European Society of Computational Methods in Sciences and Engineering (ESCMSE)

Journal of Numerical Analysis, Industrial and Applied Mathematics (JNAIAM)
vol. 16, no. 1-2, 2022, pp. 1-13
ISSN 1790–8140

A numerical continuation method for parameter-dependent boundary value problems using `bvpsuite 2.0`

Winfried Auzinger[a], Katrina N. Burdeos[d], Merlin Fallahpour[c], Othmar Koch[b], Renier G. Mendoza[d], Ewa B. Weinmüller[a]

[a]*Institute of Analysis and Scientific Computing, TU Wien, Wiedner Hauptstrasse 8-10/E101, A-1040 Wien, Austria*

[b]*Wolfgang Pauli Institute, Oskar Morgenstern-Platz 1, A-1090 Wien, Austria*

[c]*Department of Mathematics and Computer Science, University of Basel, Spiegelgasse 1, 4051 Basel, Switzerland*

[d]*Institute of Mathematics, University of the Philippines Diliman, Quezon City,Philippines, 1101*

Abstract: The Matlab package `bvpsuite 2.0` is a numerical collocation code for the approximation of solutions of a broad range of boundary value problems in ordinary differential equations. In this article, its newly implemented pathfollowing module with automated step-length control is presented. Two versions using the pseudo-arclength continuation method, allowing pathfollowing beyond turning points, were developed, both taking advantage of the existing features of `bvpsuite 2.0` such as error estimation and mesh adaptation. The first version is based on the Gauss-Newton method. The second version is now contained in the package `bvpsuite 2.0` and uses its built-in iterative method, the Fast Frozen Newton method. Their operating principles are presented and their performance is compared by means of the computation of some pathfollowing problems. Furthermore, the results of computations with `bvpsuite 2.0` for a problem with path bifurcations are presented.

1. Introduction

A *continuation method* refers to the process of computing successive points on a solution curve, usually of a parameter-dependent system of nonlinear equations of the form

$$\boldsymbol{F}(\boldsymbol{x};\lambda) = 0, \tag{1}$$

where $F\colon D \subset \mathbb{R}^d \times \mathbb{R} \longrightarrow \mathbb{R}^d$, $d \in \mathbb{N}$ [1]. Here, $\boldsymbol{x}$ is the unknown variable and λ is the pathfollowing parameter.

`bvpsuite 2.0`[1] is a software package for the numerical solution of boundary value problems (BVPs) in ordinary differential equations (ODEs). In this work, we consider BVPs with pathfollowing parameter $\lambda \in \mathbb{R}$ of the form[2]

$$\begin{aligned} \boldsymbol{f}(t,p,\boldsymbol{z}(t),\boldsymbol{z}'(t),\dots,\boldsymbol{z}^{(l)}(t);\lambda) = 0, \quad t \in (a,b], \\ \boldsymbol{g}(p,\boldsymbol{z}(a),\boldsymbol{z}'(a),\dots,\boldsymbol{z}^{(l-1)}(a),\boldsymbol{z}(b),\boldsymbol{z}'(b),\dots,\boldsymbol{z}^{(l-1)}(b)) = 0, \end{aligned} \tag{2}$$

where $\boldsymbol{z}$ is at least l-times continuously differentiable, $\boldsymbol{z} \in C^l[a,b]$, $p \in \mathbb{R}$ is a parameter fixed by the boundary conditions and $a, b \in \mathbb{R}$. Here, $\boldsymbol{z} : [a,b] \to \mathbb{R}^n$, $\boldsymbol{f} : (a,b] \times \underbrace{\mathbb{R}^n \times \ldots \times \mathbb{R}^n}_{(l+1)\text{ times}} \times \mathbb{R} \to \mathbb{R}^n$,

and $\boldsymbol{g} : \underbrace{\mathbb{R}^n \times \ldots \times \mathbb{R}^n}_{l\text{ times}} \times \underbrace{\mathbb{R}^n \times \ldots \times \mathbb{R}^n}_{l\text{ times}} \to \mathbb{R}^{ln}$, $l, n \in \mathbb{N}$.

