Journal of Symbolic Computation 79 (2017) 508–515

Contents lists available at ScienceDirect

Journal of Symbolic Computation

www.elsevier.com/locate/jsc

Decoupling highly structured polynomial systems ☆

Daniel J. Bates [a], Andrew J. Newell [b], Matthew E. Niemerg [c]

[a] Department of Mathematics, Colorado State University, Fort Collins, CO, USA
[b] Department of Marine, Earth, and Atmospheric Sciences, North Carolina State University, Raleigh, NC, USA
[c] Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China

ARTICLE INFO

Article history:
Received 16 July 2015
Accepted 19 August 2015
Available online 28 July 2016

Keywords:
Polynomial system
Homotopy continuation
Decoupling
Numerical algebraic geometry

ABSTRACT

An efficient technique for finding numerical approximations of isolated solutions of polynomial systems with a particular structure is presented. This structure is quite specific but arises naturally, for example when computing the critical points of a symmetric polynomial energy function. An illustrative example from magnetism is presented, along with some timing comparisons.

1. Introduction

Let $f : \mathbb{C}^{Nk} \to \mathbb{C}^{Nk}$ be a polynomial system of the form

$$f(z_1, z_2, \ldots, z_N) = \begin{cases} f_1(z_1, z_2, \ldots, z_N) \\ f_2(z_1, z_2, \ldots, z_N) \\ \vdots \\ f_N(z_1, z_2, \ldots, z_N) \end{cases}$$

☆ First and second authors partially supported by NSF grant DMS-1025564; third author partially supported by NSF grant DMS-1025544. First author partially supported by the Mathematical Biosciences Institute (MBI). Third author partially supported by the Simons Institute for the Theory of Computing and the Institute for Interdisciplinary Information Sciences (IIIS).

E-mail addresses: bates@math.colostate.edu (D.J. Bates), ajnewell@ncsu.edu (A.J. Newell), research@matthewniemerg.com (M.E. Niemerg).

URLs: http://www.math.colostate.edu/~bates (D.J. Bates), http://www4.ncsu.edu/~ajnewell (A.J. Newell), http://www.matthewniemerg.com (M.E. Niemerg).

http://dx.doi.org/10.1016/j.jsc.2016.07.016
0747-7171/

$$= \begin{cases} g_1(z_1) & & & + & h_1(z_1, z_2, \ldots, z_N) \\ & g_2(z_2) & & + & h_2(z_1, z_2, \ldots, z_N) \\ & & \ddots & & \vdots \\ & & g_N(z_N) & + & h_N(z_1, z_2, \ldots, z_N), \end{cases} \tag{1}$$

where each f_i, g_i, and h_i is a k-tuple of non-constant polynomials and the z_i are non-overlapping k-tuples of variables for $i = 1, \ldots, N$ ($N \geq 2$). Notice that for each i, the polynomial system g_i depends only on variables z_i while h_i may depend on all of the variables. We also require that any monomial in f_i depending only on z_i appears only in g_i, not h_i. Finally, we require that the polynomials g_i all have exactly the same monomial structure. As a very simple example, suppose that the set of equations breaks into 1-tuples and, for each i, g_i is a single polynomial given by $g_i(x) = a_i x_i^3 + b_i$. Then the only difference between $g_i(x_i)$ and another block $g_j(x_j)$ is in their coefficients and the labels on their variables. While this structure seems quite specific, it arises naturally in at least one fairly general setting, described below.

The isolated solutions of

$$f(z_1, z_2, \cdots, z_N) = 0$$

in $\mathbb{C}^{Nk}$ (including all real solutions isolated over the complex numbers) may be approximated to arbitrarily high accuracy with the methods of numerical algebraic geometry, based on homotopy continuation. In this article, we present a novel method, *decoupling*, that will provide at least some of these solutions (sometimes all) much more efficiently and in a more scalable manner, for problems with the special structure described above.

While the structure required in (1) and the description around it may seem restrictive, it is precisely the structure attained when computing the critical points of a symmetric polynomial energy function by solving the system of first partial derivatives. We encountered this structure when working on a particular set of magnetism problems with multiple dipoles. The terms $g_i(z_i)$ come from the energy that dipole i would have in the absence of the other dipoles, while the terms $h_i(z_1, z_2, \cdots, z_N)$ come from interactions between dipoles. However, this structure is by no means restricted to magnetism and should arise whenever there is a system with both long-range interactions and internal energies.

