Modular algorithms for computing triangular decompositions of polynomial systems

Marc Moreno Maza

University of Western Ontario, London, Canada
`{moreno}@csd.uwo.ca`

Since the early works of Ritt [27], Wu [33], and Yang and Zhang [36], the Characteristic Set Method has been extended and improved by many researchers. This effort has produced more powerful decomposition algorithms, and now applies to different types of polynomial systems or decompositions: parametric algebraic systems [13,17,35], differential systems [4,14,20], difference systems [19], unmixed decompositions and primary decomposition [29] of polynomial ideals, intersection multiplicities [24], cylindrical algebraic decomposition [11,22], quantifier elimination [12], parametric [35] and non-parametric [9] semi-algebraic systems. Today, triangular decomposition algorithms are available in several software packages [2,8,31,32,34]. Moreover, they provide back-engines for computer algebra system front-end solvers, such as MAPLE's `solve` command.

Despite of their successful application in various areas (automatic theorem proving, dynamical systems, program verification, to name a few), solvers based on triangular decompositions are sometimes put to challenge with input polynomial systems that appear to be easy to solve by other methods, based on Gröbner bases. Of course, one should keep in mind that different solvers may have different specifications, not always easy to compare. Nevertheless, for certain classes of systems, say zero-dimensional systems, one can expect that a triangular decomposition on one hand, and the computation of a lexicographical Gröbner basis (followed by the application of Lazard' s `Lextriangular` algorithm [23]) on the other, produce essentially the same thing.

While the development of modular methods for computing Gröbner bases took off in the 1980's thanks to Traverso [30] and Faugère [18], with follow-up works by Arnold [1] and others, the development of such methods for triangular decompositions started only in 2005 with the paper [16] by Dahan, Moreno Maza, Schost, Wu and Xie. This latter method computes a triangular decomposition Δ of a zero-dimensional polynomial system $V(F)$ over the rational numbers by

1. first computing a triangular decomposition, say Δ_p, of that system modulo a sufficiently large prime number p;
2. transforming Δ_p into a canonical triangular decomposition of $V(F \bmod p)$, called the equiprojectable decomposition, E_p of $V(F \bmod p)$; and
3. finally, lifting E_p (using the techniques of Schost [28]) into the equiprojectable decomposition of $V(F)$.

Hence, this method helps to control the effect of expression swell at the level of the numerical coefficients, which resulted in a significant efficiency improvement on a number of famous test systems. However, this modular method has no benefits on expression swell when expression swell manifests as an (unnecessary) inflation on the number of terms. This phenomenon is generally caused by the so-called extraneous or spurious factors in resultants, which have been studied in the case of Dixon resultants [21]. Most algorithms for computing triangular decompositions compute *iterated resultants*, either explicitly or implicitly.

In broad terms, the iterated resultant $\mathrm{res}(f, T)$ between f and a regular chain $T \subseteq \mathbf{k}[X_1 < \ldots X_n]$ encodes conditions for the hypersurface $V(f)$ and the quasi-component $W(T)$ to have a non-empty intersection.

To be precise, we recall some of the results in Section 6 of [10]. Assume that T is a zero-dimensional regular chain. We denote by $V_M(T)$ the multiset of the zeros of T, where each zero of T appears a number of times equal to its local multiplicity as defined in Chapter 4 of [15]. If T is normalized, that is, the initial of every polynomial in T is a constant, then we have:

$$\mathrm{res}(f, T) = \prod_{\alpha \in V_M(T)} f(\alpha).$$

This *Poisson Formula* tells us that, if T is normalized, then $\mathrm{res}(f, T)$ is "fully meaningful". In other words, it does not contain extraneous factors. Now, let us relax the fact that T is normalized. For $i = 1, \ldots, n$, we denote respectively by t_i, h_i, r_i: (1) the polynomial of T whose main variable is X_i, (2) the initial of t_i, (3) the iterated resultant $\mathrm{res}(\{t_1, \ldots, t_{i-1}\}, h_i)$. In particular, we have $r_1 = h_1$. We also define: (1) $e_n = \deg(f, X_n)$, (2) $f_i = \mathrm{res}(\{t_{i+1}, \ldots, t_n\}, f)$, for $0 \leq i \leq n-1$, (3) $e_i = \deg(f_i, x_i)$, for $1 \leq i \leq n-1$. Then, $\mathrm{res}(T, f)$ is given by:

$$h_1^{e_1} \left(\prod_{\beta_1 \in V_M(t_1)} h_2(\beta_1) \right)^{e_2} \cdots \left(\prod_{\beta_{n-1} \in V_M(t_1, \ldots, t_{n-1})} h_n(\beta_{n-1}) \right)^{e_n} \left(\prod_{\alpha \in V_M(T)} f(\alpha) \right)$$

From that second Poisson formula, we can see that all factors but the rightmost one (that is, the one from the first Poisson formula) are extraneous. Indeed, in the intersection $V(f) \cap W(T)$ there are no points cancelling the initials $h_2, \ldots, h_n$.

