Efficient Projection Orders for CAD

Andreas Dolzmann[*]
Fakultät für Mathematik und Informatik
Universität Passau
Germany
dolzmann@uni-passau.de

Andreas Seidl[†]
Departmento de Matematicas, Estadistica y Computacion
Universidad de Cantabria
Spain
seidl@fmi.uni-passau.de

Thomas Sturm[‡]
Fakultät für Mathematik und Informatik
Universität Passau
Germany
sturm@uni-passau.de

ABSTRACT

We introduce an efficient algorithm for determining a suitable projection order for performing cylindrical algebraic decomposition. Our algorithm is motivated by a statistical analysis of comprehensive test set computations. This analysis introduces several measures on both the projection sets and the entire computation, which turn out to be highly correlated. The statistical data also shows that the orders generated by our algorithm are significantly close to optimal.

Categories and Subject Descriptors

G.4. [**Mathematics of Computing**]: Mathematical Software—*Algorithm design and analysis*

General Terms

Algorithms,Performance,Experimentation,Theory

Keywords

Partial CAD,Projection Operator,REDLOG

1. INTRODUCTION

Cylindrical algebraic decomposition (CAD) [1] is an important and powerful tool in real algebraic geometry. One major application is real quantifier elimination [4]; one prominent other application is adjacency algorithms [2, 15].

The pure decomposition algorithm constructs for a finite set $A \subset \mathbb{R}[x_1, \dots, x_r]$ of multivariate polynomials in r variables a decomposition of real r-space into finitely many *cells* such that over each cell all polynomials in A are sign-invariant. With each cell there is simultaneously a *sample point* inside this cell constructed in such a form that the sign of the polynomials in A can be effectively determined.

The CAD construction is commonly described as consisting of three phases:

1. The *projection phase* starts out with the original set $A_r = A$ of polynomials in r variables. In $r-1$ projection steps there are $r-1$ further finite sets A_{r-1}, …, A_1 of polynomials with one fewer variable in each step generated.

2. In the *base phase*, the finitely many zeros of the set A_1 of univariate polynomials obtained in the last projection step are computed in some exact symbolic representation. Both the zeros and the open intervals between these zeros are cells that yield a decomposition D_1 of 1-space. The zeros are at the same time sample points for themselves. For the intervals there are *sample points* explicitly computed.

3. The *extension phase* starts out with the decomposition D_1 of 1-space and successively constructs decompositions D_2, …, D_r of higher-dimensional spaces. For obtaining D_{i+1} there are the sample points of D_i plugged into the polynomials of A_{i+1}, which yields univariate polynomials. Then one proceeds like in the base phase.

One crucial result, which makes this technique work, is that the projection generates polynomials in such a way that the polynomials in A_2 are sign-invariant over the cells obtained from A_1 in the base phase, and that this remains true for higher dimensions when plugging in sample points from the cells.

Consider now a prenex first-order formula φ containing only polynomials from A as terms, and let x_{k+1}, …, x_r denote the quantified variables:

$$\varphi = Q_{k+1}x_{k+1} \dots Q_r x_r \psi, \quad Q_i \in \{\exists, \forall\}.$$

It is easy to see that ψ has an invariant truth value over each cell in D_n, and using the sample points, this truth value can be effectively determined. Moreover, it is possible to produce quantifier-free descriptions for the cells in D_k. A CAD can be exploited for eliminating the quantifiers from φ. For this one uses a rather straightforward recursive algorithm, which checks for $j \in \{k+1, \dots, r\}$ for each cell C_{j-1} in D_{j-1}

[*]`http://www.fmi.uni-passau.de/~dolzmann/`
[†]`http://www.fmi.uni-passau.de/~seidl/`
[‡]`http://www.fmi.uni-passau.de/~sturm/`

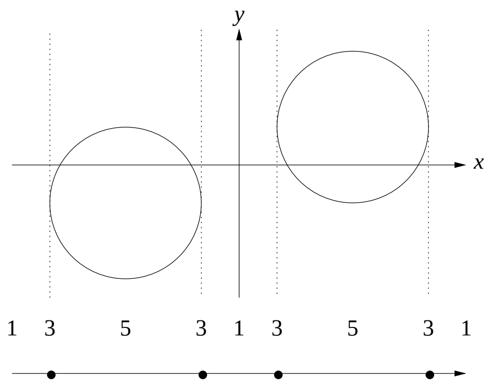


Figure 1: CAD wrt. the variable order $y \rightarrow x$.

according to the quantifier Q_j either *at least one* or *all* cells in D_j that lie in the cylinder over C_{j-1}.

During the past 30 years there have been considerable research and publications on optimizing CAD. For the application to real quantifier elimination, the introduction of *partial* CAD (PCAD) [5] has been one major progress, which affects the extension phase.

The vast majority of improvements, however, extremely focused on improving the projection phase [12, 14, 8, 11, 3, 16]. Most surprisingly, all these contributions concentrating on improved projection operators never examined the relevance of the *order* in which the variables are projected. If one is only interested in pure CAD, then this order can be chosen completely arbitrarily. For quantifier elimination there are restrictions imposed by projecting unquantified variables last and not interchanging $\exists$ with $\forall$. There is, however, still a considerable degree of freedom.

We are going to demonstrate by means of a small example that the projection order is highly relevant for the practical complexity of the overall procedure: We consider two circles of radius 2,

$$\begin{aligned} c_1 &= (x+3)^2 + (y+1)^2 - 4, \\ c_2 &= (x-3)^2 + (y-1)^2 - 4, \end{aligned}$$

one located at $(-3, -1)$ and the other one located at $(3, 1)$:

Figure 1 shows a CAD for these circles choosing the projection order $y \rightarrow x$. This CAD contains $1+3+5+3+1+3+5+3+1 = 25$ cells.

