An Extended Continuation Problem for Bifurcation Analysis in the Presence of Constraints

Harry Dankowicz
Department of Mechanical Science and Engineering
University of Illinois at Urbana-Champaign
Urbana, IL 61801
Email: danko@illinois.edu

Frank Schilder
Department of Mathematics
Technical University of Denmark
2800 Kgs. Lyngby
Denmark
Email: F.Schilder@mat.dtu.dk

ABSTRACT

This paper presents an extended formulation of the basic continuation problem for implicitly-defined, embedded manifolds in $\mathbb{R}^n$. The formulation is chosen so as to allow for the arbitrary imposition of additional constraints during continuation and the restriction to selective parametrizations of the corresponding higher-co-dimension solution manifolds. In particular, the formalism is demonstrated to clearly separate between the essential functionality required of core routines *in application-oriented continuation packages, on the one hand; and the functionality provided by* auxiliary toolboxes *that encode classes of continuation problems and* user-definitions *that narrowly focus on a particular problem implementation, on the other hand. Several examples are chosen to illustrate the formalism and its implementation in the recently developed continuation core package* COCO *and auxiliary toolboxes, including continuation of families of periodic orbits in a hybrid dynamical system with impacts and friction as well as detection and constrained continuation of selected degeneracies characteristic of such systems, such as grazing and switching-sliding bifurcations.*

1 Introduction

Continuation is a numerical technique for computing implicitly-defined manifolds that relies on the Implicit Function Theorem (IFT) and its constructive proof. Starting with a single chart, i.e., a point on the manifold together with a representation of the tangent space at this point, continuation employs a *covering algorithm* for computing nearby charts. The process is subsequently repeated for each of the nearby charts. The manifold through the initial point is called a *branch* or a *family* and the computed atlas of charts is a *covering* of this branch. General-purpose covering algorithms were first developed for the case of one-dimensional manifolds, the most successful one being the pseudo-arc length continuation method [1,2]. More recently, algorithms have been developed for manifolds of arbitrary dimension [3]. In this paper, we restrict the discussion of explicit examples to the one-dimensional case, which is implemented in all existing packages that employ continuation methods. The fundamental aspects of the presentation, however, are independent of the manifold dimension.

It is instructive to distinguish between three different layers of an application-oriented implementation of a continuation problem, a distinction that has been made in all modern continuation packages. At the *core* layer one finds the covering algorithm and other general-purpose tools that provide further useful functionality for continuation. The *toolbox* layer contains wrappers to the core that encode algorithms for solving specific classes of continuation problems as well as auxiliary toolboxes that provide additional functionality for these specific classes. As an example, a toolbox for computing and characterizing branches of periodic solutions of ordinary differential equations (ODEs) might make use of an auxiliary toolbox implementing a collocation method for two-point, boundary-value problems of ODEs. Finally, the outermost *user* layer of an implementation of a continuation problem contains user-provided functions and data that define a specific continuation problem, e.g., the continuation of periodic solutions of a given ODE.

The objective of this paper is to propose a novel core design. Compared to existing formulations, the proposed core allows greater flexibility to toolbox developers and more clearly distinguishes between the choices made in deploying a particular covering algorithm and the choices made in formulating a continuation problem. A central theme of the proposed design is the philosophy of an *extended continuation problem*, a mathematical formulation that naturally supports the idea of *task embedding*. Using a prototype for a continuation problem with arbitrarily large sets of additional algebraic constraints, namely, the continuation of a hybrid periodic orbit, this paper demonstrates that the extended formulation enables innovative computations that are not supported in a similar way as ‘built-in’ functionality by any existing core implementations. As argued below, other computations that would profit heavily from support for embedding are the continuation of connecting orbits using algorithms based on Lin’s method [4], the computation of Arnol’d tongue scenarios [5], and the continuation of relative periodic orbits [6].

A significant number of computational tools for continuation and bifurcation analysis of characteristic classes of solutions of dynamical systems have been developed in the past and have significantly guided the development effort presented in this paper. These include general algebraic and two-point boundary-value solvers for ordinary differential equations, such as AUTO [7] (and specialized drivers, such as HOMCONT [8], SLIDECONT [9], and TC-HAT [10]), CONTENT [11], MATCONT [12, 13], and SYMPERCON [14]; boundary-value solvers for delay differential equations, such as DDE-BIFTOOL [15] and PDDE-CONT [16]; and tools for large-scale systems, such as LOCA [17]. All these packages contain a continuation core and we borrowed as many ideas as we could.

The remainder of the paper is organized as follows. Section 2 presents the general continuation framework and highlights two common situations that motivate the proposed core design. A mathematical formulation of the extended continuation problem and its advantages is described in Secs. 3 and 4. The reference implementation of the design philosophy in the package COCO and its auxiliary toolboxes is detailed in Sec. 5. Section 6 presents an illustration of the application of the design philosophy and the COCO package to the constrained continuation of periodic trajectories in a hybrid dynamical system modeling a mechanical system with impacts and friction. Finally, a concluding discussion that points the way to further redesign at the toolbox level is presented in Sec. 7.

2 Fundamentals of continuation

Consider the non-linear equation

$$F(u) = 0, \quad F : \mathbb{R}^n \to \mathbb{R}^m, \quad d = n - m > 0, \tag{1}$$

and let $J := F_u(u^*) \in \mathbb{R}^{m\times n}$ denote the Jacobian of F at some solution point u^*, for which $F(u^*) = 0$. If all matrices composed of m columns of the Jacobian J have full rank, the IFT states that there exists a locally unique, d-dimensional solution manifold through u^* that can be parametrized by any d components of u. The condition on J can be weakened, in which case restrictions to the choice of local parametrizations arise.

As a simple example motivated by dynamical systems theory consider the so-called cusp normal form defined by the

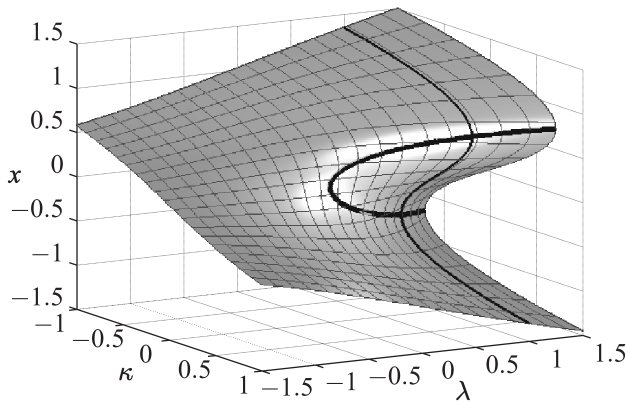

Fig. 1. Cusp normal form equilibrium manifold. Highlighted are a branch of equilibria (thin) and a branch of saddle-node bifurcation points (thick).

