Efficient Computations of Irredundant Triangular Decompositions with the RegularChains Library

Changbo Chen[†], François Lemaire[‡], Marc Moreno Maza[†], Wei Pan[†] and Yuzhen Xie[†]

[†] University of Western Ontario, London N6A 1M8, Canada
[‡] Université de Lille 1, 59655 Villeneuve d'Ascq Cedex, France

Abstract. We present new functionalities that we have added to the RegularChains library in Maple to efficiently compute irredundant triangular decompositions. We report on the implementation of different strategies. Our experiments show that, for difficult input systems, the computing time for removing redundant components can be reduced to a small portion of the total time needed for solving these systems. Since testing the inclusion between two quasi-components can be as expensive as a radical membership test, and many pairs of quasi-components may need to be compared, we believe that we have obtained an efficient solution.

1 Introduction

Efficient symbolic solving of parametric polynomial systems is an increasing need in robotics, geometric modeling, stability analysis of dynamical systems and other areas. Triangular decomposition provides a powerful tool for these systems. However, for parametric systems, and more generally for systems in positive dimension, these decomposition have to face the problem of *removing redundant components*. This problem is not limited to triangular decompositions and is also an important issue in other symbolic decomposition algorithms such as the one of [5] and in numerical approaches [9].

We study and compare different criteria and algorithms for deciding whether a quasi-component is contained in another. Then, based on these tools, we obtain several algorithms for removing redundant components in a triangular decomposition. We report on the implementation of these different solutions within the RegularChains library [6].

We have performed extensive comparisons of these approaches using well-known problems in positive dimension [10]. Our experiments show that, the removal of the redundant components is never a bottleneck. Moreover, we have developed a heuristic inclusion test which provides very good running time performances and which fails very rarely in detecting an inclusion. We believe that we have obtained an efficient solution for computing irredundant triangular decompositions.

2 Inclusion test of quasi-components

In this section we describe our strategies for the inclusion test of quasi-components based on the RegularChains library. We refer to [1, 8, 6] for the notion of a regular chain, its related concepts, such as initial, saturated ideals, quasi-components and the related operations.

Let $\mathbb{K}$ be a field and X be an ordered set of variables. Let $T, U \subset \mathbb{K}[X]$ be two regular chains. Let h_T and h_U be the respective products of their initials. We denote by $\mathrm{Sat}(T)$ the saturated ideal of T. In this section, we discuss how to decide whether the quasi-component $W(T)$ is contained in $W(U)$ or not. Unproved algorithms for this inclusion test are stated in [4] and [7]; they appeared not to be satisfactory in practice, since they are relying on normalized regular chains, which tend to have much larger coefficients that non-normalized regular chains as verified experimentally in [2] and formally proved in [3]. In this paper, we rely on the propositions below, whose proofs are standard and omitted here. Moreover, we aim at using the RegularChains library, which implies avoiding Gröbner basis computations.

Proposition 1. *The inclusion $W(T) \subseteq W(U)$ holds if and only if the following both statements hold*

(C_1) *for all $p \in U$ we have $p \in \sqrt{\mathrm{Sat}(T)}$,*
(C_2) *we have $W(T) \cap V(h_U) = \emptyset$.*

If $\mathrm{Sat}(T)$ is radical, then condition (C_1) can be replaced by:

(C_1') for all $p \in U$ we have $p \in \mathrm{Sat}(T)$,

which is easier to check. Checking (C_2) can be approached in different ways, depending on the computational cost that one is willing to pay. The RegularChains library provides an operation Intersect(p, T) returning regular chains $T_1, \ldots, T_e$ such that we have

$$V(p) \cap W(T) \subseteq W(T_1) \cup \cdots \cup W(T_e) \subseteq V(p) \cap \overline{W(T)}.$$

A call to Intersect can be seen as relatively cheap, since Intersect(p, T) exploits the fact that T is a regular chain. Checking

(C_h) Intersect$(h_U, T)=\emptyset$,

is a good criterion for (C_2). However, when Intersect(h_U, T) does not return the empty list, we cannot conclude. To overcome this limitation, we rely on Proposition 2 and the operation Triangularize of the RegularChains library. For a polynomial system, Triangularize(F) returns regular chains $T_1, \ldots, T_e$ such that $V(F) = W(T_1) \cup \cdots \cup W(T_e)$.