Figure 2 shows, by the way of contrast, a corresponding CAD using the projection order $x \rightarrow y$. Note that the y-axis is drawn horizontally here. We obtain considerably more cells: $1+3+5+7+9+7+5+3+1 = 41$.

It is obvious that the computation of this second CAD requires more computational resources while delivering a result of equal quality for most purposes.

This paper provides results for determining good variable orders from the input beforehand, i.e., without actually constructing any CAD, in order to then construct the desired CAD wrt. such an order.

The plan of the paper is as follows: In Section 2, we introduce time and space measures for characterizing the complexity of a particular CAD computation. These measures apply partly to the projection phase and partly to the overall computation. We discuss a comprehensive example set of CAD's wrt. all relevant orders and apply all our measures

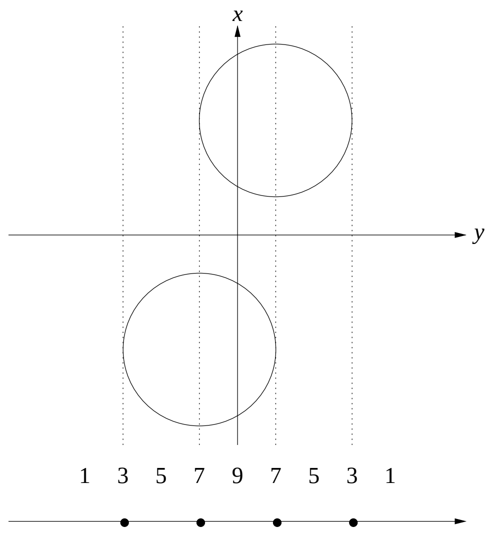


Figure 2: CAD wrt. the variable order $x \rightarrow y$.

to all results. We then show on a precise formal basis that for this set of examples all our measures are statistically strongly correlated. In particular there are measures on the projection that are suitable for predicting the complexity of the overall computation.

In Section 3, we introduce a heuristic algorithm for efficiently constructing one good projection order wrt. to the relevant measures on the projection phase. This is done *without* trying all relevant projection orders. In fact, it cannot be avoided constructing several alternatives for the projection of each variable, but the number of such construction steps in our algorithm is only quadratic in contrast to exponential in the number of variables. We reuse our example database from Section 2 to show two facts: First, our heuristic algorithm yields projection orders that are statistically significantly close to optimal. Second, the overhead originating from constructing the good projection order is negligible while the gain is immense.

In Section 4 we conclude our contribution by summarizing and evaluating our results.

Section 5 finally contains an appendix listing the variable orders used for the computation of the example set in Section 2 such that these computations can be reproduced.

2. MEASURES ON CAD COMPUTATIONS

In this section we consider the following situation: There is a set A of polynomials in r variables given, which possibly origin from a prenex formula φ. For a variable order $X = (x_r, \dots, x_1)$, which we also write $x_r \rightarrow \dots \rightarrow x_1$, the projection results in projection sets $A_r, \dots, A_1$ of projection factors; for convenience we set here and in the sequel $A_r := A$. Based on these sets, the CAD tree $D = (D_1, \dots, D_r)$ is constructed. The levels $D_1, \dots, D_r$ of D are CAD's for $\mathbb{R}^1, \dots, \mathbb{R}^r$, respectively. This situation is completely specified by (A, X) or (φ, X), respectively. We have developed our methods on the basis of the classical Collins–Hong projection. They are, however, also applicable to the other projection operators discussed in the literature.

Our goal is to find for given A or φ a favorable projection order X at the earliest possible stage. For this we want to be able to draw conclusions on the size of the CAD from properties of the intermediate projection sets A_r, A_{r-1}, …, A_1.

Therefore, we are going to systematically investigate numerous complete CAD's wrt. all relevant projection orders, consider certain *measures* on the projection sets as well as on the CAD's, and examine statistical correlations between these measures.

2.1 Measures

There are six measures that we take into account:

1. The number of projection factors in all the projection sets A_r, …, A_1:
$$\operatorname{card}(A, X) = \sum_{i=1}^{r} |A_i|.$$
2. The sum of total degrees of *all* monomials of *all* polynomials in *all* the projection sets A_r, …, A_1:
$$\operatorname{sotd}(A, X) = \sum_{i=1}^{r} \sum_{f \in A_i} \sigma(f),$$
where, using the convention $\mathbf{e} = (e_1, \dots, e_r)$,
$$\sigma\Bigl(\sum_{\mathbf{e}\in E} a_{\mathbf{e}} x_1^{e_1} \cdots x_r^{e_r}\Bigr) = \sum_{\mathbf{e}\in E} \sum_{i=1}^{r} e_i.$$
3. The number of cells in the resulting full CAD:
$$\operatorname{ncad}(A, X) = |D_r|.$$
4. The overall computation time of the full CAD computation in seconds:
$$\operatorname{tcad}(A, X).$$
5. The number of leaves in the partial CAD tree that is generated for quantifier elimination:
$$\operatorname{npcad}(\varphi, X) = \bigl|\{\, c \in D_k \cup \dots \cup D_r \mid c \text{ is a leaf} \,\}\bigr|.$$
6. The overall computation time of the quantifier elimination by partial CAD in seconds:
$$\operatorname{tqe}(\varphi, X).$$

The time measures tcad and tqe depend on the implementation and the machine. We have used the CAD implementation of the second author in the REDLOG package [7] of the widespread computer algebra system REDUCE. We have carried out all our computations on a 2.0 GHz Intel Pentium IV using 128 MB of RAM.

