Geometric Completion of Differential Systems using Numeric-Symbolic Continuation

Submitted to SIGSAM, June 2002

Greg Reid* Chris Smith† Jan Verschelde‡

June 25, 2002

Abstract

Symbolic algorithms using a finite number of exact differentiations and eliminations are able to reduce over and under-determined systems of polynomially nonlinear differential equations to involutive form. The output involutive form enables the identification of consistent initial values, and eases the application of exact or numerical integration methods.

Motivated to avoid expression swell of pure symbolic approaches and with the desire to handle systems with approximate coefficients, we propose the use of homotopy continuation methods to perform the differential-elimination process on such non-square systems. Examples such as the classic index 3 Pendulum illustrate the new procedure. Our approach uses slicing by random linear subspaces to intersect its jet components in finitely many points. Generation of enough generic points enables irreducible jet components of the differential system to be interpolated.

*Ontario Research Centre for Computer Algebra, c/o Department of Computer Science, University of Western Ontario, London, Ontario, N6A 5B7, Canada. *Email*: reid@uwo.ca. *URL:* http://www.orcca.on.ca/~reid. Support from NSERC is gratefully acknowledged.

†Ontario Research Centre for Computer Algebra, c/o Department of Computer Science, University of Western Ontario, London, Ontario, N6A 5B7, Canada. *Email:* csmith8@uwo.ca. Support from NSERC is gratefully acknowledged.

‡Department of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago, 851 South Morgan (M/C 249), Chicago, IL 60607-7045, U.S.A. *Email:* jan@math.uic.edu or jan.verschelde@na-net.ornl.gov. *URL:* http://www.math.uic.edu/~jan. This material is based upon work supported by the National Science Foundation under Grant No. 0105739 and Grant No. 0134611.

2000 Mathematics Subject Classification. Primary 34M15; Secondary 14Q99, 65H10, 68W30.
Key words and phrases. Component of solutions, DAE (Differential-Algebraic Equation), differential elimination, differential index, embedding, geometric completion, interpolation, irreducible component, irreducible decomposition, generic point, homotopy continuation, numerical algebraic geometry, numerical jet geometry, path following, polynomial system.

1 Introduction

Over and under-determined (*non-square*) systems of ODE arise in applications such as constrained multibody mechanics and control systems. For example, differential-algebraic equations (DAE) arise from constrained Lagrangian mechanics [2, 28, 30].

In this article we consider systems of constrained ODE which are polynomial functions of their dependent and independent variables. For example consider the class of DAE of the form $\mathbf{u}_t = \mathbf{f}(\mathbf{u})$, $\boldsymbol{\phi}(\mathbf{u}) = \mathbf{0}$. An initial condition $\mathbf{u}(t^0) = \mathbf{u}^0$ which satisfies the constraint $\boldsymbol{\phi}(\mathbf{u}^0) = \mathbf{0}$ can be found by applying Newton's method to the constraint starting from an initial guess. Then the numerical solution of the system proceeds by using the differential equation $\mathbf{u}_t = \mathbf{f}(\mathbf{u})$ as a predictor, and then variations of Newton's method as to correct or project the solutions onto the constraint $\boldsymbol{\phi}(\mathbf{u}) = \mathbf{0}$ during the integration [2, 42].

Differentiation of the constraint yields, in the notation[1] of [2], the relation $\frac{d}{dt}\boldsymbol{\phi} = \boldsymbol{\phi}_{\mathbf{u}}\mathbf{u}_t = \mathbf{0}$. Then elimination (or geometrically: projection) yields $\boldsymbol{\phi}_{\mathbf{u}}\mathbf{f} = \mathbf{0}$. This is potentially a new restriction on initial values. Location of all such new constraints (or so-called reduction of the index) is important for finding consistent initial values and easing the numerical solution of DAE. The differential index of a DAE is r if a minimum of $r+1$ (geometric) differentiations of the DAE are required before no new constraints are found. It has been used as a measure of the difficulty of numerically solving such systems, with low (0 or 1) index systems being the easiest to solve numerically. This index was originally introduced by Gear and its definition was refined and made coordinate independent in [23, 24]. For algebraic developments see [22, 41].

[1] Or more explicitly $\frac{d}{dt}\phi_j = \sum_k \frac{\partial \phi_j}{\partial u_k}\frac{\partial u_k}{\partial t} = 0$.

Recently there has been much progress [4, 17, 19, 20, 22, 25, 27, 41, 44] in the theoretical development and implementation of symbolic differential-elimination algorithms to locate all such constraints. The elimination phase can be complicated because the equations are generally nonlinear. Typically the symbolic methods use Gröbner bases or triangular sets to perform eliminations.

We were motivated to propose a new generation of differential-elimination algorithms by the exploding memory consumption of the symbolic differential-elimination algorithms on complicated examples. This explosion reflects the underlying non-polynomial complexity, an exception is the probabilistic method for determining algebraic observability in polynomial time given in [26]. Moreover, symbolic methods cannot handle approximate input.

An underlying principle of our approach, is our strong emphasis on geometry. In particular we emphasize jet space geometry, the geometry of differential systems [18, 22, 27, 42]. In comparison to the symbolic differentiation-elimination approaches, where symbolic or algebraic manipulations of the equations figure heavily, our approach focuses on the solutions of the system regarded as algebraic equations. We note that these solutions, are possible initial conditions, and not the actual solutions of the differential system. In this jet space picture of a differential equation one regards all the appearing variables (derivatives etc.) as formal unknowns on an equal footing.

A major principle of our approach is to replace what is usually a symbolic preprocessing step with a variation of Newton's method, to locate points on the hidden constraints. This is quite natural since ultimately variants of Newton's method are used in solving the differential system. The new class of methods presented here can be compared to using an iterative method to solve systems (in our case Newton's method), instead of exact elimination (e.g. Gaussian elimination, or its nonlinear cousin Gröbner bases).

Newton's method is one of the most widely used methods to solve systems of nonlinear equations numerically. Although the convergence of this iterative method is quadratic, it is only local in a small neighborhood of a solution. To achieve global convergence, we embed the system we have to solve in a family of systems, a so-called *homotopy*. For polynomial systems, a typical homotopy to solve $\mathbf{p}(\mathbf{x}) = (p_1(\mathbf{x}), ..., p_n(\mathbf{x})) = \mathbf{0}$ is

$$\mathbf{H}(\mathbf{x}, t) = (1 - t)\mathbf{q}(\mathbf{x}) + t\mathbf{p}(\mathbf{x}) = \mathbf{0}, \tag{1}$$

which defines solution paths $\mathbf{x}(t)$ as the continuation parameter t varies from 0 to 1. The paths $\mathbf{x}(t)$ start at $t = 0$ at known solutions of the start system

$\mathbf{q}(\mathbf{x}) = \mathbf{0}$ and lead to the solutions of $\mathbf{p}(\mathbf{x}) = \mathbf{0}$ as t approaches 1. When $\mathbf{q}$ has a structure similar to that of $\mathbf{p}$ (see [16] for precise choices of $\mathbf{q}$), we are guaranteed that numerical path following methods [1] will lead to approximations to all isolated solutions of $\mathbf{p}(\mathbf{x}) = \mathbf{0}$.