ODE

$$\dot{x} = \phi(x,\kappa,\lambda) := \kappa - x\left(\lambda - x^2\right) \tag{2}$$

together with the function

$$\psi(x,\kappa,\lambda) := \partial_x \phi(x,\kappa,\lambda) = 3x^2 - \lambda. \tag{3}$$

Roots of ϕ correspond to equilibrium points of ODE (2) for certain values of κ and λ, and the sign of ψ at such an equilibrium point indicates stability. If $\psi(x,\kappa,\lambda) = 0$ the equilibrium point is a so-called *saddle-node*. The set of equilibrium points forms a manifold in the three-dimensional (x,κ,λ)-space; see Fig. 1.

Now suppose that we desire a covering of a branch of saddle-node points. Given a point (x^*,κ^*,λ^*) on the equilibrium manifold with $\psi(x^*,\kappa^*,\lambda^*) \neq 0$, we may achieve this objective in two stages. In the first stage, continuation applied to the nonlinear function

$$F(x,\kappa) := \phi(x,\kappa,1) \tag{4}$$

yields a one-dimensional branch along the equilibrium manifold for $\lambda = 1$; see Fig. 1. To locate a saddle-node point we monitor the sign of $\psi(x,\kappa,1)$ along the computed branch, associating a change in sign with such a point. Having detected such a sign change, we now employ a subdivision algorithm to locate the saddle-node $\left(x^\#,\kappa^\#,1\right)$ as a simultaneous root of $\phi(x,\kappa,1)$ and $\psi(x,\kappa,1)$ with prescribed accuracy. In the second stage, continuation applied to the nonlinear function

$$F(x,\kappa,\lambda) := \begin{pmatrix} \phi(x,\kappa,\lambda) \\ \psi(x,\kappa,\lambda) \end{pmatrix} \tag{5}$$

and based at $\left(x^\#,\kappa^\#,1\right)$ yields the desired branch of saddle-node points; see Fig. 1.

This example illustrates two common situations. Firstly, continuation problems are usually formulated using more variables than are actually used for continuation. In Eqn. (4), the function F is a reduced formulation of the original problem, in which we interpret λ as a constant parameter. Secondly, continuation problems usually include the detection of *events* corresponding to zero-crossings of *monitor functions* or *test functions* and the subsequent continuation of branches of such *special points* by suitably augmenting the original formulation. This process of locating and continuing special points might be repeated, leading to the computation of curves of higher and higher co-dimension with arbitrary sets of monitor functions being added as additional constraints. As an example, in the first stage of continuation described above, the function $\psi(x,\kappa,1)$ played the role of a monitor function. In contrast, in the second stage, the equation $\psi(x,\kappa,\lambda) = 0$ instead played the role of an additional constraint. In fact, we can interpret the first continuation problem as a reduction of the second one; in Eqn. (4) we reduce the full problem (5) by one function and one variable.

The philosophy of constructing reduced continuation problems at the toolbox level has been followed in all implementations of general-purpose continuation packages to date. A direct consequence of this philosophy is that continuations of special points require the set-up of separate continuation problems for each type of point. From a programming point of view, it is obvious that this style of developing algorithms quickly reaches its limits. Just imagine the situation that we have k different monitor functions and want to implement all $\binom{k}{l}$ continuation problems for special points with l additional constraints.

A further consequence of this philosophy is that a toolbox developer has to decide in advance which variables should be used; which parameters can be included as variables; and what types of constraints might be allowed. Once this decision has been taken it is rather hard for end users – and might be even for toolbox developers themselves – to add further functionality. As we will see in Sec. 3, not only can we remove these limitations, but we can also simplify the development of toolboxes and their algorithms by using a single extended continuation problem instead of a large number of distinct reduced problems.

3 An extended continuation problem

The objective of this section is to develop an extended formulation that retains the full complexity of Eqn. (5) and that allows for user-defined restrictions that are equivalent to the two continuation problems discussed in Sec. 2.

To this end, consider the nonlinear function

$$F(x,\kappa,\lambda;\mu_1,\mu_2,\mu_3) := \begin{pmatrix} \phi(x,\kappa,\lambda) \\ \psi(x,\kappa,\lambda) \\ \kappa \\ \lambda \end{pmatrix} - \begin{pmatrix} 0 \\ \mu_1 \\ \mu_2 \\ \mu_3 \end{pmatrix}. \tag{6}$$

It is easy to verify that the restriction

$$F(x,\kappa,\lambda;\mu_1,\mu_2,\mu_3)\,|_{\{\mu_3=1\}} \tag{7}$$

is fully equivalent to the continuation problem corresponding to Eqn. (4). Here μ_1 and μ_2 assume the values of the monitor function $\psi(x,\kappa,1)$ and the variable κ, respectively, during continuation. Similarly, the restriction

$$F(x,\kappa,\lambda;\mu_1,\mu_2,\mu_3)\,|_{\{\mu_1=0\}} \tag{8}$$

is fully equivalent to the continuation problem corresponding to Eqn. (5). Here, μ_2 and μ_3 assume the values of the variables κ and λ, respectively, during continuation.

What did we gain here? The formulation in Eqn. (6) has extended the range dimension of the nonlinear function by two and the dimension of the domain by three as compared to Eqn. (5). At a first glance this looks more complicated than the traditional approach. However, this is not the case. The first and major improvement is somewhat subtle. Although the full continuation problem has been augmented by three additional variables μ_1, μ_2 and μ_3, the construction of the nonlinear function in Eqn. (6) demonstrates that these variables do not appear in the user or toolbox-specific formulations of the problem (which are restricted to defining ϕ and ψ and assigning the variables x, κ, and λ as problem "parameters", as discussed below). Hence, by introducing the auxiliary vector μ, the process of restricting the problem has been delegated from toolboxes to the core, where it can be handled in great generality and driven by run-time user definitions.

The second improvement is that the process of constructing a problem for a locus of special points has been transformed into a simple exchange of parameters. The only difference between Eqns. (7) and (8) is that we exchanged the restriction $\mu_3 = 1$ for $\mu_1 = 0$. It is not hard to imagine that this formalism can be extended to arbitrary numbers and combinations of monitor functions and constraints, making the set-up of involved continuation problems a trivial exercise.

A third improvement is that the dimension of the implicitly defined manifold is now determined by the core alone. Consequently, toolboxes do not need to be specifically designed for higher-dimensional covering algorithms. For example, the "empty restriction"

$$F(x,\kappa,\lambda;\mu_1,\mu_2,\mu_3)\,|_{\{\}} \tag{9}$$

could be used to compute a two-dimensional covering of the full equilibrium manifold without touching the underlying problem. Here, the points for which $\mu_1 = 0$ again mark saddle-node points.