These observations generalize to regular chains of positive dimension (just seeing the field $\mathbf{k}$ as a field of rational functions) and can explain how the calculation of iterated resultants can cause expression swells in triangular decomposition algorithms. To deal with that problem, the authors of [10] study a few trivariate systems consisting of a polynomial $f(X_1, X_2, X_3)$ and a regular chain $T = \{t_2(X_1, X_2), t_3(X_1, X_2, X_3)\}$. They compute $\mathrm{res}(T, f)$ by

1. specializing X_1 at sufficiently many well-chosen values a,
2. computing $R(a) := \mathrm{res}(N(a), f(a))$ where $f(a) = f(a, X_2, X_3)$ and $N(a)$ is the normalized regular chain generating the ideal $\langle t_2(a, X_2), t_3(a, X_2, X_3) \rangle$ in $\mathbf{k}[X_2, X_3]$, and
3. combining the $R(a)$'s and applying rational function reconstruction.

The numerator of the reconstructed fraction is essentially the desired non-extraneous factor of $\mathrm{res}(T, f)$.

In a recent article [7], we have extended the ideas of [10] so that one can actually compute $V(f) \cap W(T)$ and not just obtain conditions on the existence of those common solutions for f and T. Computing such intersections is the core routine of the incremental triangular decomposition method initiated by Lazard in [22] and further developed by Chen and Moreno Maza [10, 25]. Consequently, we have implemented the proposed techniques and measured the benefits that they bring to the solver presented in [2].

We stress the fact that our objective is to optimize the Intersect algorithm [10] for computing intersections of the form $V(f) \cap W(T)$. Moreover, one of the main applications of our work in this area is to support algorithms in differential algebra, as in the articles [5, 6]. With the challenges of that application[1] in mind and noting the success obtained in applying regular chain theory to differential algebra, our approach to optimize the Intersect algorithm must remain free of (explicit) Gröbner basis computations.

We observe that if Gröbner basis computations are to be used to support triangular decompositions, efficient algorithms exist since the 1990's. As shown in [26], applying Lazard' s `Lextriangular` to the lexicographical Gröbner basis $G(F)$ of a zero-dimensional polynomial ideal $\langle F \rangle$ produces a triangular decomposition of the algebraic variety $V(F)$ in a time which is negligible comparing to that of computing $G(F)$. This efficiency follows from the structure of a lexicographical Gröbner basis as stated by the Gianni-Kalkbrener theorem [23].

The presentation of the modular method presented in [7] for computing $V(f) \cap W(T)$ is dedicated to the case where T is one-dimensional. The cases where T is of dimension higher than one are work in progress and will be discussed in this talk.

Our approach to the design of such a modular method is as follows. We start by identifying hypotheses under which $V(f) \cap W(T)$ is given by a single regular chain C. These hypotheses can be seen as *genericity assumptions*, in particular when T is one-dimensional since C is then *shape lemma* in the sense of [3].

Next, we develop a modular method which computes C, if the genericity assumptions hold, and detects which assumptions do not hold otherwise. One intention of that algorithm is that, whenever a genericity assumption fails, one should be able to recycle the computations performed by the modular method, in order to finish the computations.

This is then followed by a third step where the modular method is enhanced by relaxing the genericity assumptions. We also describe two variants of this modular method: one deterministic and one probabilistic. Experimentation is provided and offers promising results. Indeed, our solver based on this modular method can process various systems which were previously unsolved by our solver (without the modular method).

[1] The differential ideal generated by finitely many differential polynomials is generally not finitely generated, when regarded as an algebraic ideal.