Since the functions g_i all have the same monomial structure up to a permutation on the labeling of the variables, we can do an *ab initio* solve for g_*, where g_* has the same monomial structure as each g_i but with random complex values for its coefficients. Why we do this will be explained more precisely in §2.2. Then, we solve each of the g_i blocks independently with a parameter homotopy. This is the source of the term *decoupling* – we remove the h_i blocks to decouple the g_i blocks from one another. Since each of the g_i blocks is independent, i.e., there is no overlap of variables between the self-interaction terms, we can easily combinatorially build the solutions of the entire g block, i.e., $g_1 = g_2 = \ldots = g_N = 0$. Finally, we bring in the interaction terms h_i that occur between the blocks using a simple homotopy. Whether this procedure yields all isolated solutions or only a proper subset of them depends on the structure of the system. This is described briefly at the end of §3.

In §2, we provide the necessary background on our computational engine, homotopy continuation, particularly in the polynomial system setting. We follow a formal statement of the method in §3 (including some comparisons to other common solving techniques) with an illustration in §4. In §5 we provide some computational results.

2. Homotopy continuation

Numerical continuation is a well-known and widely-used tool for approximating solutions of systems of equations. The book Allgower and Georg (2003) is a nice reference for these methods, in a general setting. If we restrict our attention to polynomial equations over complex space, we have a variety of useful guarantees that make feasible the computation of *all* isolated solutions. The field dedicated to the development and implementation of homotopy continuation for polynomial systems is commonly referred to as *numerical algebraic geometry*. General references include Bates et al. (2013b), Sommese and Wampler (2005), Li (1997).

In this section, we introduce only the most basic ideas of homotopy continuation and parameter homotopies, leaving most details to the references. The section concludes with a few remarks about software available for these computations.

2.1. Basic homotopy continuation for polynomial systems

To find the solutions of a polynomial system $f(z) = 0$, homotopy continuation begins with a choice of *start system* $g(z)$ that is similar to $f(z)$ but easily solved. We then deform from $g(z)$ to $f(z)$ via a homotopy function,

$$h(z, t) = tg(z) + (1 - t)\gamma f(z),$$

as t goes from 1 to 0, and let $\gamma \in \mathbb{C}$ be random for technical reasons (Sommese and Wampler, 2005). With probability one, for each $t \in (0, 1]$, $h(z, t) = 0$ is a polynomial system having the same number of solutions K as $h(t, 1) = g(z) = 0$. Thus, as t varies, we have K solution paths to follow, which can be traced via numerical predictor–corrector methods. Except for a set of measure 0, paths neither cross nor diverge, and they will either converge to a solution of $f(z) = 0$ or diverge to infinity at $t = 0$. This is the essence of homotopy continuation with virtually no details; refer to the references above for further details.

There is one finer point needed later in this article, the choice of start system $g(z)$. There is a simple, canonical choice called the *Bézout* or *total degree* start system. For each i, polynomial g_i is chosen to be of the form $z_i^{d_i} - 1$, where d_i is the degree of polynomial f_i. This start system is guaranteed to have $\prod_{i=1}^{n} d_i$ isolated, nonsingular solutions for a polynomial system with n equations and variables. Although this start system is trivially built and solved, it also results in the most solution paths of virtually any start system.

Another efficient algorithm recently introduced is *regeneration* (Hauenstein et al., 2011; Bates et al., 2014). This equation-by-equation solver does not quite fit the format above. However, the core of the method is still homotopy continuation and regeneration is sometimes the most efficient homotopy-based method. In the example below, we chose to compare our method to both a total degree start system and regeneration.

2.2. Parameter homotopies

When needing to solve many polynomial systems which differ from one another only in the coefficients, one common choice is the *parameter homotopy*, first introduced in Morgan and Sommese (1989) and Li et al. (1989). We begin by choosing a start system with the same monomial structure as all of the other polynomial systems in the parameterized family under consideration, with random complex coefficients. We solve this system using any standard homotopy method, *e.g.*, a total degree homotopy or regeneration. With probability one, this system has the maximum number of solutions for any polynomial system with the same monomial structure. Thus, all solutions of all polynomial systems in our parameterized family may be solved by moving from the random system to the target systems one at a time. The number of paths saved by using parameter homotopies can be striking, sometimes dropping the number of paths by several orders of magnitude.