The systems of polynomial equations involved in our approach are usually not square and generally have positive dimensional components (submanifolds) of solutions. Our treatment of such systems, using homotopy methods, is made possible by recent theoretical progress initiated by [40], in the developing area of Numerical Algebraic Geometry. The new techniques rely on embedding the given systems in square polynomial systems, by the inclusion of extra (*slack*) variables and random linear equations if necessary. The key idea behind these methods of Sommese, Verschelde and Wampler [33, 34, 35, 36, 37] is to reduce to the zero dimensional case (where there are only finitely many solutions), by slicing the solutions with a random linear space of the appropriate dimension. Enough points are located to interpolate the smooth components of a differential system by lowest degree polynomials.

To motivate the discussion, in Section 2 we will use the example of the Pendulum, a classic in the literature [2, 23, 24]. We first informally show the application of an exact differential-elimination algorithm to the Pendulum equations. In Section 3 we present our geometric method in a more formal manner. First we apply our new numeric-symbolic methods to an easily visualizable example in Section 4.1, and then to the Pendulum in Section 4.2. A discussion of numerical aspects is given in Section 5, and of complexity and related issues in Section 6. Concluding remarks are given in Section 7.

Acknowledgements. One of the authors (GR) would like to thank George Corliss for helpful discussions on automatic differentiation and Kamyar Hazaveh for discussions on the algorithms in the paper.

2 Exact Differential Elimination applied to the Pendulum

By the standard formulation of constrained Lagrangian mechanics, the equations of motion for a unit mass bead constrained to move on a wire of shape $\phi(x, y)$ have form $x_t = u$, $y_t = v$, $u_t = \lambda\phi_x$, $v_t = \lambda\phi_y - g$, $\phi(x, y) = 0$. We use the case of the Pendulum where $\phi(x, y) = x^2 + y^2 - 1 = 0$. Denoting $\phi_1 = \phi$,

the above system is then

$$\mathbf{p}(x, y, u, v, \lambda, x_t, y_t) = \begin{cases} x_t = u, \;\; y_t = v, \;\; u_t = 2\lambda x, \;\; v_t = 2\lambda y - g, \\ \phi_1(x, y) = x^2 + y^2 - 1 = 0. \end{cases} \tag{2}$$

Of course the constraint can be easily removed here by using polar coordinates, but for multi-body systems there are often many constraints, and their elimination by a choice of coordinates may not be practically or even computationally possible.

In what follows, we present the pendulum example in a very informal manner (e.g. as in [2]). A differential-geometric treatment of this example is given in [23], an approach using Gröbner bases is given by [19], and one using characteristic sets by [41].

Here (x, y) gives the position of the bead, u and v are the horizontal and vertical velocities, and λ is the Lagrange multiplier. Initial conditions

$$x(t^0) = x^0, \; y(t^0) = y^0, \; u(t^0) = u^0, \; v(t^0) = v^0, \; \lambda(t^0) = \lambda^0 \tag{3}$$

must satisfy the constraint $\phi_1(x^0, y^0) = (x^0)^2 + (y^0)^2 - 1 = 0$ and this is the most we can deduce from the system (2) regarding it as an algebraic (not differential) system.

To determine all the missing constraints on the initial conditions the equations requires the use of differentiations and eliminations. This type of method will be the focus of this article.

The space for the constraints, the jet space J^0 of order 0, consists of points (t, x, y, u, v, λ), satisfying the constraint equations, regarded as algebraic (non-differential) equations. Graphs of solutions $(t, x(t), y(t), u(t), v(t), \lambda(t))$ consist of points satisfying these algebraic equations, but the points themselves are not solutions of the differential equation.

We will always suppress t in the coordinates of J^0. Further, the first order jet space J^1, consists of points $(t, x, y, u, v, \lambda, x_t, y_t, u_t, v_t, \lambda_t)$. Again these are not solutions of the differential equation, but rather points satisfying the equations regarded as polynomial equations. Also we will suppress the dependence on t since t does not occur explicitly.

2.1 Symbolic Differential-Elimination Procedure

Each step of the procedure consists of an Elimination-Differentiation Step.

1. **Project and Differentiate:** The constraint is obvious in the Pendulum example. However, in general the constraint may not be explicitly isolated, even in the first step. Thus we discuss elimination (projection) at this step for completeness.

   Symbolic elimination (projection) here, means eliminating first order derivatives, to find if there are any relations, amongst the 0 order variables (x, y, u, v, λ) in J^0, with the obvious result that the only constraint is $x^2 + y^2 - 1 = 0$.

   Differentiating the constraint equation and cancelling the common factor 2 yields

   $$xx_t + yy_t = 0. \tag{4}$$

2. **Project and Differentiate:** Elimination by substituting x_t and y_t using $x_t = u$, $y_t = v$ from (2) yields the constraint

   $$\phi_2 = xu + yv = 0, \tag{5}$$

   which is new since it restricts the locus of solutions of ϕ_1 in J^0. We interpret elimination as a geometric projection: $\pi : J^1 \to J^0$ where

   $$\pi : (x, y, u, v, \lambda, x_t, y_t, u_t, v_t, \lambda_t) \mapsto (x, y, u, v, \lambda). \tag{6}$$

   Here we use symbolic elimination to compute this projection, later in our numeric-symbolic method, we will not be able to use symbolic elimination and will have to use the projection as it is defined above.

   Differentiation of this constraint yields

   $$xu_t + yv_t + x_t u + y_t v = 0. \tag{7}$$

   The algorithm continues since a new constraint was found.

3. **Project and Differentiate:** Projection by eliminating u_t, v_t, x_t, y_t, using $u_t = 2\lambda x$, $v_t = 2\lambda y - g$, $x_t = u$, $y_t = v$, from (2) yields the new constraint

   $$\phi_3 = 2\lambda(x^2 + y^2) - gy + u^2 + v^2 = 0. \tag{8}$$

   Differentiating this last constraint yields

   $$2(x^2 + y^2)\lambda_t + 4(xx_t + yy_t)\lambda - gy_t + 2uu_t + 2vv_t = 0. \tag{9}$$

   Again the constraint is new.

4. **Project and Differentiate:** The derivatives u_t, v_t, x_t, y_t can be eliminated from (9) by using (2). However no previous expression is available for λ_t. Hence it is impossible to eliminate λ_t to obtain a new zeroth order constraint. Differentiation of the existing (unaltered) constraints yields nothing new. Thus the process of differentiation and elimination terminates since no new zeroth order constraints were found after 4 elimination-differentiation steps. Its output is the system of equations from the first 3 elimination-differentiation steps: (2-9). The input system (2) is said to have differential index 3 since a sequence of $4 = 3+1$ elimination-differentiation steps was required before no new constraints were obtained. The output system has differential index 0.

The process is summarized in Figure 1.

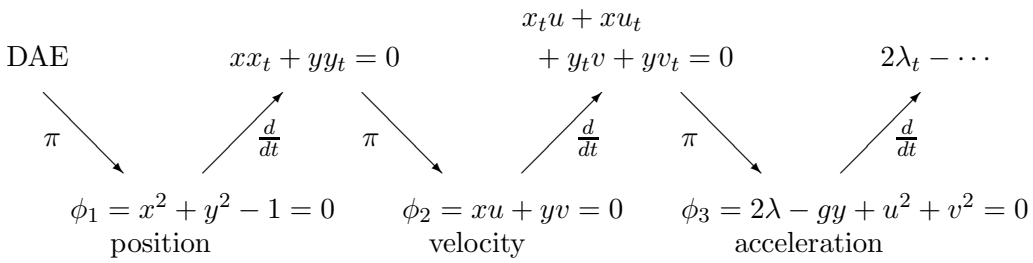

Figure 1: Projection (π) and Differentiation ($\frac{d}{dt}$) on the Pendulum Example. Repeated elimination and differentiation reveal constraints on position, velocity, and acceleration of the bead.