The first four measures card, sotd, ncad, and tcad are defined for sets A of polynomials. They can be as well applied to formulas φ by considering the set of polynomials occurring in these formulas. The last two measures npcad and tqe, in contrast, do not make sense outside a quantifier elimination context.

It might appear more natural to consider in the definition of our sotd the total degrees of the corresponding polynomials instead of sums of total degrees of monomials. Experiments have shown that these measures are highly correlated. Our choice has the advantage to favor sparse polynomials.

For the definition of npcad we have to recall the basic idea of the partial CAD procedure for quantifier elimination [5]: When using full CAD, there is the CAD tree $D = (D_1, \dots, D_r)$ computed, and then the matrix formula is evaluated at the sample points of the cells in D_r. The truth values thus obtained are propagated down to the corresponding root cell in D_k according to the types of quantifiers. Hence full CAD computation and its use for quantifier elimination are two isolated subsequent steps. For partial CAD, in contrast, one tries to determine truth values for the matrix formulas *during extension* already for cells in D_1, …, D_{r-1}, i.e., without fixing all variables to real numbers. For instance, $x_1 - 42 > 0 \land x_2^3 - 4711x_3 = x_4$ is false for $x_1 = \sqrt{2}$, no matter what x_2, …, x_r are. Whenever one succeeds this way for a cell $c \in D_i$, where $i \in \{1, \dots, r\}$, this cell c need not be further extended. In other words, the partial CAD tree is pruned at this point, and c becomes a leaf.

It is now clear, that it is not reasonable to consider the time for the partial CAD construction as a measure: This construction is not isolated from the quantifier elimination but contains a considerable part of the quantifier elimination work, viz. hard computations with algebraic numbers for trial evaluation of the matrix formula. The other part of the quantifier elimination work, viz. solution formula construction, is, in contrast, still an isolated subsequent step. From that point of view, we consider tqe an appropriate counterpart for tcad.

The first two measures card and sotd can be applied after or even during projection. They are candidates for suitable criteria for determining projection orders. The latter four measures ncad, tcad, npcad, and tqe, in contrast, can be applied only after a complete CAD computation or quantifier elimination, respectively. They are going to be used for evaluating the significance of our candidate criteria.

2.2 Computation of the Test Set

We are going to discuss a test set consisting of six CAD examples, mostly from the literature. Each example has been computed wrt. a significant number of projection orders. In fact, we have computed a much more comprehensive example set comprising 48 examples, and selected these six examples as a representative subset for this paper.

All our examples are in fact quantifier elimination examples, which allows us to apply all our measures. Note that also from the point of view of pure CAD, it is not at all a restriction to consider only such examples; they can be considered deliverers of interesting sets of polynomials.

2.2.1 Admissible Projection Orders

On the other hand, we consider the quantifier block structure of the examples in order to restrict the set of possible orders to a reasonable subset: Strictly speaking, quantifier elimination by CAD for a prenex formula
$$\varphi = Q_{k+1}x_{k+1} \dots Q_r x_r \psi$$
requires a projection order with two properties: First, all quantified variables x_r, …, x_{k+1} are projected before all unquantified variables x_k, …, x_1.

Second, the quantified variables have to be projected *essentially* in the order $x_r \to \dots \to x_{k+1}$. This requirement for a fixed projection order is weakened by the fact that in φ like neighbored quantifiers can be equivalently interchanged.

For instance, $\exists x \exists y \forall z \psi$ is equivalent to $\exists y \exists x \forall z \psi$ but *not* generally equivalent to $\exists x \forall z \exists y \psi$. Consequently $z \to y \to x$ and $z \to x \to y$ are both possible projection orders in this example, while $y \to z \to x$ is not. This observation suggests to rewrite $\exists\{x, y\}\forall\{z\}\psi$ thus making visible the *quantifier blocks*.

So returning to the general discussion, we can rewrite our prenex formula φ as

$$\varphi = Q_1 B_1 \dots Q_n B_n \psi,$$

where $Q_i \neq Q_{i+1}$ for $i \in \{1, \dots, n-1\}$ and $B_1, \dots, B_n$ are finite sets of variables. This is the unique *block representation* of a prenex formula. For convenience, let B_0 denote the set of unquantified variables.

From that point of view, *admissible* projection orders for quantifier elimination are characterized by projecting

$$B_n \to \dots \to B_1 \to B_0,$$

while within each block B_i the order can be freely chosen. Obviously, the number of admissible projection orders is given by

$$\prod_{i=0}^{n} |B_i|!.$$

2.2.2 The Quartic Problem

The `quartic` problem has been suggested by Lazard [10]. It asks for necessary and sufficient conditions on the coefficients of a quartic polynomial to be positive semidefinite:

$$\texttt{quartic} = \forall x(px^2 + qx + r + x^4 \geq 0).$$

There are $1! \cdot 3! = 6$ admissible orders. The following table presents the computation results:[1]

	card	sotd	ncad	tcad	npcad	tqe
1	*7*	54	445	4.71	251	7.04
2	*7*	54	445	83.39	251	138.18
3	*7*	*50*	*417*	*0.54*	*235*	*0.89*
4	*7*	*50*	*417*	1.64	239	2.55
5	9	66	$\perp$	>600	$\perp$	>600
6	9	66	$\perp$	>600	$\perp$	>600

We see that card cannot predict differences in ncad, where sotd can. Note that ncad and tcad are surprisingly unrelated here. On the other hand, there is one order, viz. no. 3, that is optimal wrt. all criteria. We have automatically aborted all our computations after 10 minutes. Measures that are unknown due to such unfinished computations are marked with $\perp$.