The `bvpsuite` solver is based on the collocation method [3, 4]. For $m, N \in \mathbb{N}$ and a vector $\boldsymbol{\rho} := (\rho_1, ..., \rho_m)$, $\rho_i \in (0,1)$ we have $N+1$ mesh points $a = \tau_0 < ... < \tau_N = b$, and m collocation

[1]In the sequel we mostly refer to `bvpsuite 2.0` as `bvpsuite`. A predecessor is `bvpsuite 1.1` [8].

[2]We refer to the manual [2] for the general form of BVPs `bvpsuite` was built to solve. This includes the numerical solution of systems of mixed order with boundary conditions on the boundary or inside the interval, eigenvalue problems and index-1 differential algebraic equations systems. These can be posed on a finite or semi-infinite interval. An error estimate is computed and mesh adaptation is performed. Pathfollowing can be realized for all these problems now also.

points $t_{ij} := \tau_i + \rho_j(\tau_{i+1} - \tau_i)$ in between each two consecutive mesh points. Then, polynomial collocation converts the pathfollowing problem (2) into the system of equations

$$\begin{aligned}
\boldsymbol{f}(t_{ij}, p, \boldsymbol{P}(t_{ij}), \boldsymbol{P}'(t_{ij}), \ldots, \boldsymbol{P}^{(k)}(t_{ij}); \lambda) &= 0, \qquad i = 1, \ldots, N-1,\ j = 1, \ldots, m,\\
\boldsymbol{P}_i(\tau_i) &= \boldsymbol{P}_{i+1}(\tau_i), \qquad i = 1, \ldots, N-1\\
&\vdots\\
\boldsymbol{P}_i^{(l-1)}(\tau_i) &= \boldsymbol{P}_{i+1}^{(l-1)}(\tau_i), \qquad i = 1, \ldots, N-1\\
\boldsymbol{g}(p, \boldsymbol{P}_0(a), \ldots, \boldsymbol{P}_0^{(l-1)}(a), \boldsymbol{P}_{N-1}(b), \ldots, \boldsymbol{P}_{N-1}^{(l-1)}(b)) &= 0,
\end{aligned}$$

where $\boldsymbol{P}(t)$ is a smooth piecewise polynomial function and $\boldsymbol{P}(t) := \boldsymbol{P}_i(t)$ for $t \in [\tau_i, \tau_{i+1}]$. The collocation equations are reformulated as the underdetermined pathfollowing parameter-dependent generally nonlinear function

$$\tilde{\boldsymbol{F}}(\boldsymbol{x}; \lambda) = 0, \tag{3}$$

which is of the same form as (1), where $\boldsymbol{x} \in R^{nN(m+l)}$ represents the coefficients of the piecewise polynomial $\boldsymbol{P}(t)$ and λ is the pathfollowing parameter of the BVP (2).

The package `bvpsuite` provides a mesh adaptation strategy, based on asymptotically correct error estimation. This was proposed and investigated in [5]. In the context of solving BVPs, the goal of this mesh adaptation strategy is to determine the optimal mesh on which the solution satisfies user-specified accuracy requirements. The tolerances for the absolute and relative global errors of the approximate solution are prescribed by the user. Then, this optimal mesh is obtained from an iterative process. In the first phase, the mesh points are relocated to appropriately reflect the solution behavior. Next, mesh points are added to satisfy the tolerance prescribed by the user.

The implementations of the pathfollowing module described below take advantage of the strengths of `bvpsuite` and incorporate them in a continuation method to solve the pathfollowing problem (3) arising from (2). By doing so, the proposed continuation method for BVPs is based on polynomial collocation equipped with adaptive mesh selection, which controls the residual and the estimate of the global error of the approximate solution. In the next section we present the two implemented numerical continuation methods. Then, we test these numerical approaches for some examples.

2. Two numerical continuation methods for BVPs

2.1. Pathfollowing with Gauss-Newton

We implement the continuation via a predictor-corrector (PC) method as it is more compatible with our purpose to trace the solution path of a parameter-dependent BVP. This first implementation corresponds to the method described in [6], using Gauss-Newton iteration in the corrector step.