3 Numerical-Symbolic Completion for systems of ODE

In this section we present pseudo code to describe our algorithm to find missing constraints for polynomially nonlinear DAE.

For simplicity we reduce higher DAE to first order DAE in the standard way by introducing new dependent variables corresponding to derivatives. Next by the familiar method of introducing a new dependent variable v such that $\frac{dv}{dt} = 1$, the explicit appearance of t in such systems can also be removed. So without loss of generality we consider first order systems of polynomially

nonlinear DAE with complex coefficients in $\mathbb{C}[\mathbf{u}, \mathbf{u}_t]$ where $\mathbf{u} = (u_1, u_2, ..., u_m)$ are the dependent variables in the system.

Here $J^0 = \mathbb{C}^m$ and $J^1 = \mathbb{C}^{2m}$, suppressing the t dependence. It should be noted that higher order DAE with explicit time dependence can be dealt with directly in the Jet space formalism [3, 27, 42] without recourse to the above transformations. This has the advantage of representing the same geometric object with fewer variables. For example, the pendulum example, can be executed more efficiently in its original second order form. We note also that once the computations have been completed over the complex numbers, then on a case by case basis, one could determine what happens in the real case. One difficulty is that equations can have complex solutions but no real ones, for example: $x^2 + y^2 + 1 = 0$. In general this is the more difficult subject of real algebraic geometry, applied to differential systems, and is not addressed by this article. In this paper we work with complex floating point numbers and for the examples discussed in this paper, standard machine precision suffices.

Symbolic differentiation is define by the total derivative $\frac{d}{dt} = \frac{\partial}{\partial t} + \sum_j \frac{\partial u_j}{\partial t} \frac{\partial}{\partial u_j}$. For a discussion of the subtleties of the relation between symbolic differentiation and geometric differentiation see [22, 41].

The (jet) variety of a differential system $f = 0$ in $\mathbb{C}[\mathbf{u}, \mathbf{u}_t]$ is the set:

$$V(f) = V(f_1, f_2, ..., f_q) := \{(\mathbf{u}, \mathbf{u}_t) \in J^1 = \mathbb{C}^{2m} : f_1 = 0, ..., f_q = 0\}. \quad (10)$$

Geometric projection (elimination) takes points $(\mathbf{u}, \mathbf{u}_t)$ in $V(f)$ to points $\mathbf{u}$ in J^0:

$$\pi\left(V(f)\right) = \{\mathbf{u} \in J^0 = \mathbb{C}^m : f = 0\}. \quad (11)$$

Algorithm 3.1 $[\phi_1, \phi_2, \ldots, \phi_I] = \mathbf{NumericSymbolicCompletion}(f)$

Input : f a polynomial system in $\mathbb{C}[\mathbf{u}, \mathbf{u}_t]$.
Output : $[\phi_1, \phi_2, \ldots, \phi_I]$ a list of contraints

$A := \mathbf{RandomMatrix}(m, m)$ [*A is a random matrix in* $\mathbb{C}^{m \times m}$]
$F := \mathbf{Square}(f, A\mathbf{u} + \mathbf{b})$ [*+random hyperplanes & slack variables*]
$I := 0$
while $\phi_I \neq \emptyset$ do
  $I := I + 1$
  $(G, k_I) := \mathbf{Decompose}(F)$ [*generic points G & top dimension* k^I]
  $\phi_I := \mathbf{Project}(F, G)$ [*eliminate variables* $\mathbf{u}_t$]

$F := \mathbf{Square}([F, \frac{d}{dt}\phi_I])$ [*+new constraints to system, diff. & square*]
end while loop
return $[\phi_1, \phi_2, \ldots, \phi_I]$.

The algorithm **NumericSymbolicCompletion** uses several subroutines from the numerical irreducible decomposition method.

Square(f,g): This algorithm was presented in [33] and returns a square system having the same number of unknowns and equations.

Decompose(F): This algorithm uses incremental interpolation [33], or monodromy as is proposed in [35] (see [6] for the application of the special case of factoring multivariate polynomials with approximate coefficients) to return (G, k) where G are the generic points classified by component, and k is the largest dimension of any component. A general method to determine the top dimension of the solution set of a polynomial system is to start the sequence of homotopies defined in [32] at dimension $n-1$ (where n is the dimension of the ambient space) and to remove linear slices until generic points are found. For the specific systems arising in the treatment of DAE, more efficient implementations of **Decompose** are possible.

Project(F,G): applies random linear projections [33] and interpolation methods to return a system in the form of a irreducible numerical decomposition. See [36] for how to apply structured sample grids in combination with symmetric functions. Here ϕ_J is a list of constraints for each J.

The algorithm above returns polynomially interpolated contraints, with the caveat that there may be some points on the constraints that are singular in that they do not satisfy all the conditions for the local existence and uniqueness theorems of the Geometric Theory of Differential Systems [18, 27]. It is a standard result of geometric elimination theory that $\pi(V(f))$ is not necessarily a variety, but expressible in terms of (set) differences of varieties (see Section 3.2 of [8]). For autonomous DAE these singular varieties result from conditions on the tangent space of an irreducible component, i.e. on $\mathrm{TM}(f)$ and in particular, that $\dim \mathrm{TM}(V(f)) = \dim \pi\left(\mathrm{TM}(V(f))\right)$ [3, 22]. This condition can be checked at the generic points by standard numerical linear algebra, and approximate rank calculations, using for example the singular value decomposition. The task of approximating these singular sets will be investigated in future work. For recent algebraic developments, see [3].

4 Illustrative Examples of Numerical Differential Elimination

Symbolic elimination methods are usually not applicable to systems containing floating point numbers. We describe through examples, how homotopy continuation methods can be used numerically implement the projection (elimination) phase of the above differentiation-elimination methods.

Based on the success of homotopy methods to reach all isolated solution of polynomial systems, Sommese and Wampler outlined in [40] an approach to describe positive dimensional solution components of polynomial systems. A central concept here is that of a *generic point* on a solution component. We can think of a generic point as a random point on the component, away from the singular points the component may have. In [32] a sequence of homotopies was proposed to find generic points on all solution components. For every component, we find exactly as many generic points as its degree. The numerical irreducible decomposition of the solution set of a polynomial system (as introduced in [33]) uses polynomials interpolating through projected generic points to represent all irreducible solution components. See [34] for a special class of projection operators. More efficient decomposition methods use monodromy [35] and interpolate on a structured grid of samples [36]. The implementation [37] as extra module to PHCpack [43] is capable of handling large systems such as the cyclic 8 and 9-roots problem, and several applications from mechanical engineering [39].