It is essential that the first system has random complex coefficients. While it may be tempting to use one of the target polynomial systems, there is no guarantee that any of these will have the maximum number of solutions.

2.3. Software

There are several software packages for computations in numerical algebraic geometry. In this paper, we focus on Bertini (Bates et al., 2006). However, PHCpack (Verschelde, 1999) or HOM4PS-2.0 (Lee et al., 2008) could have been used equally well. The software package Paramotopy (Bates et al., 2013a) was used to set up the parameter homotopies. Paramotopy uses Bertini for all path tracking, so all timings in this article used the same path tracker.

3. Algorithm

Given a polynomial system of the form (1) including the various requirements mentioned in the discussion of (1) in the first paragraph of Section 1, the process of computing some or all of the isolated solutions of $f = 0$ using decoupling has four steps.

Algorithm 1 Decoupling.

Input: Polynomial system $f : \mathbb{C}^{Nk} \to \mathbb{C}^{Nk}$, broken into k-tuples g_i and h_i for $i = 1, \ldots, N$, as in (1). Let $z_1, \ldots, z_N$ be the N k-tuples of variables.
Output: Some or all isolated solutions of $f = 0$.

1. Solve $g_*(z) = 0$, a $k \times k$ polynomial system with the same monomial structure as each $g_i(z_i)$ but random, complex coefficients. Use any standard homotopy continuation technique for this.
2. For each $i = 1, \ldots, N$, use a parameter homotopy from $g_*(z)$ to find all isolated solutions of $g_i(z_i) = 0$ in $\mathbb{C}^k$.
3. Combinatorially concatenate all solutions from all blocks to form N-tuples of solutions $(s_1, \ldots, s_N)$ for $g(z_1, z_2, \ldots, z_N) = 0$. These are the start solutions S for the final step.
4. Solve the original problem $f(z_1, z_2, \ldots, z_N) = 0$ by way of a parameter homotopy from $g(z_1, z_2, \ldots, z_N)$ to $f(z_1, z_2, \ldots, z_N)$, starting from all the solutions in S.

This algorithm clearly terminates as it consists of a finite number of homotopies, each with a finite number of paths.

Steps 1, 2, and 4 involve homotopies, but Step 3 involves no computation. This is just a reorganization of the data collected in the N Step 2 runs.

Complexity analysis. In homotopy continuation, one fundamental measure of complexity is the number of paths tracked. In fact, this is only part of the story as the quality of paths may vary from problem to problem or algorithm to algorithm. However, the well-conditioning of paths is difficult to quantify, so we focus this analysis on the number of paths tracked.

The number of paths to track with a Bézout homotopy is clear and easily computed, just the product of the degrees of the polynomials. Regeneration is more complicated to count as polynomial order and other factors affect both the number of paths and the total run time.

For decoupling, the number of paths in Steps 1 and 2 is virtually negligible compared to the Bézout number for the full system, at least for $k > 1$. Letting d_i be the total degree of the block g_i, then

$$(N+1)\prod_{i=1}^{k} d_i << \prod_{i=1}^{k} d_i^N$$

is certainly true for nontrivial nonlinear systems. Note that the left hand side is an upper bound on the number of paths tracked with decoupling. Furthermore, each of these systems consists of only k equations and variables, so the numerical linear algebra underlying homotopy continuation will be much faster than that for the full system. Step 3 is computationally trivial.

Thus, the main complexity consideration for decoupling is the number of paths in Step 4, which is difficult to predict *a priori*. If the number of solutions of each block $g_i(z_i) = 0$ is the Bézout number of that block, then decoupling is inefficient as Step 4 will involve tracking as many paths as a Bézout homotopy for the full system. However, if the number of solutions of each block $g_i(z_i) = 0$ is significantly less than the Bézout number for the block, we expect a savings in the total run time for the problem.

Of course, it should be noted that Bézout homotopies and regeneration will necessarily find *all* isolated solutions of the system (at least if the perturbed version of regeneration (Bates et al., 2014) is used). With *decoupling*, this is not always the case; see Remark 3.1.