References

1. Arnold, E.A.: Modular algorithms for computing gröbner bases. J. Symb. Comput. **35**(4), 403–419 (2003)
2. Asadi, M., Brandt, A., Moir, R.H.C., Moreno Maza, M., Xie, Y.: Parallelization of triangular decompositions: Techniques and implementation. J. Symb. Comput. **115**, 371–406 (2023)
3. Becker, E., Mora, T., Marinari, M.G., Traverso, C.: The shape of the shape lemma. In: MacCallum, M.A.H. (ed.) Proceedings of ISSAC '94. pp. 129–133. ACM (1994)
4. Boulier, F., Lazard, D., Ollivier, F., Petitot, M.: Representation for the radical of a finitely generated differential ideal. In: International Symposium on Symbolic and Algebraic Computation 1995, Proceedings. pp. 158–166. ACM (1995)
5. Boulier, F., Lazard, D., Ollivier, F., Petitot, M.: Computing representations for radicals of finitely generated differential ideals. Appl. Algebra Eng. Commun. Comput. **20**(1), 73–121 (2009)
6. Boulier, F., Lemaire, F., Moreno Maza, M.: Computing differential characteristic sets by change of ordering. J. Symb. Comput. **45**(1), 124–149 (2010)
7. Brandt, A., Trochez, J.P.G., Moreno Maza, M., Yuan, H.: A modular algorithm for computing the intersection of a one-dimensional quasi-component and a hypersurface. In: Computer Algebra in Scientific Computing (CASC '23). LNCS, Springer (2023)
8. Chen, C., Davenport, J.H., Lemaire, F., Moreno Maza, M., Xia, B., Xiao, R., Xie, Y.: Computing the real solutions of polynomial systems with the regularchains library in maple. ACM Comm. Comp. Algebra **45**(3/4) (2011)
9. Chen, C., Davenport, J.H., May, J.P., Moreno Maza, M., Xia, B., Xiao, R.: Triangular decomposition of semi-algebraic systems. J. Symb. Comput. **49**, 3–26 (2013)
10. Chen, C., Moreno Maza, M.: Algorithms for computing triangular decomposition of polynomial systems. J. Symb. Comput. **47**(6), 610–642 (2012)
11. Chen, C., Moreno Maza, M.: An incremental algorithm for computing cylindrical algebraic decompositions. In: ASCM, Proceedings. pp. 199–221. Springer (2012)
12. Chen, C., Moreno Maza, M.: Quantifier elimination by cylindrical algebraic decomposition based on regular chains. J. Symb. Comput. **75**, 74–93 (2016)
13. Chou, S., Gao, X.: Computations with parametric equations. In: ISSAC 1991, Proc. pp. 122–127 (1991)
14. Chou, S., Gao, X.: A zero structure theorem for differential parametric systems. J. Symb. Comp. **16**(6), 585–596 (1993)
15. Cox, D., Little, J., OShea, D.: Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra. Springer (2013)
16. Dahan, X., Moreno Maza, M., Schost, É., Wu, W., Xie, Y.: Lifting techniques for triangular decompositions. In: International Symposium on Symbolic and Algebraic Computation 2005, Proc. pp. 108–115 (2005)
17. Dong, R., Lu, D., Mou, C., Wang, D.: Comprehensive characteristic decomposition of parametric polynomial systems. In: International Symposium on Symbolic and Algebraic Computation 2021, Proc. pp. 123–130. ACM (2021)
18. Faugre, J.C.: Rsolution des systmes d'quations algbriques. Ph.D. thesis, PhD thesis, Universit Paris 6 (1994)
19. Gao, X., van der Hoeven, J., Yuan, C., Zhang, G.: Characteristic set method for differential-difference polynomial systems. J. Symb. Comput. **44**(9) (2009)
20. Hu, Y., Gao, X.S.: Ritt-wu characteristic set method for laurent partial differential polynomial systems. J. Syst. Sci. Complex. **32**(1), 62–77 (2019)

21. Kapur, D., Saxena, T.: Extraneous factors in the dixon resultant formulation. In: ISSAC 1997, Proc. pp. 141–148. ACM (1997)
22. Lazard, D.: A new method for solving algebraic systems of positive dimension. Discret. Appl. Math. **33**(1-3), 147–160 (1991)
23. Lazard, D.: Solving zero-dimensional algebraic systems. J. Symb. Comput. **13**(2), 117–132 (1992)
24. Moreno Maza, M., Sandford, R.: Towards extending fulton's algorithm for computing intersection multiplicities beyond the bivariate case. In: Computer Algebra in Scientific Computing 2021, Proc. LNCS, vol. 12865, pp. 232–251. Springer (2021)
25. Moreno Maza, M.: On triangular decompositions of algebraic varieties. Tech. Rep. TR 4/99, NAG Ltd, Oxford, UK (1999), presented at the MEGA-2000 Conference
26. Moreno Maza, M., Rioboo, R.: Polynomial gcd computations over towers of algebraic extensions. In: AAECC-11, Proceedings. LNCS, vol. 948, pp. 365–382. Springer (1995)
27. Ritt, J.F.: Differential Algebra. Dover Publications, Inc., New York (1966)
28. Schost, É.: Degree bounds and lifting techniques for triangular sets. In: ISSAC 2002, Proc. pp. 238–245. ACM (2002)
29. Shimoyama, T., Yokoyama, K.: Localization and primary decomposition of polynomial ideals. J. Symb. Comput. **22**(3), 247–277 (1996)
30. Traverso, C.: Gröbner trace algorithms. In: ISSAC 1988, Proc. LNCS, vol. 358, pp. 125–138. Springer (1988)
31. Wang, D.K.: The `Wsolve` package. www.mmrc.iss.ac.cn/ dwang/wsolve.html
32. Wang, D.M.: Epsilon 0.618. http://wang.cc4cm.org/epsilon/index.html
33. Wu, W.T.: A zero structure theorem for polynomial equations solving. MM Research Preprints **1**, 2–12 (1987)
34. Xia, B.: DISCOVERER: a tool for solving semi-algebraic systems. ACM Commun. Comput. Algebra **41**(3), 102–103 (2007)
35. Yang, L., Hou, X., Xia, B.: A complete algorithm for automated discovering of a class of inequality-type theorems. Sci. China Ser. F Inf. Sci. **44**(1), 33–49 (2001)
36. Yang, L., Zhang, J.: Searching dependency between algebraic equations: an algorithm applied to automated reasoning. In: Artificial intelligence in mathematics, pp. 147–156. Oxford University Press (1994)