4.1 A Visualizable Example

To illustrate our numerical procedure we consider the non-physical example

$$x_t + y^2 - 4 = 0, \quad x_t + 2x^2 + 3y^2 - 6 = 0. \tag{12}$$

0. Square the System. The dimension of the constraints (in J^0) could be at most 2, a possibility which can be checked probabilistically by intersecting the variety of the system with a random line, or equivalently the intersection of 2 random 2 planes. In terms of equations this could be achieved by appending 2 random linear equations in the J^0 variables x, y to the system. It would then, however, have 4 equations in 3 unknowns. To make it square a slack variable z_1 is also incorporated.

In particular we consider the following system which results from the squaring procedure given in Section 3.

$$\begin{cases} x_t + y^2 - 4 + \nu_{11} z_1 = 0 \\ x_t + 2x^2 + 3y^2 - 6 + \nu_{21} z_1 = 0 \\ a_{10} + a_{11} x + a_{12} y + \gamma_{11} z_1 = 0 \\ a_{20} + a_{21} x + a_{22} y + \gamma_{21} z_1 = 0 \end{cases} \tag{13}$$

where $a_{ij}, \nu_{11}, \nu_{21}, \gamma_{11}, \gamma_{21}$ are randomly generated complex constants.

1. Decompose, Project, Differentiate. Applying homotopy continuation to the above system, we have 4 paths, that converge to 4 generic points. However each of these is not a regular solution (i.e. they all have $z_1 \neq 0$). Therefore the dimension of the constraint components is less than 2. Figure 2, shows the varieties of $x_t + y^2 - 4 = 0$ (a parabolic cylinder), $x_t + 2x^2 + 3y^2 - 6 = 0$ (an elliptic cone), and their intersection (the saddle-shaped curve). It is evident from Figure 2 why a random line, orthogonal to J^0 does not intersect this curve.

An important point in what follows, is that appending a system of linear equations only involving the J^0 variables enables us to carry out the projection onto J^0. We keep following the standard decomposition procedure given in the works of [33], by next checking for 1-dimensional components in J^0. This is achieved by removing one of the linear equations[2], and the slack variable, so we now have the square system:

$$\begin{cases} x_t + y^2 - 4 = 0 \\ x_t + 2x^2 + 3y^2 - 6 = 0 \\ a_{10} + a_{11} x + a_{12} y = 0 \end{cases} \tag{14}$$

Geometrically we determine the existence of 1-dimensional components by taking a random plane and determining whether it intersects the variety of the DAE in J^1. From Figure 2, we see the given plane does intersect the variety of the DAE in two generic points. For sake of visualization, we have given a real plane (which apparently with high probability might not intersect the given curve). In fact, here and throughout the paper, the methods work over the complex numbers, where such intersections are guaranteed to occur (the reader is invited

[2]This removal is performed by homotopy continuation, see [32] for the definition of a sequence of homotopies to find generic points efficiently.

to check by doing the elimination algebraically). Since there are two generic points, when this curve is projected to J^0 (see the arrows in Figure 2, showing the projection), the projected curve had degree two.

Translating the random plane, and again applying homotopy continuation, generates more intersection points, which are used to interpolate the equation of the projected variety in J^0. As we are interpolating a planar quadric, five samples suffice. Normalizing the leading coefficient to one, we obtain an equation of the form

$$x^2 + (1+\epsilon_1)y^2 - (1+\epsilon_2) = 0, \tag{15}$$

the circle shown in the x, y plane in Figure 2. Here ϵ_1 and ϵ_2 are complex numbers whose magnitude depends on the accuracy of the generic points and the degree of the equation. The higher the degree of the constraint, the more accurate the sample points need to be.

We differentiate the polynomial in (15) and solve

$$\begin{cases} x_t + y^2 - 4 = 0 \\ x_t + 2x^2 + 3y^2 - 6 = 0 \\ a_{10} + a_{11}x + a_{12}y = 0 \\ 2xx_t + 2(1+\epsilon_1)yy_t = 0 \end{cases} \tag{16}$$

Here we find 2 generic points as solutions of this system. Thus the dimension of constraints in J^0 has not decreased, no new constraints are found, and the procedure terminates with output system (12), (15), and the derivative of (15).

4.2 Numerical Procedure applied to the Pendulum

We now describe, through our illustrative example, the application of the numerical projection elimination to the Pendulum, with system $\mathbf{p}(x, y, u, v, \lambda, x_t, y_t) = \mathbf{0}$ as defined in (2).

0. Square the System. The dimension of the constraints (in J^0) can be at most five. We consider the following system (using 10 for g) which results from the squaring procedure mentioned in Section 3.

$$\begin{cases} \mathbf{p}(x, y, u, v, \lambda, x_t, y_t) + \alpha z_1 = \mathbf{0} \\ a_{k0} + a_{k1}x + a_{k2}y + a_{k3}u + a_{k4}v + a_{k5}\lambda + \gamma_k z_1 = 0, \\ k = 1, 2, \ldots, 5 \end{cases} \tag{17}$$

where $\alpha \in \mathbb{C}^{5\times 1}$ and the coefficients a_{ij} and γ_{i1} are randomly generated complex constants. This gives us 10 equations in 10 unknowns.

1. **Decompose, Project, Differentiate:** The goal of this step is to find the constraint on the position of the bead. To find generic points, we solve the system (17) using homotopy continuation. The projection is ensured by the random linear equations involving the J^0 variables. We find no regular solutions (i.e.: no solutions with $z_1 = 0$). So there are no constraints with 5-dimensional components. Following the decomposition procedure of [33] we remove one of the random planes, and also the slack variable so that the system is still square:

$$\begin{cases} \mathbf{p}(x, y, u, v, \lambda, x_t, y_t) = \mathbf{0} \\ a_{k0} + a_{k1}x + a_{k2}y + a_{k3}u + a_{k4}v + a_{k5}\lambda = 0, \\ k = 1, 2, 3, 4 \end{cases} \tag{18}$$

With four random hyperplanes we cut the four dimensional solution set given by the original equations down to a set of dimension zero, i.e. to a set of two generic points.

The purpose of the operation is to eliminate the variables x_t, y_t, u_t, v_t, therefore, the hyperplanes do not involve these variables. For random hyperplanes with nonzero coefficients for x_t, y_t, u_t, v_t, we find four instead of two generic points.

The position constraint can be obtained by sampling more generic points, varying the four hyperplanes in the system (18). The interpolation through these generic points uses symmetric functions as in [36]. Normalizing the leading coefficient to one, we obtain an equation of the form

$$\widetilde{\phi_1} = x^2 + (1 + \epsilon_1)y^2 - (1 + \epsilon_2) = 0, \tag{19}$$

where ϵ_1 and ϵ_2 are complex numbers whose magnitude depends on the accuracy of the generic points and the degree of the equation. Other monomials in $\widetilde{\phi_1}$, involving u and v, with coefficients of a magnitude smaller than or equal to the working precision are deleted. In general, the higher the degree of the constraint, the more accurate the sample points need to be.