Finally, it should be noted that polyhedral methods are particularly well-suited for highly structured, sparse polynomial systems. This comes into play in two ways. First, one could use such methods for the initial solve of the first block, $g_*(z) = 0$, particularly if the block has more than a few variables. Second, one could consider simply using polyhedral methods for the run in place of decoupling.

This could be a better choice for some problems, though it would be difficult at this point to provide criteria to decide which algorithm would be the better choice for any given problem.

Remark 3.1. The solutions coming from Steps 1–3 of the decoupling algorithm will yield all solutions of $g = 0$, with probability one. One may wonder whether all solutions of the original system $f = 0$ will be found when homotoping g into $f = g + h$. The article Canny and Rojas (1991) provides some insight into this question. Indeed, under certain conditions described in that article, one may conclude that the sum of two polynomial systems has the same number of solutions as one of the summands (with probability one). However, those conditions are quite technical and would require a significant amount of development already available in that paper. Thus, we invite the reader to consider the article Canny and Rojas (1991) for technical statements assuring that all solutions are found via decoupling.

4. Illustrative example from magnetism

We illustrate the algorithm (described formally in §3) with an application in magnetism: a micromagnetic model for interacting single-domain ferromagnets as used in Newell (2009). A single-domain magnet has a moment of fixed magnitude that can only rotate. The moment of each magnet is represented by a vector $\boldsymbol{\mu}_i = V M_s \mathbf{m}_i$, where V is the volume of the magnet, M_s the saturation magnetization (a property of the material), and $\mathbf{m}_i$ a unit vector. The magnetic energy of a system of magnets is

$$E = E_h + E_d + E_a$$

where E_h is the energy of magnetostatic coupling with the external field; E_d is the magnetostatic self-energy (or "demagnetizing energy") of the system; and E_a is the internal (anisotropy) energy.

If N magnets are in an external magnetic field $\mathbf{H}$ (measured in amperes per meter, or $\mathrm{A\,m^{-1}}$), the energy of coupling with the field is

$$E_h = -\mu_0 M_s \mathbf{H} \cdot \sum_{i=1}^{N} V_i \mathbf{m}_i,$$

where μ_0 is the permeability of free space ($4\pi \times 10^{-7}\,\mathrm{H\,m^{-1}}$).

The demagnetizing energy is

$$E_d = \frac{\mu_0}{2} M_s^2 \sum_{i=1,\, j=1,\, j\neq i}^{N} V_i \mathbf{m}_i \mathbf{N}_{ij} \mathbf{m}_j,$$

where $\mathbf{N}_{ij}$, called the demagnetizing tensor, depends only on the geometry of the system (the sizes, shapes, orientations and positions of the magnets).

The internal energy of each magnet is the sum of magnetostatic, magnetocrystalline, and magnetoelastic factors. To lowest order, the energy of each magnet can be approximated by a quadratic expansion in the direction cosines:

$$E_a = \sum_{i=1}^{N} V_i \mathbf{m}_i \mathbf{K}_i \mathbf{m}_i,$$

where each matrix $\mathbf{K}_i$ depends on the shape of the magnet and the physical properties of the material.

The equilibrium states for this system satisfy $\partial E / \partial \mathbf{m}_i = 0$ and the constraints $|\mathbf{m}_i| = 1$ for all i. In translating these equations into a polynomial system, we assume that all the magnets and their moments are on the same plane, though this model naturally extends to 3 dimensions. Since the spatial coordinates do not appear directly, we can also represent unit vector components $m_{\mathrm{x},i}$ and $m_{\mathrm{z},i}$ as x_i and z_i. The constraints are enforced using Lagrange multipliers w_i. For simplicity we also absorb the factors V_i, μ_0, and M_s into the coefficients. For N magnets, there are $3N$ equations and