Inspired by the above discussion we generalize this formalism to arbitrary dimension and an arbitrary number of monitor functions and constraints. Let $\Phi(u)=0$, $\Phi:\mathbb{R}^m\to\mathbb{R}^k$, be a set of *zero problems* and $\Psi(u)$, $\Psi:\mathbb{R}^m\to\mathbb{R}^l$, be a set of monitor functions. We call the equation

$$F(u;\mu):=\begin{pmatrix}\Phi(u)\\ \Psi(u)\end{pmatrix}-\begin{pmatrix}0\\ \mu\end{pmatrix} \tag{10}$$

the *extended continuation problem*. Let $\mathbb{I}\subseteq\{1,\ldots,l\}$ be an index set, and $\overline{\mathbb{I}}$ its complement in $\{1,\ldots,l\}$. Let $\mu_{\mathbb{I}}:=\{\mu_i\,|\,i\in\mathbb{I}\}$ and assume that the restriction $F(u;\mu)|_{\mu_{\mathbb{I}}=\mu^*_{\mathbb{I}}}$ satisfies the IFT at some point $(u^*,\mu^*=\Psi(u^*))$. Then, the restriction $F(u;\mu)|_{\mu_{\mathbb{I}}=\mu^*_{\mathbb{I}}}$ defines a continuation problem for a d-dimensional manifold with $d=m-(k+|\mathbb{I}|)$. We call $\mu_{\overline{\mathbb{I}}}$ the set of *active continuation parameters* and $\mu_{\mathbb{I}}$ the set of *inactive continuation parameters*. Active continuation parameters change during continuation while inactive continuation parameters remain constant. Similarly, an equation corresponding to an active continuation parameter is an *inactive constraint*, while an equation corresponding to an inactive continuation parameter is an *active constraint*, imposing an additional condition on the set of zero problems.

A toolbox would typically implement some algorithm Φ and add a set of monitor functions to Ψ. Since Ψ is defined at the level of the core, an end user might add additional monitor functions to Ψ as required. She could then select any combination of parameters to be active or inactive, corresponding to setting up different constrained continuation problems. In turn, a core implementation would have to provide algorithms for constructing extended continuation problems in a straightforward way, to select active and inactive continuation parameters, and to pass restricted continuation problems to a covering algorithm.

Note that in the case that the conditions of the IFT on the restriction $F(u;\mu)|_{\mu_{\mathbb{I}}=\mu^*_{\mathbb{I}}}$ are satisfied at (u^*,μ^*) with $|\mathbb{I}|=m-k$ (i.e., $d=0$), the inverse function theorem guarantees that the point (u^*,μ^*) is a locally unique root of $F(u;\mu)|_{\mu_{\mathbb{I}}=\mu^*_{\mathbb{I}}}$. In this case, the extended continuation problem can be combined with a recursive root-finding scheme to locate the point (u^*,μ^*) given an initial guess $u\approx u^*$ and $\mu\approx\mu^*$.

Finally, note that the idea of augmenting the set of independent variables is consistent with typical covering algorithms, in which the continuation problem is extended with a function $h(u;\mu,\lambda)$, $h:\mathbb{R}^m\times\mathbb{R}^l\times\mathbb{R}^d\to\mathbb{R}^d$ chosen so as to guarantee a non-singular Jacobian with respect to $(u,\mu_{\overline{\mathbb{I}}})$ at $u=u^*$, $\mu=\mu^*$, and $\lambda=0$.

4 Task embedding

A broader interpretation of the construction of the extended continuation problem is afforded by the notion of *task embedding*–a shared responsibility for the definition of Φ and Ψ across several toolboxes. According to this notion, functionality afforded by distinct toolboxes should be formulated in such a way that it can be combined to solve composite continuation problems without code modification or reimplementation.

To give a simple but non-trivial example, consider the continuation of relative periodic orbits of a dissipative ODE $\dot{x}=f(x,\lambda)$, $f:\mathbb{R}^n\times\mathbb{R}^m\to\mathbb{R}^n$, with a rotational symmetry generated by an n-by-n matrix ξ, that is, $\{\exp(\mathbb{R}\xi)\}\simeq S^1$. Using the unfolding $\dot{x}=f(x,\lambda)-\omega\xi x$, where ω is the rotational frequency of a co-rotating frame, this problem can be translated into a continuation problem for periodic solutions; see [6]. When using collocation methods, the only technical difference to periodic orbit continuation is the introduction of the additional integral constraint $\int_0^1\tilde{x}(\tau)\cdot\xi x(\tau)\,d\tau=0$ into the boundary value problem, where $\tilde{x}$ is a reference solution as usual, for example, from a previous continuation step, and the dot "$\cdot$" denotes the usual inner product in $\mathbb{R}^n$.

Given the similarities with the problem of continuation of periodic orbits, it is natural to ask whether existing continuation packages support the application of a toolbox for periodic orbit continuation, having all its built-in functionality available, in the service of a toolbox for continuation of relative periodic orbits. To our knowledge the answer is no; a toolbox developer or user would have to copy and modify an existing toolbox to implement the required functionality. Again, to our knowledge, existing packages for periodic orbit continuation either support the inclusion of an additional integral constraint only with an accompanying loss of functionality, or do not support this inclusion at all.

Why not?

At first one might argue that this is a different problem, hence the requirement for a re-development. However, a closer look at the example above reveals that, at least for generic constraints, a large number of algorithms carry over without any need for change, for example, the computation of loci of period-doubling- and Neimark-Sacker bifurcation points; some algorithms require small changes, for example, the formulation of test functions and the computation of loci of saddle-node bifurcation points; and a few algorithms need to be re-developed, for example, algorithms for branch-switching at period-doubling points.

In fact, it is possible to develop a set of algorithms for periodic orbits that can be *embedded* into other constrained continuation problems without modification, a project that is currently pursued by the authors in great generality. In our opinion, the reason that such algorithms have not been developed yet is that existing core-level algorithms do not support embedding in the first place. This state of affairs is resolved by the formulation of the extended continuation problem, whose construction is restricted to the core, but whose definition can be incrementally added to by external toolboxes and, at run-time, by the user. Toolbox algorithms developed specifically for a core based on the extended continuation problem formulation should thus emphasize specific tasks that support embedding, such as the construction of a collocation discretization of a two-point boundary-value problem, rather than specific problems.

5 Reference implementation

The above design philosophy has been implemented in a recently developed continuation package consisting of the core COCO and a number of associated toolboxes. The example in Sec. 6 illustrates the type of numerical analyzes that this package enables as well as the power of the extended formulation described above. In this section we will briefly describe technical details that are unique to COCO and important for toolbox developers and end users.

The source code for the reference implementation of COCO may be downloaded from the web [18]. It consists of two core components, which provide an interface for step-by-step construction of an extended continuation problem and for running a continuation. In particular, at run-time the output of the constructor algorithm is a restricted continuation problem of the form $F(x) = 0$, where x is a single vector containing all variables and active continuation parameters. Moreover, the continuation algorithm implements a one-dimensional covering algorithm and includes a powerful and flexible mechanism for locating and handling continuation-triggered events at the core level. Also included with the reference implementation is an example toolbox for continuation of solutions of systems of nonlinear, finite-dimensional equations, which is included as a tutorial example for toolbox developers.

5.1 Constructor algorithm

In our philosophy, the construction of a continuation problem consists of three tasks: i) adding functions to the sets Φ and Ψ of zero problems and monitor functions, respectively; ii) restricting the extended problem by defining the set of active continuation parameters; and iii) assigning events to specific values of the active continuation parameters. In practice, this is too restrictive, because it is not always desirable nor feasible to include a monitor function into the continuation problem.