Differentiating we obtain:

$$\frac{d}{dt}\widetilde{\phi_1} = 2xx_t + 2(1 + \epsilon_1)yy_t = 0. \tag{20}$$

2. **Decompose, Project, Differentiate:** In this step we will find the constraint on the velocity of the bead. First we have to include $\frac{d}{dt}\widetilde{\phi_1}$ in the system, which then has 10 equations for 9 variables. So we introduce $1\ (= 10 - 9)$ slack variable z_1 obtaining the square system:

$$\begin{cases} \mathbf{p}(x, y, u, v, \lambda, x_t, y_t) + \alpha z_1 = \mathbf{0} \\ \frac{d}{dt}\widetilde{\phi_1} + \nu_{61} z_1 = 0 \\ a_{k0} + a_{k1}x + a_{k2}y + a_{k3}u + a_{k4}v + a_{k5}\lambda + \gamma_k z_1 = 0, \\ k = 1, 2, 3, 4 \end{cases} \tag{21}$$

We solve the system (21) using homotopy continuation and find no regular solutions (i.e.: no solutions with $z_1 = 0$). So there are no constraints with 4-dimensional components. We remove one of the random planes, and also the slack variable so that the system is still square:

$$\begin{cases} \mathbf{p}(x, y, u, v, \lambda, x_t, y_t) = \mathbf{0} \\ \frac{d}{dt}\widetilde{\phi_1} = 0 \\ a_{k0} + a_{k1}x + a_{k2}y + a_{k3}u + a_{k4}v + a_{k5}\lambda = 0, \quad k = 1, 2, 3 \end{cases} \tag{22}$$

where the coefficients a_{ij} are again chosen at random. Here we find four generic points as solutions of this system. As these four generic points are cut out by three random hyperplanes, adding the derivative to the original system has dropped the dimension by one. Note that we did not include $\widetilde{\phi_1} = 0$ in the system because the inclusion of this equation does not alter the variety of the system in J^1.

Four is the degree of the intersection of two quadratic hypersurfaces given by the exact equations

$$\phi_1 = x^2 + y^2 - 1 = 0 \quad \text{and} \quad \phi_2 = xu + yv = 0. \tag{23}$$

We have found already an interpolated equation $\widetilde{\phi_1}$, approximating ϕ_1. Since $\deg(\widetilde{\phi_1}) = 2$, we know that $\deg(\widetilde{\phi_2}) = 2$, because their intersection has degree four. Via interpolation, we find a constraint on the velocity as a linear combination of the two exact equations:

$$\widetilde{\phi_2} = \alpha\phi_1 + \beta\phi_2, \tag{24}$$

where α and β are some constants.

We differentiate (24) to obtain $\frac{d}{dt}\widetilde{\phi_2}$.

3. **Decompose, Project, Differentiate:** In this step we use $\widetilde{\phi_2}$ to generate a constraint on the acceleration of the bead. We include $\frac{d}{dt}\widetilde{\phi_2}$ and introduce slack variable z_1 to obtain the square system:

$$\begin{cases} \mathbf{p}(x, y, u, v, \lambda, x_t, y_t) + \alpha z_1 = \mathbf{0} \\ \frac{d}{dt}\widetilde{\phi_1} + \nu_1 z_1 = 0, \frac{d}{dt}\widetilde{\phi_2} + \nu_2 z_1 = 0 \\ a_{k0} + a_{k1}x + a_{k2}y + a_{k3}u + a_{k4}v + a_{k5}\lambda + \gamma_k z_1 = 0, \quad k = 1, 2, 3 \end{cases} \tag{25}$$

Using homotopy continuation we find no solutions with $z_1 = 0$ so there are no constraints with 3-dimensional components. We remove one of the random planes, and also the slack variable so that the system is still square:

$$\begin{cases} \mathbf{p}(x, y, u, v, \lambda, x_t, y_t) = \mathbf{0} \\ \frac{d}{dt}\widetilde{\phi_1} = 0, \frac{d}{dt}\widetilde{\phi_2} = 0 \\ a_{k0} + a_{k1}x + a_{k2}y + a_{k3}u + a_{k4}v + a_{k5}\lambda = 0, \quad k = 1, 2 \end{cases} \tag{26}$$

We find six generic points that lie on a two dimensional solution component. Six is the degree of the intersection of the three constraints, respectively on position, velocity, and acceleration of the bead. By interpolation in samples obtained by moving the last two hyperplanes we find a linear combination of the three quadrics; denote this third constraint by $\widetilde{\phi_3}$.

We compute $\frac{d}{dt}\widetilde{\phi_3}$.

4. **Decompose, Project, Differentiate:** The process terminates when no new constraints are found. The inclusion of $\frac{d}{dt}\widetilde{\phi_3}$ introduces 1 equation and one new jet variable λ_t, so slack variables are not required:

$$\begin{cases} \mathbf{p}(x, y, u, v, \lambda, x_t, y_t) = \mathbf{0} \\ \frac{d}{dt}\widetilde{\phi_1} = 0, \frac{d}{dt}\widetilde{\phi_2} = 0, \frac{d}{dt}\widetilde{\phi_3} = 0 \\ a_{k0} + a_{k1}x + a_{k2}y + a_{k3}u + a_{k4}v + a_{k5}\lambda = 0, \quad k = 1, 2 \end{cases} \tag{27}$$

We find again six generic points on a two dimensional solution component, thus $\frac{d}{dt}\widetilde{\phi_3} = 0$ does not cut the dimension by one as previous derivatives did.

Thus the method terminates. Also in J^1 we see that the number of generic points drops so that the differential index is three.

5 Numerical Aspects

The calculations for this paper were carried out with PHCpack [43], recently extended with routines for the numerical irreducible decomposition [37]. PHCpack contains symbolic differentiation methods for polynomials for the differentiations needed in this paper.

PHCpack uses polynomials interpolating through projected [34] generic points to represent irreducible solution components [33]. The computation of these interpolation polynomials is one of the most expensive parts of the method, and can be the least well conditioned. In contrast to the stable computation of generic points the weakest link in our method, concerns the stability of differentiating the interpolating polynomials. For a discussion of such difficulties, see [13, Chapter 5].

We now discuss some recent progress on these issues.

In [36] it was shown that the decomposition could be certified by only requiring generic points and without interpolation polynomials. In particular the decomposition is certified by using the decomposition based on monodromy with linear traces, and the need to construct filter polynomials to reject generic points lying on higher dimensional components is avoided. In non-trivial examples with multiplicity free components this new method [36] was able to be executed in standard precision arithmetic where previously multi-precision arithmetic was required.

The above results imply that the existence or non-existence of a new constraint can be certified at each differentiation step without the construction of interpolation polynomials. In the current paper, if there is a new constraint, to progress to the next differentiation step we still require the computation of the interpolation polynomials for the constraints (although see Section 6.4 for an interpolation-free completion algorithm).

In [36] progress was made on the efficient and accurate construction of interpolation polynomials on a structured grid of samples, using divided differences, and applying symmetric functions. Using a *bootstrapping* technique, the Newton form of the interpolating polynomial can be constructed. The exploitation of the Newton identities enables the number of samples to be reduced to its theoretical minimum (the number of monomials). However as numerical experiments show in [36] it is advisable, for high degree components, to use extra samples for better conditioning of the interpolants.

Finally we mention that the difficulties of higher multiplicity components are not discussed in this paper, see [38] for recent progress.

6 Indices, complexity and systems of PDE

The differential index gives a measure of complexity due to differentiation in the geometric completion of systems of DAE. We discuss the possibility of using straight line encodings of the input systems of polynomials in improving the complexity of completion methods. Finally we discuss a version of a completion algorithm for systems of partial differential equations. This version uses straight line encodings of the input differential equations and does not use interpolation polynomials.