$3N$ variables (w_i, x_i, and z_i for each i). The polynomial expressions for magnet i derived from $\partial E/\partial \mathbf{m}_i$ are:

$$f_i = \begin{cases} \left(K_{\mathrm{xx},i} - w_i\right) x_i + K_{\mathrm{xz},i} z_i - H_{\mathrm{x}} + \sum_{j=1,\, j\neq i}^{N} \left(N_{\mathrm{xx},ij} x_j + N_{\mathrm{xz},ij} z_j\right) \\ K_{\mathrm{zx},i} x_i + \left(K_{\mathrm{zz},i} - w_i\right) z_i - H_{\mathrm{z}} + \sum_{j=1,\, j\neq i}^{N} \left(N_{\mathrm{zx},ij} x_j + N_{\mathrm{zz},ij} z_j\right) \\ x_i^2 + z_i^2 - 1. \end{cases}$$

In these polynomials, the variables are in 3-tuples $\lambda_i \equiv (x_i, z_i, w_i)$. The isolated-magnet terms are

$$g_i(x_i, z_i, w_i) = \begin{cases} \left(K_{\mathrm{xx},i} - w_i\right) x_i + K_{\mathrm{xz},i} z_i - H_{\mathrm{x}} \\ K_{\mathrm{zx},i} x_i + \left(K_{\mathrm{zz},i} - w_i\right) z_i - H_{\mathrm{z}} \\ x_i^2 + z_i^2 - 1 \end{cases},$$

while the interaction terms are

$$h_i(\lambda_1, \ldots, \lambda_N) = \begin{cases} \sum_{j=1,\, j\neq i}^{N} \left(N_{\mathrm{xx},ij} x_j + N_{\mathrm{xz},ij} z_j\right) \\ \sum_{j=1,\, j\neq i}^{N} \left(N_{\mathrm{zx},ij} x_j + N_{\mathrm{zz},ij} z_j\right) \\ 0 \end{cases}.$$

(The third polynomial is identically zero, i.e. there is no interaction term for the Lagrange multiplier constraint.)

We first solve $g_* = 0$, a system with the same monomial support as each of the g_i but with all coefficients set to random complex numbers. This is Step 1 of the Algorithm. Any standard homotopy continuation method could be used to solve this nonlinear system in 3 equations and 3 unknowns. Denote the set of isolated solutions of $g_* = 0$ by S_*.

For each $i = 1, \ldots, N$, solve $g_i = 0$ via parameter homotopy from g_* and solutions S_*. This is Step 2. This produces solution sets S_i. Notice that there will need to be a trivial relabeling of variables to carry out these parameter homotopies.

The set of all solutions of the system $g_1 = \ldots = g_N = 0$ is then simply the product $S = S_1 \times S_2 \times \cdots \times S_N$ (Step 3). The original system can then be solved via Step 4, with a homotopy from $g_1 = \ldots = g_N = 0$, starting with the points in S.

In this example, half of the paths diverge to infinity in each Step 2 run. When combinatorially building the solutions, we attain a start system with the same number of solutions as the original system. None of the paths tracked from these starting points diverge to infinity, and we ultimately capture all solutions of the original system.

5. Computational results

There is a variation on the decoupling algorithm not described above. Rather than moving directly to each $g_i(z_i)$ and to the final polynomial system, one could instead solve a randomized system (meaning random complex numbers chosen for the coefficients) in place of each $g_i(z_i)$, then move from this randomized system to the final system. This generality is perhaps favorable in capturing more solutions, but analyzing this variant carefully goes beyond the scope of this paper.

The illustrative example of the previous section was run for varying N on a single 2.67 GHz Xeon processor running the CentOS operating system. Table 1 provides the average of 10 runs per entry for $N = 2, \ldots, 8$, using a Bézout homotopy, regeneration, the basic decoupling algorithm, and the previously described variant. Timings were stopped at 24 hours. Decoupling clearly saves computation time for this example.

6. Conclusions

In this article, we have presented a method for splitting a large polynomial system of a particular structure into numerous smaller problems in order to expedite the numerical solution of the polynomial system. We have illustrated the effectiveness of the method with an example from magnetism. The structure required for this article is specific but does arise naturally in at least one type of problem.

Table 1
Average run time of 10 runs for three ways of computing the solutions of the illustrative example – Bézout, regeneration, and decoupling – for various values of N. Timings in seconds except where stated otherwise.