Proposition 2. *The inclusion $W(T) \subseteq W(U)$ holds if and only if the following both statements hold*

(C_1) *for all $p \in U$ we have $p \in \sqrt{\mathrm{Sat}(T)}$,*
(C_2') *for all $S \in$ Triangularize$(T \cup \{h_U\})$ we have $h_T \in \sqrt{\mathrm{Sat}(S)}$.*

This provides an effective algorithm for testing the inclusion $W(T) \subseteq W(U)$. However, the cost for computing Triangularize$(T \cup \{h_U\})$ is clearly higher than that for Intersect(h_U, T), since the former operation cannot take advantage of the fact T is a regular chain.

3 Removing redundant components in triangular decompositions

Let $F \subset \mathbb{K}[X]$ and let $\mathcal{T} = T_1, \ldots, T_e$ be a *triangular decomposition* of $V(F)$, that is, a set of regular chains such that we have $V(F) = W(T_1) \cup \cdots \cup W(T_e)$. We aim at removing every T_i such that there exists T_j, with $i \neq j$ and $W(T_i) \subseteq W(T_j)$. Note that if $T_{i,1}, \ldots, T_{i,e_i}$ are regular chains such that $W(T_i) \subseteq W(T_{i,1}) \cup \cdots \cup W(T_{i,e_i}) \subseteq \overline{W(T_i)}$ holds, then replacing T_i by $T_{i,1}, \ldots, T_{i,e_i}$ in $T_1, \ldots, T_e$ leads again to a triangular decomposition of $V(F)$. Based on the results of Section 2, we have developed the following strategies for testing the inclusion $W(T) \subseteq W(U)$.

`heuristics-no-split`: Tests the conditions (C_1') and (C_h). If both hold, then $W(T) \subseteq W(U)$ has been established, otherwise no conclusions can be made.

`heuristically-with-split`: Tests the conditions (C_1) and (C_h). Checking (C_1) is achieved by means of the operation Regularize [6, 8]: for a polynomial p and a regular chain T, Regularize(p, T) returns regular chains $T_1, \ldots, T_e$ such that we have

- $W(T) \subseteq W(T_1) \cup \cdots \cup W(T_e) \subseteq \overline{W(T)}$,
- for each $1 \leq i \leq e$ the polynomial p is either 0 or regular modulo $\mathrm{Sat}(T_i)$.

Therefore, Condition (C_1) holds iff for all T_i returned by Regularize(p, T) we have $p \not\equiv 0 \mod \mathrm{Sat}(T_i)$. Thus, when applying (C_1), the set T is replaced by regular chains $S_1, \ldots, S_f$ such that we have $W(T) \subseteq W(S_1) \cup \cdots \cup W(S_f) \subseteq \overline{W(T)}$ and such that for every S_i we could check whether $\langle U \rangle \subseteq \sqrt{\mathrm{Sat}(S_i)}$ holds or not.

`certified`: It checks conditions (C_1) and (C_2'). If both hold, then $W(T) \subseteq W(U)$ has been established. If at least one of the conditions (C_1) or (C_2') does not hold, then the inclusion $W(T) \subseteq W(U)$ does not hold either.

The input polynomial systems are well-known systems which can be found at [10]. For each of them, the zero set has dimension at least one. Indeed, redundant components appear rarely when computing triangular decompositions of zero-dimensional polynomial systems.