2.2.3 A Real Implicitization Problem

This example is an exercise on complex implicitization in a textbook [6]. Our formulation asks for a corresponding real implicitization:

$$\texttt{cls7} = \exists u \exists v(x = uv \wedge y = uv^2 \wedge z = u^2).$$

The number of admissible orders is $2! \cdot 3! = 12$.

[1]For all our examples discussed thoughout this section, the actual variable orders used in each table row are collected in an appendix in Section 5.

	card	sotd	ncad	tcad	npcad	tqe
1	*9*	*25*	*889*	*0.09*	*266*	0.16
2	*9*	*25*	*889*	*0.09*	*266*	*0.15*
3	*9*	*25*	*889*	0.15	268	0.20
4	*9*	*25*	*889*	0.14	*266*	0.22
5	*9*	*25*	*889*	0.14	268	0.19
6	*9*	*25*	*889*	0.15	*266*	0.19
7	11	36	1571	0.19	508	0.28
8	11	36	1571	0.18	508	0.28
9	11	36	1571	0.17	582	0.29
10	11	36	1571	0.16	580	0.29
11	11	36	1571	0.17	582	0.29
12	11	36	1571	0.17	580	0.28

Compared to the previous example there is much less variation here. Note that a choice of projection order according to card or sotd yields significantly good ncad, tcad, npcad, and tqe.

2.2.4 Range of Lower Bounds

The following formula asks for the possible range of strict lower bounds on the values of a parabola that has no real zeros.

$$\texttt{as6} = \forall x \forall a \forall b \forall c \exists x' \big((a > 0 \wedge ax'^2 + bx' + c \neq 0) \longrightarrow y < ax^2 + bx + c\big).$$

There are $1! \cdot 4! \cdot 1! = 24$ admissible orders:

	card	sotd	ncad	tcad	npcad	tqe
1	11	42	4199	0.39	283	0.07
2	11	42	4199	0.48	307	0.09
3	12	44	5231	0.55	487	0.15
4	12	44	5231	0.55	487	0.14
5	13	53	6389	1.37	341	0.13
6	12	49	6389	0.38	357	0.09
7	11	50	4007	0.44	*241*	*0.06*
8	11	50	4007	0.55	255	0.08
9	12	50	5027	0.62	395	0.12
10	12	50	5027	0.62	395	0.11
11	12	50	5027	0.58	305	0.10
12	12	50	5027	0.59	321	0.10
13	12	43	5007	0.46	523	0.18
14	12	43	5007	0.40	523	0.17
15	11	39	3975	0.38	423	0.16
16	11	39	3975	0.40	423	0.15
17	11	39	3975	0.28	415	0.16
18	11	39	3975	0.29	415	0.14
19	11	36	3709	0.52	348	0.16
20	11	36	3709	0.21	358	0.11
21	*9*	*28*	*2365*	0.27	272	0.11
22	*9*	*28*	*2365*	0.27	288	0.11
23	*9*	*28*	*2365*	*0.13*	290	0.08
24	*9*	*28*	*2365*	0.14	290	0.08

This is another example with little variation. Here card and sotd do not discover optimal orders wrt. npcad or tqe. The orders that they point at are, however, absolutely acceptable.

2.2.5 Consistency in Strict Inequalities

This problem decides whether the intersection of the open ball with radius 1 centered at the origin and the open cylinder with radius 1 and axis the line $x = 0$, $y + 2 = 2$ is nonempty. It has been introduced by McCallum and used by Collins and Hong for demonstrating PCAD [13, 5]:

$$\texttt{con} = \exists z \exists x \exists y \big(x^2 + y^2 + z^2 < 1 \wedge x^2 + (y + z - 2)^2 < 1\big).$$

There are $3! = 6$ admissible orders:

	card	sotd	ncad	tcad	npcad	tqe
1	12	46	251	*0.02*	51	0.01
2	*8*	43	365	2.24	43	0.04
3	11	*33*	*193*	*0.02*	*29*	0.01
4	11	*33*	*193*	*0.02*	37	*<0.01*
5	*8*	43	365	2.90	47	0.07
6	12	46	251	*0.02*	51	0.01

Comparing the lines 2 and 5 with 3 and 4, we observe that card and sotd contradict each other. Following sotd in these cases yields the best values for ncad, tcad, npcad, and tqe.

2.2.6 Parametrized Collision Problem

The following formula asks if two moving objects, a circle and a square, are going to collide at some time t in the future. The circle is moving with constant velocity $(1, 0)$, while the velocity of the square is parameterized with (v_x, v_y). This example has been used by Collins and Hong for several fixed choices of (v_x, v_y) [5].

$$\begin{aligned}\texttt{pcol} \;=\; & \exists t \exists x \exists y (t > 0 \wedge -1 \leq x - v_x t \leq 1 \wedge \\ & \qquad -9 \leq y - v_y t \leq -7 \wedge (x - t)^2 + y^2 \leq 1).\end{aligned}$$

The number of admissible orders is $3! \cdot 2! = 12$:

	card	sotd	ncad	tcad	npcad	tqe
1	62	250	*144971*	*47.86*	*6969*	*3.78*
2	62	250	*144971*	121.13	*6969*	4.91
3	⊥	⊥	⊥	>600	⊥	>600
4	4678	227337	⊥	>600	⊥	>600
5	62	*248*	149925	58.95	13310	4.53
6	62	*248*	149925	151.50	13310	7.51
7	*57*	323	⊥	>600	⊥	>600
8	*57*	323	⊥	>600	⊥	>600
9	⊥	⊥	⊥	>600	⊥	>600
10	⊥	⊥	⊥	>600	⊥	>600
11	⊥	⊥	⊥	>600	⊥	>600
12	⊥	⊥	⊥	>600	⊥	>600

In this example we observe an immense variety in all measures. In particular, the orders in the lines 3, 9–12 do not even allows to finish projection within the time limit. The probability of failing to finish full as well as partial CAD computation for a random order is 2/3. Minimal card misleadingly points at such failing orders. Minimal sotd does not point at the optimal order but still at acceptable ones.