N	Bézout	Regeneration	Decoupling
2	0.5	0.6	**0.4**
3	11.0	2.4	**0.8**
4	268.6	13.2	**1.6**
5	3757.2	89.0	**3.7**
6	59131.5	519.9	**18.9**
7	$>$ 24 hours	3470.7	**105.2**
8	$>$ 24 hours	18109.6	**903.6**

It would be interesting to consider variations on this method. For example, for a system consisting of n blocks of size k (n even), one could instead break the system into $\frac{n}{2}$ blocks of size $2k$ (or otherwise), perhaps yielding more efficiency. The value of this sort of variation on the method is likely problem-dependent.

The chief benefit of decoupling is that a large problem is replaced by many smaller problems and that these smaller problems can be solved very efficiently. We did not parallelize this method, but this latter round of solving many small problems is trivially parallelizable. Similarly, the final run could (and should) be parallelized, making this method even more efficient.

Decoupling in the specific setting of our example may have an interesting physical interpretation. The g blocks provide a system with self-interactions but no interactions between different magnets. The h terms that are introduced in Step 4 of the algorithm bring in interactions between magnets, thus making the solutions more physically meaningful.

Finally, it should be noted that a somewhat different version of this method was useful in the past. Hao et al. (2013) aimed to solve a system of partial differential equations related to tumor growth. They found it useful to first solve the system with an extra viscosity term, then let the coefficient of that term go to zero. This allowed them to solve polynomial systems that they were not otherwise able to solve.

Acknowledgements

The first and third authors were partially supported by NSF grant DMS-1025564; the third author was partially supported by NSF grant DMS-1025544. The first author was also partially supported by the Mathematical Biosciences Institute (MBI). The third author was also partially supported by the Simons Institute for the Theory of Computing and the Institute for Interdisciplinary Information Sciences (IIIS). We also wish to thank the referees for their many useful comments.

References

Allgower, E., Georg, K., 2003. Introduction to Numerical Continuation Methods. Class. Appl. Math., vol. 45. SIAM.
Bates, D., Brake, D., Niemerg, M., 2013a. Paramotopy: parameter homotopies in parallel, submitted for publication.
Bates, D., Davis, B., Eklund, D., Hanson, E., Peterson, C., 2014. Perturbed homotopies for finding all isolated solutions of polynomial systems. Appl. Math. Comput. 247, 301–311.
Bates, D., Hauenstein, J.D., Sommese, A., Wampler, C., 2006. Bertini: software for numerical algebraic geometry. Software available at www.bertini.nd.edu.
Bates, D., Hauenstein, J., Sommese, A., Wampler, C., 2013b. Numerically Solving Polynomial Systems with Bertini. SIAM.
Canny, J., Rojas, J.M., 1991. An optimal condition for determining the exact number of roots of a polynomial system. In: Proceedings of the 1991 International Symposium on Symbolic and Algebraic Computation. ACM, pp. 96–102.
Hao, W., Hu, B., Sommese, A., 2013. Cell cycle control and bifurcation for a free boundary problem modeling tissue growth. J. Sci. Comput. 56 (2), 350–365.
Hauenstein, J., Sommese, A., Wampler, C., 2011. Regeneration homotopies for solving systems of polynomials. Math. Comput. 80, 345–377.
Lee, T., Li, T., Tsai, C., 2008. Hom4ps-2.0, a software package for solving polynomial systems by the polyhedral homotopy continuation method. Computing 83, 109–133.
Li, T., 1997. Numerical solution of multivariate polynomial systems by homotopy continuation methods. Acta Numer. 6, 399–436.

Li, T., Sauer, T., Yorke, J., 1989. The cheater's homotopy: an efficient procedure for solving systems of polynomial equations. SIAM J. Numer. Anal. 26 (5), 1241–1251.
Morgan, A.P., Sommese, A.J., 1989. Coefficient-parameter polynomial continuation. Appl. Math. Comput. 29 (2), 123–160.
Newell, A.J., 2009. Transition to superparamagnetism in chains of magnetosome crystals. Geochem. Geophys. Geosyst. 10 (Q11Z08).
Sommese, A., Wampler, C., 2005. Numerical Solution of Polynomial Systems Arising in Engineering and Science. World Scientific.
Verschelde, J., 1999. Algorithm 795: Phcpack: a general-purpose solver for polynomial systems by homotopy continuation. ACM Trans. Math. Softw. 25, 251–276, software and example systems available at www.math.uic.edu/~jan.