliminated from a polynomial. If the variable x_k is eliminated from a polynomial p then the decomposition based around p

Table 14. TNoI for [CMMXY09]

	$<_{\Delta\mathbf{R}}$			$=_G/<_{\Delta\mathbf{R}}$		
	TNoI	**Time**	**Cells**	TNoI	**Time**	**Cells**
Cyclic–3	9	3136	381	6	20 + 245 = 265	21
Cyclic–4	16	> 1000*s*	?	6	64 + 5813 = 5877	621
2	7	2249	895	14	22 + 1845 = 1867	579
4	6	3225	421	11	24 + 19738 = 19762	1481
6	4	363	41	5	20 + 918 = 938	89
7	8	3667	895	22	26 + 6537 = 6563	1211
8	6	3216	365	5	21 + 174 = 195	51
13	9	14342	4949	4	18 + 220 = 238	81
14	11	334860	27551	9	21 + 971 = 992	423

will be greatly simplified. This will again simplify the overall CAD, reducing the number of cells produced and hence the time taken to calculate the decomposition.

3. A polynomial in a large number of variables, say k, is replaced by j polynomials each with n_i variables such that $\sum n_i < k$. Intuitively this would increase the *number* of discriminants and resultants calculated, be it in the projection phase of $<_{\text{Col}}$ or in $\neq_{\Delta\mathbf{C}}$, but the results appear in lower levels of the projection tree, and this effect is more potent than the apparent increase in the number of discriminants and resultants. We have yet to build a good model of this, though.

Obviously, in general, a combination of these factors will be the reason for the decrease in TNoI. Also, there may be opposing increases in TNoI, which presumably explains why the 'false positive' of [CMMXY09, Example 2] shows an increase in TNoI but an improvement in the CAD efficiency.

7 Conclusions

- For both $<_{\text{Col}}$ and $<_{\Delta\mathbf{R}}$ and $\neq_{\Delta\mathbf{C}}$, pre-conditioning the equations (where applicable) by means of a Gröbner calculation is often well worth doing.
- Gröbner reduction of inequalities with respect to equalities has never, in our examples, made things worse.
- *A priori*, it can be quite difficult to see which combinations of Gröbner base and Gröbner reduction will be best, but the Gröbner side is generally cheap[4].
- We therefore have multiple equivalent formulations of a given problem. We have investigated the metrics of [DSS04], but have concluded that, at the level of choice of formulation, TNoI is a better predictor. It does not help for predicting the best ordering of variables, for which [DSS04] or the Brown heuristic [Bro04] are appropriate. Phisanbut [Phi11, Chapter 8] found the Brown heuristic sufficiently good, and simpler to compute.

[4] This is a significant change from [BH91], who had examples where the Gröbner calculations was much more expensive than the Cylindrical Algebraic Decomposition.

– In Section 3.2 we saw how $g = 0 \wedge f > 0$ could be reduced to $g = 0 \wedge \text{sprecond}(f, g) > 0$. In principle, given $s_1 = 0 \wedge \cdots \wedge s_k = 0 \wedge f > 0$, after computing a Gröbner base G for the s_i, we could attempt a more general reduction of f by G. Pure `NormalForm` reduction has proved useful (Tables 6, 7), but we do not have enough good examples to measure the utility of a more general pseudoremainder-like reduction.

References

ALMM99. P. Aubry, D. Lazard, and M. Moreno Maza. On the Theories of Triangular Sets. *J. Symbolic Comp.*, 28:105–124, 1999.

BGK85. W. Böge, R. Gebauer, and H. Kredel. Gröbner Bases Using SAC2. In *Proceedings EUROCAL 85*, pages 272–274, 1985.

BH91. B. Buchberger and H. Hong. Speeding-up Quantifier Elimination by Gröbner Bases. Technical Report 91-06, 1991.

Bro03. C.W. Brown. QEPCAD B: a program for computing with semi-algebraic sets using CADs. *ACM SIGSAM Bulletin 4*, 37:97–108, 2003.

Bro04. C.W. Brown. Tutorial handout. `http://www.cs.usna.edu/~wcbrown/research/ISSAC04/handout.pdf`, 2004.

Bro05. C.W. Brown. SLFQ — simplifying large formulas with QEPCAD B. `http://www.cs.usna.edu/~qepcad/SLFQ/Home.html`, 2005.

Buc70. B. Buchberger. Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystem (English translation in [Buc98]). *Aequationes Mathematicae*, 4:374–383, 1970.

Buc98. B. Buchberger. An Algorithmic Criterion for the Solvability of a System of Algebraic Equations. *In Gröbner Bases and Applications*, pages 535–545, 1998.

CH91. G.E. Collins and H. Hong. Partial Cylindrical Algebraic Decomposition for Quantifier Elimination. *J. Symbolic Comp.*, 12:299–328, 1991.

CMMXY09. C. Chen, M. Moreno Maza, B. Xia, and L. Yang. Computing Cylindrical Algebraic Decomposition via Triangular Decomposition. In J. May, editor, *Proceedings ISSAC 2009*, pages 95–102, 2009.

Col75. G.E. Collins. Quantifier Elimination for Real Closed Fields by Cylindrical Algebraic Decomposition. In *Proceedings 2nd. GI Conference Automata Theory & Formal Languages*, pages 134–183, 1975.

DSS04. A. Dolzmann, A. Seidl, and Th. Sturm. Efficient Projection Orders for CAD. In J. Gutierrez, editor, *Proceedings ISSAC 2004*, pages 111–118, 2004.

Gia89. P. Gianni. Properties of Gröbner bases under specializations. In *Proceedings EUROCAL 87*, pages 293–297, 1989.

Kal89. M. Kalkbrener. Solving systems of algebraic equations by using Gröbner bases. In *Proceedings EUROCAL 87*, pages 282–292, 1989.

MM05. M. Moreno Maza. On Triangular Decompositions of Algebraic Varieties. `http://www.csd.uwo.ca/~moreno/Publications/M3-MEGA-2005.pdf`, 2005.

Phi11. N. Phisanbut. *Practical Simplification of Elementary Functions using Cylindrical Algebraic Decomposition.* PhD thesis, University of Bath, 2011.

PQR09. A. Platzer, J.-D. Quesel, and P. Rümmer. Real World Verificatiom. In R.A. Schmidt, editor, *Proceedings CADE 2009*, pages 485–501, 2009.

Tar51. A. Tarski. A Decision Method for Elementary Algebra and Geometry, 2nd ed. *Univ. Cal. Press*, 1951.

Wil12. D.J. Wilson. Real Geometry and Connectness via Triangular Description: CAD Example Bank. `http://opus.bath.ac.uk/29503`, 2012.