2.2.7 The X-Axis Ellipse Problem

The X-Axis Ellipse Problem has been firstly stated by Kahan [9]. It has been formulated as a quantifier elimination problem by Lazard [10]. The problem is to write down conditions such that the ellipse

$$\frac{(x-c)^2}{a^2} + \frac{(y-d)^2}{b^2} = 1$$

is inside the circle $x^2 + y^2 = 1$. We treat the special case $d = 0$:

$$\texttt{ell} = \forall x \forall y \big(b^2 (x-c)^2 + a^2 y^2 = a^2 b^2 \longrightarrow x^2 + y^2 \leq 1\big).$$

There are $2! \cdot 3! = 12$ admissible orders:

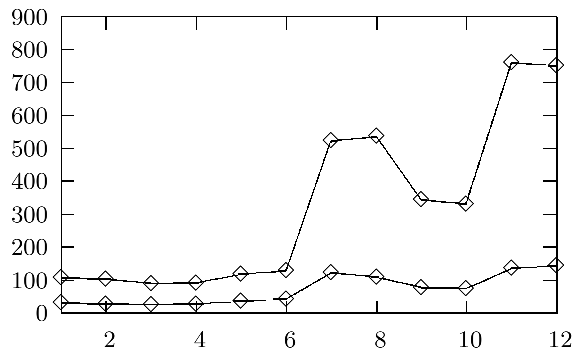
	card	sotd	ncad	tcad	npcad	tqe
1	32	107	114541	46.64	37883	29.68
2	28	103	*51477*	45.54	*11635*	28.70
3	*26*	*89*	64625	*17.10*	21059	*15.59*
4	27	91	96833	29.71	20431	17.70
5	36	119	⊥	>600	74587	260.70
6	43	129	⊥	>600	⊥	>600
7	122	522	⊥	>600	⊥	>600
8	109	537	⊥	>600	⊥	>600
9	77	345	⊥	>600	⊥	>600
10	74	331	⊥	>600	⊥	>600
11	136	761	⊥	>600	⊥	>600
12	143	751	⊥	>600	⊥	>600

Similar to the previous example, there is probability of only 1/3 to finish full CAD and only a slightly higher probability to finish partial CAD for a random order. Both minimal card and sotd point at an optimal order wrt. tcad, npcad, and tqe and this order is almost optimal wrt. ncad.

2.3 Statistical Correlations

In the informal remarks after presenting for each of our examples the resulting table, we have collected some positive and negative observations concerning the possible correlations between our measures.

We are now going to systematically examine the correlations on a precise formal basis. To motivate this, consider the x-axis ellipse problem from Section 2.2.7. For the 12 admissible projection orders the card values (lower line) and the corresponding sotd values (upper line) differ pretty much as can be seen from the following interpolated plot:



Nevertheless, it is our impression that there is a correlation between these measures. To substantiate this, we introduce a formal notion of correspondence: Consider lists $p = (p_1, \dots, p_l)$ and $q = (q_1, \dots, q_l)$ in $\mathbb{R} \cup \{\perp\}$ like the card column and the sotd column for the ellipse example. The value $\perp$ is used to encode that the corresponding value is unknown. Define

$$I = \big\{ \{i, j\} \mid 1 \leq i < j \leq l \text{ and } p_i,\, p_j,\, q_i,\, q_j \in \mathbb{R} \big\};$$

this is the set of all unordered pairs of different indices (orders) for which all values are known. On this basis, we define

$$C = \big\{ \{i, j\} \in I \mid \operatorname{sign}(p_i - p_j) = \operatorname{sign}(q_i - q_j) \big\},$$

the subset of all pairs of indices where the corresponding values for p are ordered in exactly the same way as the ones for q. Finally, we define

$$C' = \big\{ \{i, j\} \in I \mid |\operatorname{sign}(p_i - p_j)| \neq |\operatorname{sign}(q_i - q_j)| \big\},$$

where the corresponding orders between the values are at least not completely opposite. In these definitions, $\operatorname{sign}(x)$ is 1, 0, or -1 if x is positive, zero, or negative, respectively, and $|\cdot|$ denotes the usual absolute value.

If we observe $C = I$, then this would suggest very good correspondence. If $\{i, j\} \notin C$ for some $\{i, j\} \in I$, then we would consider it good correspondence to at least have $\{i, j\} \in C'$. This gives rise to the following definition of the *degree of correspondence* between p and q. It is defined for $|I| > 0$:

$$\operatorname{doc}(p, q) = \frac{|C| + 0.5|C'|}{|I|} \in [0, 1].$$

Note that doc is commutative.

It is not hard to see that for fixed list length l, the average over the degrees of correspondences for all possible choices of pairs of lists is 0.5. A value greater than 0.5 thus indicates an above-average correspondence. For our motivating ellipse example above, we obtain, e.g., $\operatorname{doc}(\text{card}, \text{sotd}) = 0.97$.