Table 1 and Table 2 report the number of components and running time of different approaches for these input systems, based on which we make the following observations:

1. The heuristic removal without split performs very well. First, for all examples, except sys 8, it discovers all redundant components. Second, for all examples, except sys 8, its running time is a relatively small portion of the solving time (third column of Table 1).
2. Theoretically, the heuristic removal with split can eliminate more *redundancies* than the other strategies. Indeed, it can discover that a quasi-component is contained in the union of two others, meanwhile these three components are pairwise noninclusive.
3. In practice, the heuristic removal with split does not discover more irredundant components than the heuristic removal without split, except for systems 5 and 6. However, the running time overhead is large (roughly, it multiplies by 3 the solving time).
4. The direct deterministic removal is also quite expensive on several systems (5, 6, 8). Unfortunately, the heuristic removal without split, used as pre-cleaning process does not really reduce the cost of a certified removal of the redundant components.

References

1. P. Aubry, D. Lazard, and M. Moreno Maza. On the theories of triangular sets. *J. Symb. Comp.*, 28(1-2):105–124, 1999.
2. P. Aubry and M. Moreno Maza. Triangular sets for solving polynomial systems: A comparative implementation of four methods. *J. Symb. Comp.*, 28(1-2):125–154, 1999.
3. X. Dahan and É. Schost. Sharp estimates for triangular sets. In *ISSAC 04*, pages 103–110. ACM, 2004.
4. D. Lazard. A new method for solving algebraic systems of positive dimension. *Discr. App. Math*, 33:147–160, 1991.
5. G. Lecerf. Computing the equidimensional decomposition of an algebraic closed set by means of lifting fibers. *J. Complexity*, 19(4):564–596, 2003.

Sys	Name	Triangularize (No removal)		Certified (Deterministic) Proposition 2	
		♯ RC	time(s)	♯ RC	time(s)
1	genLinSyst-3-2	20	1.684	17	1.182
2	Butcher	15	9.528	7	0.267
3	MacLane	161	12.733	27	7.144
4	neural	10	14.349	4	8.948
5	Vermeer	6	27.870	5	58.396
6	Liu-Lorenz	23	29.044	16	121.793
7	chemical	7	71.364	5	7.727
8	Pappus	393	37.122	120	141.702
9	Liu-Lorenz-Li	22	1796.622	9	96.364
10	KdV572c11s21	41	8898.024	7	6.980
11	KdV572c11s22	34	8710.136	4	9.509

Table 1: Triangularize without removal, certified removal

Sys	Heuristic (C_1') and (C_h) (without split)		Certification (Deterministic)		Heuristic (C_1) and (C_h) (with split)		Certification (Deterministic)	
	♯ RC	time(s)	♯ RC	time(s)	♯ RC	time(s)	♯ RC	time(s)
1	17	0.382	17	1.240	17	0.270	17	1.214
2	7	0.178	7	0.259	7	0.147	7	0.325
3	27	3.437	27	8.470	27	3.358	27	8.239
4	4	1.881	4	8.353	4	6.429	4	14.045
5	5	0.771	5	60.108	8	54.455	8	109.928
6	16	1.937	16	123.052	18	96.492	18	203.937
7	5	0.243	5	7.828	5	5.180	5	12.842
8	124	42.817	120	135.780	124	48.756	120	148.341
9	9	8.186	9	101.668	10	105.598	10	217.837
10	7	4.878	7	6.688	7	5.881	7	7.424
11	4	9.576	4	9.509	5	16.273	5	19.517

Table 2: Heuristic removal, without and with split, followed by certification

6. F. Lemaire, M. Moreno Maza, and Y. Xie. The `RegularChains` library. In Ilias S. Kotsireas, editor, Maple Conference 2005, pages 355–368, 2005.
7. M. Moreno Maza. *Calculs de Pgcd au-dessus des Tours d'Extensions Simples et Résolution des Systèmes d'Équations Algébriques*. PhD thesis, Université Paris 6, 1997.
8. M. Moreno Maza. On triangular decompositions of algebraic varieties. Technical Report TR 4/99, NAG Ltd, Oxford, UK, 1999. http://www.csd.uwo.ca/∼moreno.
9. A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical decomposition of the solution sets of polynomial systems into irreducible components. *SIAM J. Numer. Anal.*, 38(6):2022–2046, 2001.
10. *The SymbolicData Project*. http://www.SymbolicData.org, 2000–2006.