We assume that the function $\tilde{\boldsymbol{F}}$ in (3) is sufficiently smooth. We also assume that the Jacobian $D\tilde{\boldsymbol{F}}$ is of maximal rank, i.e., $\text{rank}(D\tilde{\boldsymbol{F}}(\boldsymbol{x}_0; \lambda_0)) = nN(m+l)$, when $\boldsymbol{F}(\boldsymbol{x}_0; \lambda_0) = \boldsymbol{0}$. A continuation method yields a solution for the nonlinear system, denoted as the set $S := \{(\boldsymbol{x}, \lambda) \colon \tilde{F}(\boldsymbol{x}, \lambda) = \boldsymbol{0}\}$. A solution $(\boldsymbol{x}_0, \lambda_0)$ of (3) in $D \subset R^n \times [a, b]$ is said to be regular if the Jacobian $D\tilde{\boldsymbol{F}}(\boldsymbol{x}; \lambda)$ has maximal rank, otherwise, a solution is said to be singular. For a regular point $(\boldsymbol{x}_0, \lambda_0)$, the Implicit Function Theorem implies that the solution near $(\boldsymbol{x}_0, \lambda_0)$ can be represented as a differentiable curve [1]. We consider only solution curves composed of regular points. Different types of PC methods are determined by the kind of predictor-step used in the algorithm. Classical and tangent continuation are the two widely-used PC methods [1, 7]. In this study, we consider the tangent continuation method.

A PC method starts with an initial estimate $(\hat{\boldsymbol{x}}_0, \lambda_0)$. Ideally, $(\hat{\boldsymbol{x}}_0, \lambda_0)$ is chosen such that the value of $F(\hat{\boldsymbol{x}}_0; \lambda_0)$ is not far from 0. The algorithm uses Gauss-Newton iteration to obtain $(\boldsymbol{x}_0, \lambda_0)$ such that $\tilde{\boldsymbol{F}}(\boldsymbol{x}_0; \lambda_0) = 0$. To obtain the next point $(\boldsymbol{x}_1, \lambda_1)$ in the solution set S of $\tilde{\boldsymbol{F}}(x; \lambda) = \boldsymbol{0}$, a predictor step is implemented. This step calculates an approximation

$$(\hat{\boldsymbol{x}}_1, \hat{\lambda}_1) := (\boldsymbol{x}_0, \lambda_0) + s\boldsymbol{t}_0,$$

where s is the step-length for the parameter and $\boldsymbol{t}_0$ is the tangent vector of the solution curve at $(\boldsymbol{x}_0, \lambda_0)$. The Gauss-Newton method is then applied to obtain $(\boldsymbol{x}_1, \lambda_1)$ using $(\hat{\boldsymbol{x}}_1, \hat{\lambda}_1)$ as the initial guess. This is the corrector step, and the process is repeated iteratively.

In a more general case, the partial derivative $\tilde{\boldsymbol{F}}_x(\boldsymbol{x},\lambda)$ may be singular, allowing for the occurrence of turning points [1, 7]. A turning point represents a relative extremum of λ [8] and as such, the solution curve cannot be locally expressed as a function of λ. Without modification, the tangent continuation method fails to cope with turning points. To address this, we proceed with a modified tangent continuation. Here, the tangent vector $\boldsymbol{t}_k$ in the k-th step is uniquely determined by

$$D\tilde{\boldsymbol{F}}(\boldsymbol{x}_k,\lambda_k)\boldsymbol{t}_k = \boldsymbol{0}, \quad \|\boldsymbol{t}_k\| = 1, \quad \langle \boldsymbol{t}_k, \boldsymbol{t}_{k-1}\rangle > 0,$$

where $\boldsymbol{t}_{k-1}$ is the tangent vector of the preceding solution point [7]. We require that successive tangents satisfy the condition $\langle \boldsymbol{t}_k, \boldsymbol{t}_{k-1}\rangle$ for $k = 0, 1, \ldots k_{\max}$, to ensure that we are not going backward on the solution curve. For a detailed discussion of this continuation method we refer to [1, 6].