The following table collects for all our examples the doc values for the relevant combinations of card, sotd, ncad, tcad, npcad, and tqe. For convenience, the doc's are multiplied by 100 thus rescaling them to percentages:

	quartic	cls7	as6	con	pcol	ell	∅
card–ncad	67	100	89	46	67	67	72
sotd–ncad	100	100	88	74	33	67	77
card–tcad	50	81	78	30	50	100	64
sotd–tcad	83	81	85	56	33	100	73
card–npcad	58	84	67	67	67	70	68
sotd–npcad	91	84	54	93	33	70	70
card–tqe	50	82	62	33	50	100	62
sotd–tqe	83	82	47	60	33	100	67

The last column gives the averages over the corresponding lines. These averages indicate that sotd has a significantly higher doc with all measures on the complete computation than card.

We thus consider sotd in contrast to card to be a suitable indicator after projection for the time and space to be expected for the overall computation.

3. CONSTRUCTING GOOD ORDERS

We recall from the previous section that $\operatorname{sotd}(A, X)$ for a set A of polynomials and a projection order X is the sum of the total degrees of all monomials in all polynomials in all projection sets $A_r, \ldots, A_1$. For a prenex first-order formula φ we may also speak of $\operatorname{sotd}(\varphi, X)$ referring to the set of polynomials occurring in φ.

The results of the previous section provide a good indication that sotd is a suitable measure that is correlated with a high degree of correspondence to all measures that one possibly wishes to optimize in order to save computational resources:

1. small size of the full CAD (ncad),
2. fast computation time for the full CAD (tcad),
3. small size of the partial CAD (npcad),
4. fast computation time for quantifier elimination (tqe).

On the basis of this result we can conclude from the knowledge of *all* projection sets for *all* admissible orders on the interesting time and space measures listed above without actually performing any base phase or extension phase.

The remaining problem is the following: In order to get a basis for the decision, there are still projection phases wrt. *all* admissible orders to be performed. The worst-case number of admissible orders is the factorial of the number of variables and hence exponential in the word length of the input.

In this section we are going to suggest a heuristic algorithm for finding a good projection order wrt. sotd. This algorithm will require quadratically many projection steps in the number of variables and thus in the word length.

We are going to evaluate the quality of the orders determined by our algorithm on the basis of our example set introduced in the previous section. It is going to turn out that the practical computation times are absolutely negligible, while the gain obtained from using the computed orders is immense.

3.1 Greedy Projection

We suggest a *greedy* algorithm for finding a good admissible projection order wrt. sotd. By greedy, we refer to roughly the following idea: We perform the first projection step wrt. to all possible variables. Then we determine the sum of total degrees for each single obtained set. We greedily take the best one, throw away all others, and repeat like that until there are no variables left to order.

ALGORITHM 1 (GREEDY PROJECTION). *Input: a finite set $A \subset \mathbb{R}[x_1, \ldots, x_r]$, $B \subseteq \{x_1, \ldots, x_r\}$. Output: A projection order ω on B and the corresponding intermediate projection set A'.*

```
1  begin
2      $\omega := ()$
3      while $B \neq \emptyset$ do
4          $A' := \bot$
5          for each $x \in B$ do
6              $A'' := \operatorname{project}(\{ f \in A \mid x \text{ in } f \}, x)$
7              $s'' := \sum_{f \in A''} \sigma(f)$
8              if $A' = \bot$ or $s'' < s'$ then
9                  $A' := A'' \cup \{ f \in A \mid x \text{ not in } f \}$
10                 $s' := s''$
11                 $x' := x$
12             fi
13         od
14         $\omega := \omega \circ (x')$
15         $A := A'$
16         $B := B \setminus \{x'\}$
17     od
18 end
```

In Line 6, $\operatorname{project}(M, v)$ denotes one projection step on the set M wrt. the variable v. In Line 7, recall the definition of σ from Section 2.1, and note that sums over the empty set are zero.

For sets A of polynomials we can straightforwardly apply our algorithm to the given set A and the set of all variables occurring in the polynomials in A.

For formulas φ, we have to recall our discussion of admissible orders in Section 2.2.1: For a formula

$$\varphi = Q_1 B_1 \ldots Q_n B_n \psi$$

in prenex block representation, the admissible projection orders are characterized by projecting the blocks in the order

$$B_n \to \cdots \to B_1 \to B_0,$$

where B_0 denotes the set of all unquantified variables. For each of these blocks, the order can be freely chosen. The input A of our algorithm for the first block B_n is the original set of polynomials in φ. In the sequel, for each block B_i with $i \in \{0, \dots, n-1\}$ the input A is the A returned from the preceding call for B_{i+1}.

Note that the algorithm, for several reasons, does not necessarily find a projection order yielding really minimal sotd:

1. We apply a local criterion on projection levels, while sotd is a global criterion on entire projections. In other words, the projection order with smallest sotd need not necessarily have small sums of total degrees in the set obtained after the first projection step.

2. When judging in Line 7 the quality of a trial projection wrt. some variable x, the set A'' is not perfectly the set that has to be considered: Some of the polynomials sorted out in Line 6 would belong into A'' while others would not. This depends on the next variable to be projected, which we do not yet know.

Nevertheless, we shall see that our greedy algorithm applied to the previous set of examples provides orders of very high quality in the following sense: First, they are as a rule close to optimal. We are going to substantiate this in the following section by means of a statistical analysis. Second, and even more important, the computed orders never exceed the time limit in any of our examples, even when there is a high probability of failing.