For example, a natural test function for limit points of an algebraic continuation problem is the determinant of the Jacobian, which is usually computed using Gauss elimination [19]. There are several problems with this approach. Firstly, due to pivoting, this algorithm is only piecewise differentiable. Secondly, its derivative is hard to compute. Thirdly, including monitor functions that are products of eigenvalues into a continuation problem may lead to an ill-conditioned system [20]. Adding equations for all eigenvalues is also not an option, since pairs of eigenvalues of real matrices may undergo transitions from complex conjugate to real and vice versa. Each such transition point is a spurious limit point, beyond which the continuation would not proceed unless special precautions were taken.

The traditional way to deal with situations like this is to monitor the value of the determinant outside the continuation problem and to, subsequently, add equations for fixing one eigenvalue at zero to the continuation problem, if one wants to compute a locus of limit points. This functionality is implemented in COCO, which supports traditional monitor functions that are excluded from the continuation problem, as well as embedded monitor functions that are included with the continuation problem, making the latter available as additional constraints if feasible. As our example involving switching-sliding- and grazing-functions in Sec. 6 below demonstrates, there are many functions that naturally play the role of both, a monitor function or a constraint and one of the aims of our development is to simplify setting up such problems as much as possible.

In COCO we distinguish between four different types of functions, which we label as *zero, continuation, regular* and *singular functions*, respectively. Here, a zero function is included into the zero problem Φ; a continuation function is included into the set of monitor functions Ψ that are part of the extended continuation problem; and regular and singular functions are included into a set of monitor functions evaluated outside the continuation problem. This classification of monitor functions is closely related to the types of events they are associated with; see Sec. 5.2. A monitor function can be added to the set of continuation functions, if the extended continuation problem **including** the function generically satisfies the IFT along a branch and at special points associated with this function. A monitor function can be added to the set of regular functions, if the extended continuation problem **excluding** the function generically satisfies the IFT along a branch and at special points associated with this function. A monitor function that is neither a continuation nor a regular function must be added to the set of singular functions. Examples of continuation functions are given in Sec. 6. An example of a regular monitor function is the test function for limit points mentioned at the beginning of this section. Finally, an example of a singular monitor function is a test function for branch points, where two branches intersect.

Within the class of continuation functions we make a further distinction. A continuation function may be added as either *active*, *inactive*, or *internal*. Active and internal functions are added as monitor functions, whereas inactive functions are added as constraints. This initial allocation of functions can later be modified by exchanging parameters as discussed in Sec. 3. Internal continuation functions are introduced to enable an automatic exchange of parameters, which is used for simplifying frequently performed computations. An example, is the addition of the period of a periodic orbit as an internal function, which may be automatically constrained in the continuation of periodic solutions with fixed period.

The interface to the core components of COCO is fully accessible to both toolbox developers and end users, which may both add arbitrary sets of functions to all parts of a continuation problem. As discussed in the previous section, we refer to this process of combining functions as one of task embedding. As the description of COCO above illustrates, moving towards the philosophy of extended continuation problems enables a new way of developing algorithms. While COCO supports the traditional style of having a continuation problem defined as a single function plus a set of monitor functions, it also supports embedding functions into a continuation problem to solve a problem. In other words, algorithms or functions to be used with COCO may be developed having a specific task in mind instead of having a specific problem in mind. This is a non-trivial requirement and many algorithms will need some modification or even a re-development to support task embedding. However, we are convinced that this higher requirement is well worth the effort, because it simplifies computations that are only possible with significantly more effort using the traditional style for setting up continuation problems.

5.2 Covering algorithm and event handling

The one-dimensional covering algorithm is implemented in COCO as a finite-state machine (FSM). This is in alignment with MATLAB's support for event-driven software design and allows for easy integration into a graphical user interface and educational applications of continuation. Furthermore, transitions to a new state of the FSM occur at natural boundaries of the covering algorithm. Hence, it is possible to invoke callback functions before and after each state, thereby allowing additional code to be executed at well-defined points during a continuation. This is useful, for example, for intermediate graphical output or for saving the whole data content of the FSM at regular intervals during large-scale computations, so as to enable an immediate resumption of an interrupted computation from the last computed chart.

The covering algorithm tracks the values of all monitor functions and detects events as usual by a change of sign of these functions. Depending on the type of monitor function, an appropriate algorithm for locating the event is employed and an action is taken subsequent to successful location. In our implementation of the covering algorithm, we use two overlapping classifications of events, one distinguishing events according to the action to be taken, and another one according to the specific algorithm employed for locating the event. In the first classification we distinguish between *boundary*, *terminal*, and *special events*; in the second one between *continuation*, *regular*, *singular*, and *group events*.

A boundary event occurs if the continuation crosses a computational boundary of the branch. Our implementation locates the boundary point, includes it in the atlas and discards points outside the computational domain. A typical example of a boundary event is the restriction of a continuation to a finite parameter interval. A terminal event is an event that indicates that a newly computed point is unacceptable for some reason or requires an interruption of the continuation algorithm. Our implementation does not locate events of this type, but includes the last successfully computed point in the atlas and terminates the continuation in the current direction. A typical example is given by an event that occurs if some approximation error exceeds a user-prescribed limit. A special event is an event that is neither a boundary nor a terminal event. Our implementation locates the special point, includes it in the atlas and proceeds with the continuation. The computation of events proceeds in the order indicated, first we locate boundary events, then we check if a terminal event occurred, and finally we locate all special events.

Continuation, regular, and singular events are special points associated with a specific value of exactly one monitor function of type continuation, regular or singular, respectively. Our implementation locates continuation events by applying Newton's method directly to a restriction of the extended continuation problem that has the special point as a unique solution. It locates regular events by applying a combined subdivision-Newton method along the branch. Finally, singular events are located by subdivision along an interpolation of the branch.

A group event is a collection of more than one event. A group event is detected if at least one event in the collection is detected. Since a group event is assumed to be the simultaneous occurrence of all events in a group, it is treated as singular, since several events do not generically occur at the same point. To locate a group event following detection, a multi-valued subdivision algorithm along an interpolation of the branch is used to locate all apparently simultaneously occurring events and a heuristic is then used to decide if all events of the group did indeed occur simultaneously. As an example, branch points can be detected as a simultaneous zero of two monitor functions.

When assigning events to monitor functions, COCO supports the specification of a function that allows a toolbox developer or user to change the default behavior of the event handling implemented in the covering algorithm. This function communicates with the event handling algorithms via a reverse communication protocol. In a simple application, one could distinguish between sub- and super-critical bifurcation points. In a more sophisticated application, one can successively locate hierarchical subsets of events of a group event. An example is a group event for a branch point, where one monitor function is identical to the test function for limit points and another one tests for a degeneracy of the problem. Since the test function for limit points also equals zero at a branch point, one would test for a branch point first and locate a limit point if this test fails. Depending on the type of an event from a group, our implementation will choose an appropriate algorithm from the list above for locating each sub-event.