6.1 Indices

The differential index we use is geometric [22], and and is closely related to that introduced by Reich [24] and Rabier and Rheinboldt [23]. For systems with several components, it can take different values, depending on the component.

There has been considerable discussion about different definitions of indices, some non-geometric, some geometric and some involving perturbations of the solutions of such systems [5, 10, 15, 29].

The geometric definition of the index we use is bounded by the number m of dependent variables [22, 41]. It is important that geometric differentiations are used in the completion procedure. Symbolic differentiations, with Gröbner bases being used for the constraints (or even for membership testing for new constraints) can lead to incorrect and arbitrarily high values for the index (e.g. see [22, equations (41)-(43)]). Since we use lowest degree polynomials for interpolating the constraints, this ensures that our symbolic differentiations of the interpolated constraints are geometric differentiations.

6.2 Complexity

When Gröbner bases or similar exact methods are used to test for new constraints at each differentiation step, there is generally underlying non-polynomial complexity in the symbolic differential elimination methods.

We now outline some possibilities for improving the efficiency and underlying complexity of our method. Newton's method has quadratic order of convergence, once it is in the basin of attraction of a root. Moreover, Shub and Smale [31] demonstrated that the problem of whether a generic system has a root, can be answered in polynomial time (under certain circumstances,

which include the absence of multiplicities). For the homotopy methods used in the numerical irreducible decomposition, the number of roots (the geometric degree), is bounded above by the Bézout bound, or more sharply by the mixed volume bound. The geometric degree is thus an important parameter in the complexity of such methods.

The lengths of the polynomials themselves are also crucial parameters in the complexity of such methods. Representing polynomials as evaluation maps (straight line programs) can, if suitable programs are constructed, use much less space than using the monomial (dense) representation. A classical illustration is to symbolically represent a determinant as a sum of monomials versus more cheaply evaluating it numerically at a point. This line of research has been followed by Giusti, Heintz [9], their collaborators [11] and recently Lecerf [14]. They have obtained a probabilistic algorithm (the geometric resolution) which if the input system is successfully encoded as straight line programs, can solve systems with complexity that is a polynomial function of the geometric degree, and several quantities including the length of the straight line programs. Worst case non-polynomial complexity is still obtained for some systems.

The encoding of input polynomials being used in the papers of Sommese, Verschelde and Wampler currently represents polynomials in their (dense) monomial representation due to the interpolation phase. However as was mentioned in the previous section, the numerical irreducible decomposition can be described using generic points, *without the construction of interpolation polynomials* [36]. Thus the irreducible numerical decomposition method can take as input polynomial systems encoded as straight line programs. Moreover Newton methods, at the core of this method, can use methods such as automatic differentiation [7] to differentiate the programs, and implement the homotopy methods. Junk points can be rejected by a homotopy (and thus can again use a straight line encoding). These observations, form the basis of obtaining improved complexity estimates, analogous to those obtained in the works of Giusti, Heintz, and their collaborators.

6.3 Geometric completion of partial differential systems

First we note that Jet methods, were originally formulated for, and are in theory just equally applicable to systems of partial differential equations.

Consider a qth order system with: n independent variables $x = (x_1, x_2, ..., x_n)$, m dependent variables $u = (u^1, u^2, ..., u^m)$, first order derivatives $\underset{1}{u}$, second order derivatives $\underset{2}{u}$, etc. The formal total derivative is: $D_{x_j} = \frac{\partial}{\partial x_j} + \sum_l u^l_{x_j} \frac{\partial}{\partial u^l} + ...$ Thus we consider systems of total derivative order q, of form $R^1 = 0, ..., R^s = 0$, or more concisely $R = 0$, where $R^k : J^q \to \mathbb{C}$, $J^q = \mathbb{C}^{N_q}$. Here $N_q = n + m\begin{pmatrix} q+n \\ q \end{pmatrix}$, is the number of jet variables of order less than or equal to q. The jet variety of the system is:

$$V(R) := \{(x, u, \underset{1}{u}, ..., \underset{q}{u}) \in J^q : R^k(x, u, \underset{1}{u}, ..., \underset{q}{u}) = 0\}. \tag{28}$$

If all of the equations in the system have derivative order exactly q then a single symbolic prolongation of the system is taken to be:

$$D(R) := \{(x, u, \underset{1}{u}, ..., \underset{q}{u}) \in J^q : R^k = 0,\ D_{x_i} R^k = 0\}. \tag{29}$$

If there are equations with derivative order less than q then derivatives of these equations are appended to the system, and this process continued until no undifferentiated equations of lower order remain.

A single geometric projection is defined as:

$$\pi^q_{q-1}(V(R)) := \{(x, u, \underset{1}{u}, ..., \underset{q-1}{u}) \in J^{q-1} : R^k(x, u, \underset{1}{u}, ..., \underset{q-1}{u}, \underset{q}{u}) = 0\}. \tag{30}$$

We note that just as in the ODE case projection can be implemented by intersecting the jet variety of the system in J^q with random affine linear spaces in the jet variables of J^{q-1}.

Algorithm 3.1 is then easily adapted to partial differential systems. However in general it does not find all constraints, and the new phenomenon which must be taken into account of integrability conditions. The full method to complete systems of partial differential equations is the Cartan-Kuranishni algorithm [22, 27]. This method prolongs the system to order $q + 1$, then projects to order q to test for the existence of new constraints. This is continued until no new constraints are found. If the symbol of the resulting qth order system is involutive, then the method has terminated and the system is involutive. If the symbol is not involutive, the system is prolonged until its symbol becomes involutive. The system is again tested for the existence of constraints by prolongation and projection. See [22, 27] for definitions of symbol and the properties of involutivity of the system and its symbol.

The only new ingredient in the PDE case is to check the involutivity of the symbol, numerically. A probabilistic method to do this using Numerical Linear Algebra, and in particular the singular value decomposition, is given in [45, Section 6]. Numerical difficulties can occur, if there are multiplicities, and that case is under investigation.

6.4 Interpolation free completion algorithms

Although straight line encodings can be used in many parts of the geometric completion procedure described earlier in this article it can not be directly used in the interpolation phase.

A method that only uses prolongations (but not homotopy continuation for the termination tests) is presented in [45, Section 6]. We describe an interpolation free version of that method that uses homotopy continuation to check the termination criteria.

The usual termination tests which are based on the highest derivatives of a system do not always apply (see [45, Section 6, Example 4] for a case where they fail). As shown in [45] a projected version of those tests should be used to see when a projected system becomes involutive.

Checking dimensions, and existence of components in these projected systems, can be done without interpolation polynomials, as explained in the previous subsection. Checking for involutivity of the projected symbol, can be carried out when there are only multiplicity free components by using Numerical Linear Algebra, again without using interpolation polynomials.

The difficulty of applying this method to the case where components have multiplicity greater than 1 is that the symbolic derivative of an equation, may not be equivalent to a geometric derivative of an equation. This phenomenon is discussed in detail in [21, 22]. Simply speaking, for the symbolic algorithm to be a faithful representation of the underlying geometric algorithm, the ideals generated by the intermediate systems should be radical. This is a generalization of the algebra - geometry correspondence to differential systems. This difficulty does not occur for linear systems of partial differential equations. While it avoids the use of interpolation polynomials this method does have the disadvantage the number of jet variables is generally much larger than that in the method using interpolation.