3.2 CAD Performance with Greedy Projection

The following tables provide a statistical analysis of the performance of the orders generated by our greedy algorithm. It shows that the performance of these orders is considerably above-average. The extra computing time is negligible.

	`quartic`	`cls7`	`as6`
order no. by greedy	3	1	22
time for greedy	0.01	<0.01	0.01
ncad by greedy	417	889	2365
rank w/i ncad	1 of 6	1 of 12	1 of 24
median of ncad	445.00	1230.00	4103.00
mean of ncad	$\perp$	1230.00	4273.00
tcad by greedy	0.54	0.09	0.27
rank w/i tcad	1 of 6	1 of 12	4 of 24
median of tcad	44.05	0.16	0.42
mean of tcad	>215.04	0.15	0.45
npcad by greedy	235	266	288
rank w/i npcad	1 of 6	1 of 12	5 of 24
median of npcad	251.00	388.00	352.50
mean of npcad	$\perp$	411.67	364.25
tqe by greedy	0.89	0.16	0.11
rank w/i tqe	1 of 6	2 of 12	10 of 24
median of tqe	72.61	0.25	0.11
mean of tqe	>224.78	0.24	0.11

	`con`	`pcol`	`ell`
order no. by greedy	3	5	3
time for greedy	<0.01	0.13	0.01
ncad by greedy	193	149925	64625
rank w/i ncad	1 of 6	3 of 12	2 of 12
median of ncad	251.00	$\perp$	$\perp$
mean of ncad	269.67	$\perp$	$\perp$
tcad by greedy	0.02	58.95	17.10
rank w/i tcad	1 of 6	2 of 12	1 of 12
median of tcad	0.02	$\perp$	$\perp$
mean of tcad	0.87	>431.62	>411.59
npcad by greedy	29	13310	21059
rank w/i npcad	1 of 6	3 of 12	3 of 12
median of npcad	45.00	$\perp$	$\perp$
mean of npcad	43.00	$\perp$	$\perp$
tqe by greedy	0.01	4.53	15.59
rank w/i tqe	2 of 6	2 of 12	1 of 12
median of tqe	0.01	$\perp$	$\perp$
mean of tqe	0.02	>401.72	>379.36

4. CONCLUSIONS

We have obtained strong statistical evidence that the projection order is of crucial importance for the success of CAD computations. For determining the quality of a given projection order, we have shown that there is one single measure, viz. sotd, on the projection sets that is correlated to all interesting time and space measures on the computation of a full CAD as well as on quantifier elimination by partial CAD. We have introduced a greedy algorithm for efficiently constructing a good projection order wrt. sotd. Our work closes a considerable gap within the CAD framework.

5. APPENDIX: CATALOGUE OF ORDERS

We finally list for our example set computed in Section 2.2 the projection orders used there. The numbering here corresponds to the that of the table rows in Section 2.2:

`quartic`: 1. $x \to r \to q \to p$, 2. $x \to r \to p \to q$, 3. $x \to q \to r \to p$, 4. $x \to q \to p \to r$, 5. $x \to p \to r \to q$, 6. $x \to p \to q \to r$.

`cls7`: 1. $v \to u \to z \to y \to x$, 2. $v \to u \to z \to x \to y$, 3. $v \to u \to y \to z \to x$, 4. $v \to u \to y \to x \to z$, 5. $v \to u \to x \to z \to y$, 6. $v \to u \to x \to y \to z$, 7. $u \to v \to z \to y \to x$, 8. $u \to v \to z \to x \to y$, 9. $u \to v \to y \to z \to x$, 10. $u \to v \to y \to x \to z$, 11. $u \to v \to x \to z \to y$, 12. $u \to v \to x \to y \to z$.

`as6`: 1. $x' \to c \to b \to a \to x \to y$,
2. $x' \to c \to b \to x \to a \to y$, 3. $x' \to c \to a \to b \to x \to y$,
4. $x' \to c \to a \to x \to b \to y$, 5. $x' \to c \to x \to b \to a \to y$,
6. $x' \to c \to x \to a \to b \to y$, 7. $x' \to b \to c \to a \to x \to y$,
8. $x' \to b \to c \to x \to a \to y$, 9. $x' \to b \to a \to c \to x \to y$,
10. $x' \to b \to a \to x \to c \to y$,
11. $x' \to b \to x \to c \to a \to y$,
12. $x' \to b \to x \to a \to c \to y$,
13. $x' \to a \to c \to b \to x \to y$,
14. $x' \to a \to c \to x \to b \to y$,
15. $x' \to a \to b \to c \to x \to y$,
16. $x' \to a \to b \to x \to c \to y$,
17. $x' \to a \to x \to c \to b \to y$,
18. $x' \to a \to x \to b \to c \to y$,
19. $x' \to x \to c \to b \to a \to y$,
20. $x' \to x \to c \to a \to b \to y$,
21. $x' \to x \to b \to c \to a \to y$,