6 A multi-point boundary-value problem

The results below illustrate the application of a basic COCO-compatible toolbox for the continuation of periodic solutions in hybrid dynamical systems to an example mechanism originally studied in Svahn & Dankowicz [21]. The discussion partially revisits a bifurcation analysis performed in Thota & Dankowicz [10] and used there to illustrate the FORTRAN-based toolbox $\widehat{\text{TC}}$ (TC-HAT) written by the authors to implement continuation of multi-segment periodic trajectories in AUTO 97 (see also [22]). Indeed, the development of $\widehat{\text{TC}}$ illustrates many of the shortcomings of existing continuation packages outlined in Sec. 2.

As an example, the AUTO 97 core includes a collocation method that implements a two-point, boundary-value problem formulation for ODEs. While this is adequate for single-segment trajectories, its use for continuation of multi-segment trajectories requires identical time partitions across each of the trajectory segments. As the existing implementation is closely integrated with the core, its replacement with a multi-point, boundary-value formulation would require significant recoding of the core. In contrast, the results below have been obtained through the use of a COCO-compatible, auxiliary, multi-point, boundary-value-problem toolbox that allows for segment-specific partitions.

More specific to the issues raised in previous section, implementing $\widehat{\text{TC}}$ required the addition of a further reduced continuation problem in the AUTO 97 executable that would handle continuation of specific discontinuity-induced bifurcations, e.g., grazing bifurcations and switching-sliding bifurcations. In contrast, the extended formulation described in Sec. 3 reduces this to the restriction of the appropriately formulated extended continuation problem by fixing various components of μ. As a result, it is a trivial exercise to choose, at run-time, to perform a co-dimension-two continuation of a grazing-switching-sliding bifurcation curve. In this case, the additional boundary conditions are applied at the terminal points of different solution segments. Since the $\widehat{\text{TC}}$ implementation was written only to handle a single additional boundary condition at the terminal point of the first solution segment, such a continuation would not be possible with $\widehat{\text{TC}}$ without the creation of a new reduced continuation problem.

The implementation of the COCO-compatible toolbox for the continuation of periodic solutions in hybrid dynamical systems illustrates the notion of task embedding described above. Here, the collocation part of the extended continuation problem is added to the zero-problem Φ by the COCO-compatible, auxiliary, multi-point, boundary-value-problem toolbox, whereas the hybrid periodic solution toolbox adds to Φ inter-segment boundary conditions as well as the condition of a closed trajectory. The latter toolbox further adds monitor functions to Ψ given by the system parameters, including the unknown segment times-of-flight. Finally, bifurcation conditions associated with grazing and switching-sliding are added to Ψ by the user.

6.1 Numerical results

Following Thota & Dankowicz [10], let q and

$$q_c(t) := -b + a\sin\omega t \tag{11}$$

for $a, b, \omega > 0$ denote the lateral displacements of a mass along a rough surface and of a harmonically oscillating unilateral constraint, respectively, relative to the reference position of a linear restoring force of stiffness k applied to the mass; see Fig. 2. Let

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} := \begin{pmatrix} q \\ \dot{q} \\ \omega t \bmod 2\pi \end{pmatrix} \in \mathbb{R}^2 \times \mathbb{S}^1 \tag{12}$$

represent the state of the corresponding dynamical system. A solution trajectory then consists of a sequence of smooth curve segments in state space, such that

- the i-th solution segment is governed by a smooth vector field $\mathbf{f}_i$;
- the i-th solution segment terminates on the zero-level surface of a smooth event function h_i; and
- the terminal point on the i-th solution segment is mapped to the initial point on the subsequent segment through an associated smooth jump function $\mathbf{g}_i$.

Each solution segment is thus characterized by the triplet $\{\mathbf{f}_i, h_i, \mathbf{g}_i\}$. Finally, the overall *signature* Σ of the solution trajectory is the sequence of corresponding triplets.

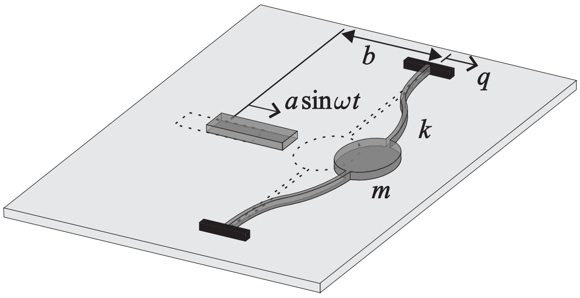

Fig. 2. Schematic of a mass m resting against a rough substrate and acted upon by a linear restoring force with stiffness k and collisional interactions with a harmonically oscillating unilateral constraint.

Specifically, consider the vector fields

$$\mathbf{f}_{\text{positive slip}}(\mathbf{x}) := \begin{pmatrix} x_2 \\ (-F_f - kx_1)/m \\ \omega \end{pmatrix}, \tag{13}$$

$$\mathbf{f}_{\text{negative slip}}(\mathbf{x}) := \begin{pmatrix} x_2 \\ (F_f - kx_1)/m \\ \omega \end{pmatrix}, \tag{14}$$

and

$$\mathbf{f}_{\text{stick}}(\mathbf{x}) := \begin{pmatrix} 0 \\ 0 \\ \omega \end{pmatrix}, \tag{15}$$

such that the mass is in positive slip when $x_2 > 0$ or when $x_2 = 0$ and $kx_1 < -F_f$; in negative slip when $x_2 < 0$ or when $x_2 = 0$ and $kx_1 > F_f$; and in stick otherwise. Here, F_f is the magnitude of the dry friction force between the mass and the substrate. We omit from consideration the possibility of sustained contact of the mass with the unilateral constraint.

Let the onset of contact between the mass and the unilateral constraint be characterized by the vanishing of the event function

$$h_{\text{impact}}(\mathbf{x}) := x_1 - q_c(x_3). \tag{16}$$

such that the associated jump function is given by

$$\mathbf{g}_{\text{impact}}(\mathbf{x}) := \begin{pmatrix} x_1 \\ -ex_2 + (1+e)\,\omega q_c'(x_3) \\ x_3 \end{pmatrix}, \tag{17}$$

where e is a kinematic coefficient of restitution.

Moreover, characterize discontinuous changes in the vector field by the vanishing of the event functions

$$h_{\text{stick}\pm}(\mathbf{x}) := \pm x_2. \tag{18}$$

with the associated jump function

$$\mathbf{g}_{\text{identity}}(\mathbf{x}) := \mathbf{x}. \tag{19}$$

Similarly, to accommodate the restriction of x_3 to $\mathbb{S}^1$, associate the state jump function

$$\mathbf{g}_{\text{phase}}(\mathbf{x}) := \begin{pmatrix} x_1 \\ x_2 \\ 0 \end{pmatrix} \tag{20}$$

with the event function

$$h_{\text{phase}}(\mathbf{x}) := 2\pi - x_3. \tag{21}$$

Finally, for purposes of detection of grazing, zero-relative-velocity contact between the mass and the unilateral constraint, consider the event function

$$h_{\text{turning}}(\mathbf{x}) := x_2 - \omega q_c'(x_3) \tag{22}$$

and the associated state jump function $\mathbf{g}_{\text{identity}}$.