7 Concluding Remarks

Systems of differential-algebraic equations are becoming increasingly important in applications, such as the control of constrained mechanisms. With constraints expressed by polynomials, these applications result in polynomially nonlinear differential systems with approximate coefficients.

Our paper presents the first of a new generation of methods to address the problem of developing an approximate differential-elimination geometric completion process to identify and include the missing constraints in such systems. The foundations of the new method consist of the geometrical theory of differential equations and the homotopy continuation methods of Sommese, Verschelde and Wampler for positive dimensional polynomial systems.

The missing constraints are computed via interpolation through generic points obtained by intersecting the jet submanifolds of the differential-algebraic system with sufficiently many random hyperplanes. Differentiation of the constraints is then performed by differentiating the interpolating polynomials. The calculations in this paper with the aid of PHCpack offer a first feasibility study, on a simple visualizable example and the classic index 3 pendulum. We get results equivalent to those obtained by the exact symbolic methods, but their representation differs (e.g. we obtain approximate linear combinations of the symbolically calculated constraints).

The new homotopy methods can handle systems with approximate coefficients to which symbolic algorithms cannot be applied. Moreover, the numerical irreducible decomposition of the solution set of a polynomial system is conceptually simpler than the primary decomposition of the ideal generated by the polynomials in the system. The symbolic algorithms to produce primary decompositions are notoriously complicated and difficult to implement (implementations do not exist in Mathematica and Maple). For partial differential systems, Hubert [12] has created factorization-free decomposition algorithms and Arponen has designed algorithms for analogues of primary decomposition for ordinary differential systems [3].

Any differential system becomes linear in its highest derivatives after differentiation. Thus in the development of efficient algorithms, as in the symbolic case, we anticipate the development of methods that use Numerical Linear Algebra for the highest derivatives[45], and the more expensive homotopy methods for the nonlinear lower order subsystems.

Just as numerical algebraic geometry is still in its infancy, numerical jet geometry is newly born. We believe that an exciting future lies ahead for

numerical jet geometry.

References

[1] E.L. Allgower and K. Georg. Numerical path following. In *Scientific Computing (Part 2)*, edited by P.G. Ciarlet and J.L. Lions, pages 3–203. Volume 5 of *Handbook of Numerical Analysis*, North-Holland, 1997.

[2] U.M. Ascher and L.R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations.* SIAM, 1998.

[3] T. Arponen. The complete form of a differential algebraic equation. *Helsinki Univ. of Tech. Inst. of Math. Research Rep.* Number A438, 2001. Available at `http://www.math.hut.fi/reports/`

[4] F. Boulier, D. Lazard, F. Ollivier, and M. Petitot. Representation for the radical of a finitely generated differential ideal. Proc. ISSAC 1995. ACM Press. 158–166, 1995.

[5] S.L. Campbell and C.W. Gear. The index of general nonlinear DAEs. *Numer. Math.* 72: 173–196, 1995.

[6] R.M. Corless, A. Galligo, I.S. Kotsireas, and S.M. Watt. A symbolic-geometric algorithm for factoring multivariate Polynomials. Proc. ISSAC 2002, to appear.

[7] G. Corliss, C. Faure, A. Griewank, L. Hascoët, and U. Naumann (eds.) *Automatic Differentiation 2000: From Simulation to Optimization.* Springer, New York, 2001.

[8] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms.* Springer, 1996 (2nd Edition).

[9] M. Giusti and J. Heintz. Kronecker's smart, little black boxes. In *Foundations of Computational Mathematics*, edited by R.A. DeVore, A. Iserles, and E. Süli. Volume 284 of the *London Mathematical Society Lecture Note Series*, Cambridge University Press, 2001.

[10] M. Hausdorf and W.M. Seiler. Perturbation versus differentiation indices. In *Computer Algebra in Scientific Computing—CASC'01*, edited

by V.G. Ganzha, E.W. Mayr and E.V. Vorozhtsov, pages 323–337. (Springer, Berlin) 2001.

[11] J. Heintz, T. Krick, S. Puddu, J. Sabia, and A. Waissbein. Deformation techniques for efficient polynomial equation solving. *J. Complexity* 16(1): 70–109, 2000.

[12] E. Hubert. Factorization free decomposition algorithms in differential algebra. *J. Symbolic Computation* 29: 641–662, 2000.

[13] C. Lanczos. Applied Analysis. Prentice Hall, 1956.

[14] G. Lecerf. *Une alternative aux méthodes de réécriture pour la résolution des systèmes algébriques.* Thèse de doctorat de l'École polytechnique, 2001.

[15] G. Le Vey. Some remarks on solvability and various indices for implicit differential equations. *Numer. Algorithms* 19: 127–145, 1998.

[16] T.Y. Li. Numerical solution of multivariate polynomial systems by homotopy continuation methods. *Acta Numerica* 6: 399–436, 1997.

[17] E. Mansfield. *Differential Gröbner Bases.* Ph.D. thesis, Univ. of Sydney, 1991.

[18] J.F. Pommaret. *Systems of Partial Differential Equations and Lie Pseudogroups.* Gordon and Breach Science Publishers, Inc. 1978.

[19] F. Pritchard and W. Sit. On initial value problems for ordinary DAEs. To appear in *J. Symbolic Computation.*

[20] M.P. Quéré and G. Villard. An algorithm for the reduction of linear DAE. Proc. ISSAC 1995. ACM Press. 223–231, 1995.

[21] G.J. Reid, A.D. Wittkopf and A. Boulton. Reduction of systems of nonlinear partial differential equations to simplified involutive forms. *Eur. J. of Appl. Math.* 7: 604–635.

[22] G.J. Reid, P. Lin, and A.D. Wittkopf. Differential elimination-completion algorithms for DAE and PDAE. *Studies in Applied Mathematics* 106(1): 1–45, 2001.

[23] P.J. Rabier and W.C. Rheinboldt. A geometric treatment of implicit differential-algebraic equations. *Journal of Differential Equations* 109: 110–146, 1994.

[24] S. Reich. *Beitrag zur Theorie der Algebrodifferentialgleichungen.* Ph.D. dissertation in Engineering, T.U. Dresden. 1989.

[25] C. Rust, G.J. Reid, and A.D. Wittkopf. Existence and uniqueness theorems for formal power series solutions of analytic differential systems. Proc. ISSAC 1999. ACM Press. 105–112, 1999.

[26] A. Sedoglavic. A probabilistic algorithm to test local algebraic observability in polynomial time. Proc. ISSAC 2001. ACM Press. 309–316, 2001.

[27] W.M. Seiler. *Analysis and application of the formal theory of partial differential equations.* Ph.D. thesis, Lancaster University, 1994.

[28] W.M. Seiler. Involution and constrained dynamics II: the Faddeev-Jackiw approach. *J. Phys. A.* 28:7315–7331, 1995.

[29] W.M. Seiler. Indices and solvability for general systems of differential equations. In *Computer algebra in scientific computing—CASC'99*, edited by V.G. Ganzha, E.W. Mayr and E.V. Vorozhtsov, pages 365–385. (Springer, Berlin) 1999.

[30] W.M. Seiler and R.W. Tucker. Involution and constrained dynamics I: the Dirac approach. *J. Phys. A.* 28:4431–4451, 1995.

[31] M. Shub and S. Smale. Complexity of Bezout's theorem V: Polynomial time. *Theoretical Computer Science* 133(1):141–164, 1994.