22. $x' \rightarrow x \rightarrow b \rightarrow a \rightarrow c \rightarrow y$,
23. $x' \rightarrow x \rightarrow a \rightarrow c \rightarrow b \rightarrow y$,
24. $x' \rightarrow x \rightarrow a \rightarrow b \rightarrow c \rightarrow y$.
`con`: 1. $y \rightarrow x \rightarrow z$, 2. $y \rightarrow z \rightarrow x$, 3. $x \rightarrow y \rightarrow z$,
4. $x \rightarrow z \rightarrow y$, 5. $z \rightarrow y \rightarrow x$, 6. $z \rightarrow x \rightarrow y$.
`pcol`: 1. $y \rightarrow x \rightarrow t \rightarrow v_y \rightarrow v_x$, 2. $y \rightarrow x \rightarrow t \rightarrow v_x \rightarrow v_y$,
3. $y \rightarrow t \rightarrow x \rightarrow v_y \rightarrow v_x$, 4. $y \rightarrow t \rightarrow x \rightarrow v_x \rightarrow v_y$,
5. $x \rightarrow y \rightarrow t \rightarrow v_y \rightarrow v_x$, 6. $x \rightarrow y \rightarrow t \rightarrow v_x \rightarrow v_y$,
7. $x \rightarrow t \rightarrow y \rightarrow v_y \rightarrow v_x$, 8. $x \rightarrow t \rightarrow y \rightarrow v_x \rightarrow v_y$,
9. $t \rightarrow y \rightarrow x \rightarrow v_y \rightarrow v_x$, 10. $t \rightarrow y \rightarrow x \rightarrow v_x \rightarrow v_y$,
11. $t \rightarrow x \rightarrow y \rightarrow v_y \rightarrow v_x$, 12. $t \rightarrow x \rightarrow y \rightarrow v_x \rightarrow v_y$.
`ell`: 1. $y \rightarrow x \rightarrow c \rightarrow b \rightarrow a$, 2. $y \rightarrow x \rightarrow c \rightarrow a \rightarrow b$,
3. $y \rightarrow x \rightarrow b \rightarrow c \rightarrow a$, 4. $y \rightarrow x \rightarrow b \rightarrow a \rightarrow c$,
5. $y \rightarrow x \rightarrow a \rightarrow c \rightarrow b$, 6. $y \rightarrow x \rightarrow a \rightarrow b \rightarrow c$,
7. $x \rightarrow y \rightarrow c \rightarrow b \rightarrow a$, 8. $x \rightarrow y \rightarrow c \rightarrow a \rightarrow b$,
9. $x \rightarrow y \rightarrow b \rightarrow c \rightarrow a$, 10. $x \rightarrow y \rightarrow b \rightarrow a \rightarrow c$,
11. $x \rightarrow y \rightarrow a \rightarrow c \rightarrow b$, 12. $x \rightarrow y \rightarrow a \rightarrow b \rightarrow c$.

6. ACKNOWLEDGMENT

The second author has been supported by the DFG, Project WE 566/5, and by the European RTNetwork RAAG, contract no. HPRN-CT-2001-00271.

7. REFERENCES

[1] D. S. Arnon, G. E. Collins, and S. McCallum. Cylindrical algebraic decomposition I: The basic algorithm. *SIAM Journal on Computing*, 13(4):865–877, Nov. 1984.

[2] D. S. Arnon, G. E. Collins, and S. McCallum. Cylindrical algebraic decomposition II: An adjacency algorithm for the plane. *SIAM Journal on Computing*, 13(4):878–889, Nov. 1984.

[3] C. W. Brown. Improved projection for cylindrical algebraic decomposition. *Journal of Symbolic Computation*, 32(5):447–465, Nov. 2001.

[4] G. E. Collins. Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition. In H. Brakhage, editor, *Automata Theory and Formal Languages. 2nd GI Conference*, volume 33 of *Lecture Notes in Computer Science*, pages 134–183. Springer-Verlag, Berlin, Heidelberg, New York, 1975.

[5] G. E. Collins and H. Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *Journal of Symbolic Computation*, 12(3):299–328, Sept. 1991.

[6] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties and Algorithms*. Undergraduate Texts in Mathematics. Springer-Verlag, New York, Berlin, Heidelberg, 1992.

[7] A. Dolzmann and T. Sturm. Redlog: Computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, June 1997.

[8] H. Hong. An improvement of the projection operator in cylindrical algebraic decomposition. In S. Watanabe and M. Nagata, editors, *Proceedings of the International Symposium on Symbolic and Algebraic Computation (ISSAC 90)*, pages 261–264. ACM Press, New York, Aug. 1990.

[9] W. Kahan. An ellipse problem. *ACM SIGSAM Bulletin*, 9(3):11, Aug. 1975. SIGSAM Problem #9.

[10] D. Lazard. Quantifier elimination: Optimal solution for two classical examples. *Journal of Symbolic Computation*, 5(1&2):261–266, Feb. 1988.

[11] D. Lazard. An improved projection for cylindrical algebraic decomposition. In C. L. Bajaj, editor, *Algebraic Geometry and its Applications: Collections of Papers from Shreeram S. Abhyankar's 60th Birthday Conference*. Springer, Berlin, 1994.

[12] S. McCallum. *An Improved Projection Operation for Cylindrical Algebraic Decomposition*. Ph.D. dissertation, Computer Sciences Department, University of Wisconsin, Madison, 1984.

[13] S. McCallum. Solving polynomial strict inequalities using cylindrical algebraic decomposition. Technical Report 87-25.0, RISC, Johannes Kepler University, Linz, Austria, 1987.

[14] S. McCallum. An improved projection operation for cylindrical algebraic decomposition of three-dimensional space. *Journal of Symbolic Computation*, 5(1–2):141–161, Feb.–Apr. 1988.

[15] J. T. Schwartz and M. Sharir. On the piano movers' problem. II. General techniques for computing topological properties of real algebraic manifolds. *Advances in Applied Mathematics*, 4:298–351, 1983.

[16] A. Seidl and T. Sturm. A generic projection operator for partial cylindrical algebraic decomposition. In R. Sendra, editor, *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation (ISSAC 03), Philadelphia, Pennsylvania*, pages 240–247. ACM Press, New York, NY, 2003.