For $m = 1$, $F_f = 0.7961$, $k = 5.5$, $\omega = 1$, $a = 1$, $b = 0.8471$, and $e = 0.9$, there exists a periodic solution trajectory of the hybrid dynamical system based at the initial condition $\mathbf{x} = (-0.1332, 0, 0)^T$ and characterized by the signature $\Sigma = \{\Sigma_i\}_{i=1}^5$, where

$$\Sigma_1 = \left\{\mathbf{f}_{\text{stick}}, h_{\text{impact}}, \mathbf{g}_{\text{impact}}\right\} \tag{23}$$
$$\Sigma_2 = \left\{\mathbf{f}_{\text{positive slip}}, h_{\text{stick}+}, \mathbf{g}_{\text{identity}}\right\} \tag{24}$$
$$\Sigma_3 = \left\{\mathbf{f}_{\text{negative slip}}, h_{\text{turning}}, \mathbf{g}_{\text{identity}}\right\} \tag{25}$$
$$\Sigma_4 = \left\{\mathbf{f}_{\text{negative slip}}, h_{\text{stick}-}, \mathbf{g}_{\text{identity}}\right\} \tag{26}$$
$$\Sigma_5 = \left\{\mathbf{f}_{\text{stick}}, h_{\text{phase}}, \mathbf{g}_{\text{phase}}\right\}. \tag{27}$$

Solution trajectories with this signature are compatible with the physics of the underlying mechanical system provided that h_{impact} is positive at the terminal point of the third segment and that $h_{\text{switching}}(\mathbf{x}) := kx_1 + F_f$ is positive at the terminal point of the fourth segment.

Now consider the extended continuation problem whose zero-problem corresponds to the collocation implementation of the multi-point, boundary-value problem associated with a periodic trajectory of the above signature. To ensure compatibility, add to the core monitor functions the event functions h_{impact} and $h_{\text{switching}}$ restricted to the terminal points of the third and fourth segment, respectively. Here, u consists of the components of the state vector along a discretization of the periodic solution trajectory together with the system parameters m, F_f, k, ω, a, b, and e. To enable higher-co-dimension constrained continuation, the latter are also added to the core monitor functions.

Figure 3 shows the result of continuation applied to the restriction of the extended continuation problem obtained by fixing the values of those components of μ corresponding to m, F_f, ω, a, b, and e. In particular, only those points with $k < 5.573$ are compatible with the condition on the terminal point on the fourth segment. As seen in Fig. 4, the fourth segment of the state-space trajectory found when $k \approx 5.573$ terminates on the boundary between initial conditions that result in a subsequent phase of stick (as in the original signature) and initial conditions that results in a subsequent phase of positive slip. The point with $k \approx 5.573$ is referred to in the literature as a switching-sliding bifurcation [23].

Figure 5 shows the results of two separate stages of continuation applied to distinct restrictions of the extended continuation problem. In the first stage, continuation is applied to a restricted continuation problem obtained by fixing the values of those components of μ corresponding to the restriction of $h_{\text{switching}}$ to the terminal point of the fourth segment and m, F_f, ω, a, and e, respectively, and based at the critical trajectory found for $k \approx 5.573$ (solid dot in Figs. 3 and 5). Of the trajectories found along the resultant switching-sliding bifurcation curve (the solid curve in Fig. 5), only those for which $b > 0.6614$ are compatible with the condition on the terminal point of the third segment. As seen in Fig. 6, the third segment of the state-space trajectory found when $k \approx 3.448$ and $b \approx 0.6614$ terminates at a tangential, grazing intersection with the surface $h_{\text{impact}} = 0$ while the fourth segment again terminates on the boundary between initial conditions that result in a subsequent phase of stick and initial conditions that results in a subsequent phase of positive slip.

Indeed, in the second stage, continuation is applied to a restricted continuation problem obtained by fixing the values of those components of μ corresponding to the restriction of h_{impact} to the terminal point of the third segment and m, F_f, ω, a, and e, respectively, and based at the critical trajectory found in the previous stage for $b \approx 0.6614$. The resultant grazing bifurcation curve is represented by the dashed curve in Fig. 5.

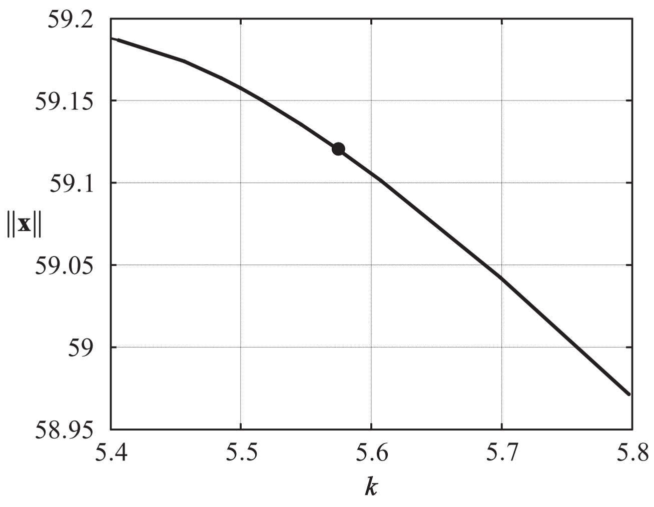

Fig. 3. Branch of periodic trajectories of the hybrid dynamical system of a given signature under variations in k. Only those points on the branch to the left of and including the point with $k \approx 5.573$ satisfy the compatibility conditions.

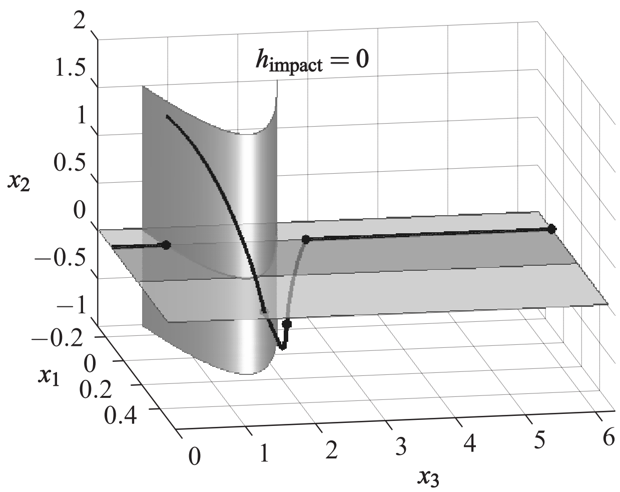

Fig. 4. A five-segment state-space trajectory obtained when $k \approx 5.573$. Note in particular the termination of the second negative slip segment on the boundary between initial conditions that result in a subsequent phase of stick and initial conditions that result in a subsequent phase of positive slip.