[32] A.J. Sommese and J. Verschelde. Numerical homotopies to compute generic points on positive dimensional algebraic sets. *J. Complexity* 16(3):572–602, 2000.

[33] 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.

[34] A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical irreducible decomposition using projections from points on the components. In *Symbolic Computation: Solving Equations in Algebra, Geometry, and Engineering*, Volume 286 of *Contemporary Mathematics*, edited by E.L. Green, S. Hoşten, R.C. Laubenbacher, and V. Powers, pages 37–51. AMS 2001.

[35] A.J. Sommese, J. Verschelde, and C.W. Wampler. Using monodromy to decompose solution sets of polynomial systems into irreducible components. In *Application of Algebraic Geometry to Coding Theory, Physics and Computation*, edited by C. Ciliberto, F. Hirzebruch, R. Miranda, and M. Teicher, 297–315, 2001. Proceedings of a NATO Conference, February 25 - March 1, 2001, Eilat, Israel. Kluwer Academic Publishers.

[36] A.J. Sommese, J. Verschelde, and C.W. Wampler. Symmetric functions applied to decomposing solution sets of polynomial systems. Accepted for publication in *SIAM J. Numer. Anal.*

[37] A.J. Sommese, J. Verschelde, and C.W. Wampler. Numerical irreducible decomposition using PHCpack. To appear in *Mathematics and Visualization*, edited by M. Joswig and N. Takayama. Springer-Verlag.

[38] A.J. Sommese, J. Verschelde, and C.W. Wampler. A method for tracking singular paths with application to the numerical irreducible decomposition. In *Algebraic Geometry, a Volume in Memory of Paolo Francia*, edited by M.C. Beltrametti, F. Catanese, C. Ciliberto, A. Lanteri, C. Pedrini. W. de Gruyter, to appear.

[39] A.J. Sommese, J. Verschelde, and C.W. Wampler. Advances in polynomial continuation for solving problems in kinematics. In *Proc. ASME Design Engineering Technical Conf.* (CDROM), Paper DETC2002/MECH-34254. Montreal, Quebec, Sept. 29-Oct. 2, 2002.

[40] A.J. Sommese and C.W. Wampler. Numerical algebraic geometry. In *The Mathematics of Numerical Analysis*, Volume 32 of *Lectures in Applied Mathematics*, edited by J. Renegar, M. Shub, and S. Smale, 749–763, 1996. Proceedings of the AMS-SIAM Summer Seminar in Applied Mathematics, Park City, Utah, July 17-August 11, 1995, Park City, Utah.

[41] G. Thomas. Symbolic computation of the index of quasilinear differential-algebraic equations. Proc. ISSAC 1996. ACM Press. 196–203, 1996.

[42] J. Tuomela and T. Arponen. On the numerical solution of involutive ordinary differential systems. *IMA J. Numer. Anal.* 20: 561–599, 2000.

[43] J. Verschelde. Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. *ACM Transactions on Mathematical Software* 25(2): 251–276, 1999. Software available at `http://www.math.uic.edu/~jan`.

[44] J. Visconti. *Numerical Solution of Differential Algebraic Equations, Global Error Estimation and Symbolic Index Reduction.* Ph.D. Thesis. Laboratoire de Modélisation et Calcul. Grenoble. 1999.

[45] A. Wittkopf and G.J. Reid. Fast differential elimination in C: The CDiffElim environment. *Computer Physics Communications*, 139: 192–217, 2001.

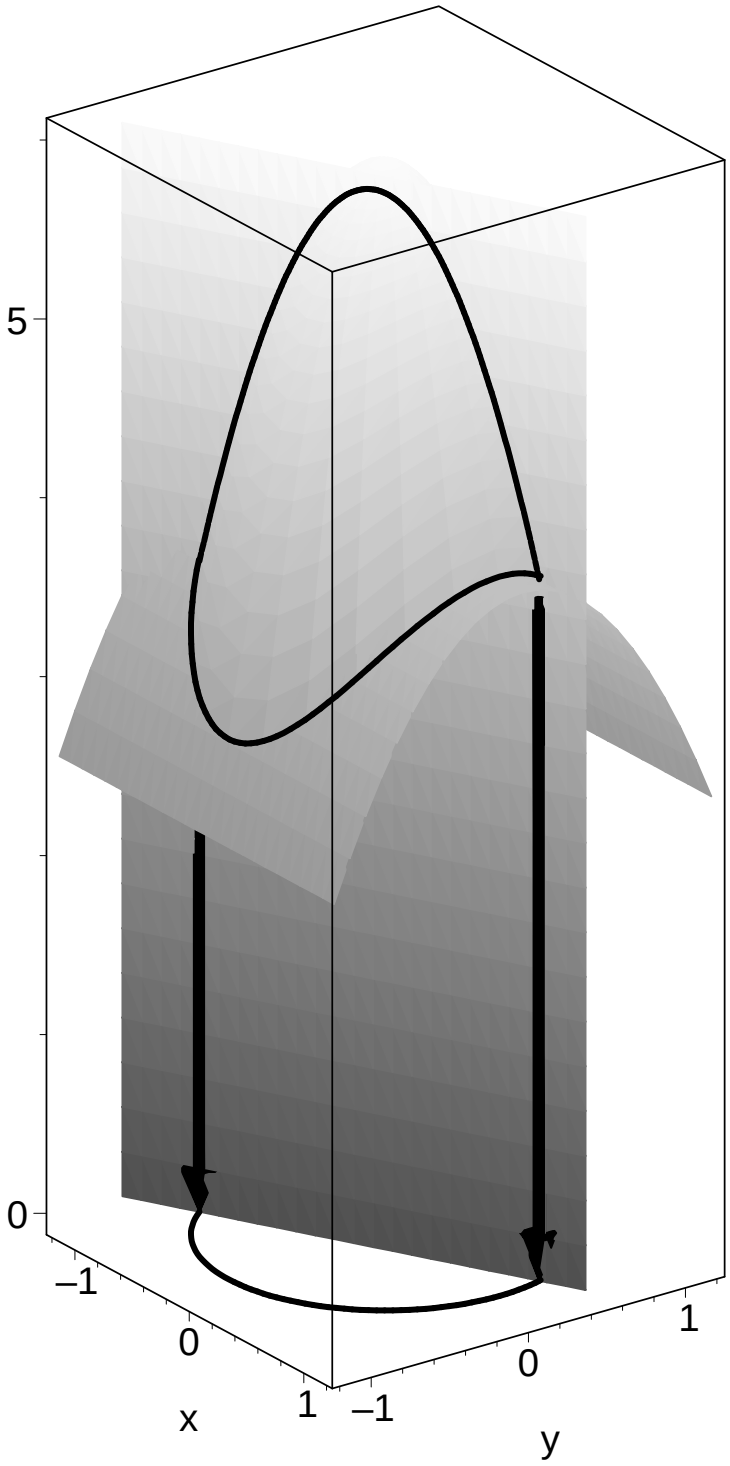

Figure 2: The varieties of $x_t + y^2 - 4 = 0$ (a parabolic cylinder), $x_t + 2x^2 + 3y^2 - 6 = 0$ (an elliptic cone), and their intersection (the saddle-shaped curve) together with its projection onto the x-y plane. Also shown is a random plane intersecting the saddle curve at two generic points.