Continuation applied to the restricted continuation problem obtained by fixing the values of those components of μ corresponding to the restriction of h_{impact} to the terminal point of the third segment, the restriction of $h_{\text{switching}}$ to the terminal point of the fourth segment, and m, F_f, a, and e, respectively, and based at the critical trajectory found in the previous stage for $b \approx 0.6614$ yields the thick solid curve in Fig. 7. The figure includes the switching-sliding (thin solid) and grazing (dashed) bifurcation curves obtained previously as well as their continuations under variations in ω.

7 Conclusions

Fully developed continuation algorithms are quite complex. Not only is it necessary to provide methods for the computation of a covering of a solution manifold, but also methods for locating and handling events, and an interface to access individual families of solutions. We have seen that established implementations have a number of shortcomings in addressing these tasks. We have shown that we can remove these deficiencies by consistently following the philosophy of an extended continuation problem discussed in this paper. A further advantage of this new formulation is that we can allocate a large number of standard tasks to the core layer of continuation tools. With the MATLAB package COCO we provide a reference implementation, which was used for all examples in this paper.

The key improvement offered by our proposed approach is, however, that it allows for new and innovative computations

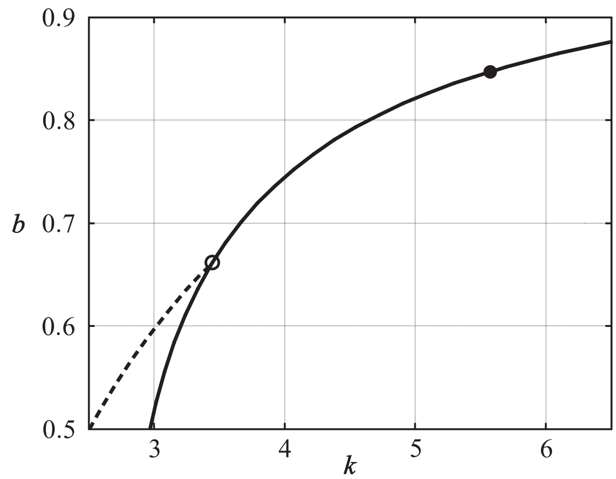

Fig. 5. Two-parameter switching-sliding (solid) and grazing (dashed) bifurcation curves under simultaneous variations in k and b. Of the points on the switching-sliding curve only those above and including the point with $k \approx 3.448$ and $b \approx 0.6614$ satisfy the compatibility conditions.

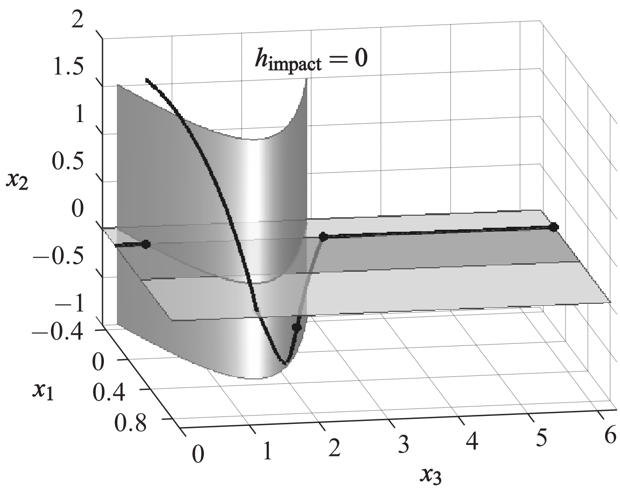

Fig. 6. A five-segment state-space trajectory obtained when $k \approx 3.448$ and $b \approx 0.6614$. Note in particular that the first negative slip segment terminates at a grazing intersection with the surface $h_{\mathrm{impact}} = 0$ while the fourth segment again terminates on the boundary between initial conditions that result in a subsequent phase of stick and initial conditions that result in a subsequent phase of positive slip.

in a straightforward way. Consider the classical problem of continuation and bifurcation analysis of periodic orbits of ODEs. A traditional implementation would compute a branch of periodic orbits as solutions of a boundary value problem (BVP). After each continuation step it would evaluate a set of natural test functions in terms of the Floquet multipliers to detect certain bifurcation points, for example, a Neimark-Sacker bifurcation point. To allow for the computation of loci of such bifurcation points, a traditional implementation would augment the original BVP by

(i) a pair of ODEs defining two solutions to the corresponding variational equation;
(ii) a pair of coupled boundary conditions constraining the initial values of these solutions to correspond to the real and imaginary parts of a pair of complex conjugate eigenvectors of the flow Jacobian; and
(iii) a constraint to fix the two corresponding multipliers on the unit circle.

Typically, if the dimension of the ODE is n, this augmented BVP therefore has a solution of dimension $3n$ and requires significantly more computation time to solve than the BVP for the periodic orbit alone. In general, a traditional implementation will only support the computation of loci of bifurcation points for which such an augmented (commonly referred to as an extended) BVP is implemented.

In stark contrast to this, consider the situation that an algorithm for periodic orbits of ODEs is developed in the spirit of our philosophy of extended continuation problems. Such an algorithm would not only compute a periodic orbit, but

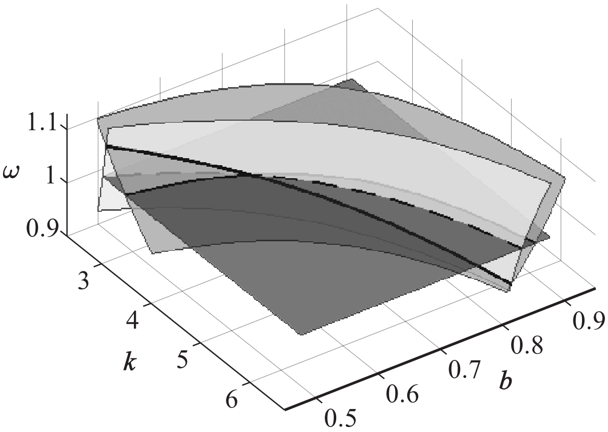

Fig. 7. Three-parameter switching-sliding/grazing (thick solid) bifurcation curve and switching-sliding and grazing bifurcation surfaces under simultaneous variations in k, b, and ω. Here, the switching-sliding (solid) and grazing (dashed) bifurcation curves are the intersection of two-dimensional bifurcation surfaces with the $\omega = 1$ plane.

also as much additional and potentially useful information as possible, for example, the complete solution of its variational equation. Interesting enough, this philosophy does not contradict efficiency as the recent developments [24–27] show. Here, the computation time required for computing a periodic orbit together with the full spectrum of its variational equation is comparable with the computation time required to compute just the periodic solution using traditional BVP methods. Embedding such algorithms into an extended continuation problem makes the set-up of continuation problems for curves of arbitrarily high co-dimension an almost trivial exercise.

To return to our above example, not only could one compute a locus of Neimark-Sacker points using the same algorithm as for the periodic solution (by simply imposing versions of (ii) and (iii) above), but this would also require no more computation time than computing a periodic orbit to begin with; as opposed to the increase in computation time experienced when using augmented (extended) BVP problems. Similarly, the continuation of a branch of co-dimension-two bifurcation points, arising for example from the simultaneous crossing of the unit circle of a pair of complex conjugate multipliers and a single real multiplier at -1, would be accomplished by exchanging inactive parameters for corresponding (active) multipliers. In light of this discussion, we believe that the philosophy of extended continuation problems will spark the development of new classes of algorithms that make additional information available at little or no cost and enable computations that are hard or impossible to perform when following the philosophy of reduced continuation problems.

8 Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant nos. 0237370 (HD) and 0635469 (HD) and by the Engineering and Physical Sciences Research Council under Grant no. EP/D063906/1 (FS).

References

[1] Keller, H.B., 1977, “Numerical Solution of Bifurcation and Nonlinear Eigenvalue Problems,” in Applications of bifurcation theory (Proc. Advanced Sem., Univ. Wisconsin, Madison, Wis., 1976), pp. 359–384. Publ. Math. Res. Center, No. 38, Academic Press, New York.

[2] Menzel, R., and Schwetlick, H., 1978, “Zur Lösung Parameterabhängiger Nichtlinearer Gleichungen mit Singulären Jacobi-Matrizen,” (German) Numer. Math., **30(1)**, pp. 65–79.

[3] Henderson, M.E., 2002, “Multiple parameter continuation: computing implicitly defined *k*-manifolds,” Internat. J. Bifur. Chaos Appl. Sci. Engrg., **12(3)**, pp. 451–476.

[4] Krauskopf, B., and Riess, Th., 2008, “A Lin’s Method Approach to Finding and Continuing Heteroclinic Connections Involving Periodic Orbits,” Nonlinearity, **21(8)**, pp. 1655-1690.

[5] Schilder, F., and Peckham, B.B., 2006, “Computing Arnol’d Tongue Scenarios,” J. Comput. Phys., **220(2)**, pp. 932–951.

[6] Wulff, C., and Schilder, F., 2009, “Numerical Bifurcation of Hamiltonian Relative Periodic Orbits,” SIAM Journal on Applied Dynamical Systems, **8(3)**, pp. 931-966.

[7] Doedel, E., Champneys, A., Fairgrieve, T., Kuznetsov, Y., Sandstede, B., and Wang, X., 1997, “AUTO97: Continuation and Bifurcation Software for Ordinary Differential Equations (with HomCont),” Technical report, Concordia University.

[8] Champneys, A.R., Kuznetsov, Y.A., and Sandstede, B., 1996, “A Numerical Toolbox for Homoclinic Bifurcation Analysis,” Internat. J. Bifur. Chaos Appl. Sci. Engrg., **6(5)**, pp. 867–887.
[9] Dercole F., and Kuznetsov, Y.A., 2005, “SLIDECONT: An AUTO97 Driver for Bifurcation Analysis of Filippov Systems,” ACM Trans. Math. Software, **31(1)**, pp. 95–119.
[10] Thota, P., and Dankowicz, H., 2008, “TC-HAT ($\widehat{\text{TC}}$): A Novel Toolbox for the Continuation of Periodic Trajectories in Hybrid Dynamical Systems,” SIAM J. App. Dyn. Syst., **7(4)**, pp. 1283–1322.
[11] Kuznetsov, Y.A., and Levitin, V.V., 1997, “CONTENT: A Multiplatform Environment for Continuation and Bifurcation Analysis of Dynamical Systems.” Technical report, Centrum voor Wiskunde en Informatica, available from ftp://ftp.cwi.nl/pub/CONTENT.
[12] Govaerts, W., Kuznetsov, Y.A., and Dhooge, A., 2005, “Numerical Continuation of Bifurcations of Limit Cycles in MATLAB,” SIAM J. Sci. Comput., **27(1)**, pp. 231–252.
[13] Dhooge, A., Govaerts, W., and Kuznetsov, Y.A., 2003, “MATCONT: A MATLAB Package for Numerical Bifurcation Analysis of ODEs,” ACM Trans. Math. Software, **29(2)**, pp. 141–164.
[14] Wulff, C., and Schebesch, A., 2006, “Numerical Continuation of Symmetric Periodic Orbits,” SIAM J. Appl. Dyn. Syst., **5(3)**, pp. 435–475.
[15] Engelborghs, K., Luzyanina, T., and Roose, D., 2002, “Numerical Bifurcation Analysis of Delay Differential Equations using DDE-BIFTOOL,” ACM Trans. Math. Software, **28(1)**, pp. 1–21.
[16] Szalai, R., Stepan, G., and Hogan, S.J., 2006, “Continuation of Bifurcations in Periodic Delay-Differential Equations using Characteristic Matrices,” SIAM J. Sci. Comput., **28(4)**, pp. 1301–1317.
[17] Salinger, A.G., Burroughs, E.A., Pawlowski, R.P., Phipps, E.T., and Romero, L.A., 2005, “Bifurcation Tracking Algorithms and Software for Large Scale Applications,” J. Bifur. Chaos Appl. Sci. Engrg., **15(3)**, pp. 1015–1032.
[18] http://sourceforge.net/projects/cocotools/
[19] Kuznetsov, Y.A., 2004, Elements of Applied Bifurcation Theory, 3rd ed., Springer-Verlag New York, Inc., Section 10.2.2.
[20] Higham, D.J., and Higham, N.J., 2005, MATLAB Guide, 2nd ed., Society for Industrial and Applied Mathematics, Chapter 5.
[21] Svahn, F., and Dankowicz, H., 2008, “Energy Transfer in Vibratory Systems with Friction Exhibiting Low-Velocity Collisions,” J. Vibr. Contr., **14(1-2)**, pp. 255–284.
[22] Kang, W., Thota, P., Wilcox, B., and Dankowicz, H., 2009, “Bifurcation Analysis of a Microactuator Using a New Toolbox for Continuation of Hybrid System Trajectories,” ASME J. Comp Nonlin. Dyn., **4(1)**, pp. 011009-1–8.
[23] di Bernardo, M., Kowalczyk, P., and Nordmark, A., 2002, “Bifurcations of Dynamical Systems with Sliding: Derivation of Normal-Form Mappings,” Physica D, **170**, pp. 175-205.
[24] Haro, A., and de la Llave, R., 2006, “A Parametrization Method for the Computation of Invariant Tori and their Whiskers in Quasi-periodic Maps: Rigorous Results,” J. Diff. Eqs., **228(2)**, pp. 530–579.
[25] Haro, A., and de la Llave, R., 2006, “A Parametrization Method for the Computation of Invariant Tori and their Whiskers in Quasi-periodic Maps: Numerical Algorithms,” Discrete Contin. Dyn. Syst. Ser. B, **6(6)**, pp. 1261–1300.
[26] Haro, A., and de la Llave, R., 2007, “A Parametrization Method for the Computation of Invariant Tori and their Whiskers in Quasi-periodic Maps: Explorations and Mechanisms for the Breakdown of Hyperbolicity,” SIAM J. Appl. Dyn. Syst., **6(1)** ,pp. 142–207.
[27] Moore, G., 2005, “Floquet Theory as a Computational Tool,” SIAM J. Numer. Anal., **42(6)**, pp. 2522–2568.