To trace the solution curve of (3), we modify the above-mentioned tangent continuation PC method by adding an extra process in the predictor step. This is done by using the solution obtained in the predictor step as an initial guess. In this way, we are able to exploit the mesh adaptation strategy of `bvpsuite` to guarantee that the accuracy of the solution is within the user-specified tolerance. An algorithm for numerical continuation which shows the integration of the grid generation strategy is summarized in the algorithm found in the appendix. The default `bvpsuite` stopping criterion for the grid generation strategy is used. It indicates if there is appreciable reduction in the number of mesh points, i.e. if the number of mesh points on the new mesh is at least 90% of the previous mesh, then the mesh generation step finishes and the previous mesh is taken as the final mesh.

2.2. Pathfollowing in `bvpsuite 2.0`

An alternative method is implemented in `bvpsuite 2.0`. It follows the description of the pseudo-arclength continuation method in [6]. The main difference from the method described above, is in the corrector step. There, we add an equation to (3) to ensure the orthogonality of the iterates in the corrector step. In this way we obtain a system with the same number of equations and variables. This allows us to use the built-in Fast Frozen Newton method implemented in `bvpsuite`. Building on the implementation of the pathfollowing module in `bvpsuite 1.1` [8], an automatic steplength control was added, following the algorithm from [6].

Let $\tilde{\boldsymbol{F}}$ be given as in (3) and $\eta := nN(m+l)$. To specify our method, similarly to the above description, we assume to know a starting solution $\boldsymbol{a}_0 := (\boldsymbol{x}_0, \lambda_0) \in \mathbb{R}^{\eta+1}$ on an isolated homotopy path, i.e., solution path, of $\tilde{\boldsymbol{F}}$ in $\mathbb{R}^{\eta+1}$. In practice, the starting solution is obtained by approximating the solution of the BVP in question by `bvpsuite`, for the fixed starting value λ_0. Also, in general the solution path need not be isolated, as is discussed in the examples below.

A single pathfollowing step consists of the predictor step followed by the corrector iteration. In the predictor step, with the choice of a step-length s a predictor point $\widehat{\boldsymbol{a}}_0 := \boldsymbol{a}_0 + s\boldsymbol{t}(\boldsymbol{a}_0)$ is chosen, where $\boldsymbol{t}(\boldsymbol{a}_0)$ is the tangent in $\boldsymbol{a}_0$. To compute the tangent $\boldsymbol{t}(\boldsymbol{a}_0)$ of the solution path in the point $\boldsymbol{a}_0$, we augment $\tilde{\boldsymbol{F}}$ to $\tilde{\boldsymbol{F}}_P(\boldsymbol{x},\lambda)$ by the row $\lambda - \lambda_0$, in order to implicitly fix the value of λ in the equation $\tilde{\boldsymbol{F}}_P(\boldsymbol{x},\lambda) = \boldsymbol{0}$. Then, $\boldsymbol{t}(\boldsymbol{a}_0)$ satisfies the equation [8, 9]

$$D\tilde{\boldsymbol{F}}_P(\boldsymbol{a}_0)\,\boldsymbol{t}(\boldsymbol{a}_0) = (\boldsymbol{0}, 1)^T,$$

where $D\tilde{\boldsymbol{F}}_P$ is the Jacobian of $\tilde{\boldsymbol{F}}_0$.

In the first continuation step, `bvpsuite` uses the step-length prescribed by the user. Commencing in the subsequent continuation step, s is chosen according to the automatic step-length control strategy. The step-length matters for the convergence of the iterative method in the corrector step to a point on the isolated solution path. Since the step-length control strategy only starts performing and predicting step-lengths from the second continuation step onwards, we describe the corrector method first.

In the corrector step, we augment $\tilde{\boldsymbol{F}}$ to $\tilde{\boldsymbol{F}}_C$ by adding the term

$$(\boldsymbol{a} - \boldsymbol{a}_0) \cdot \boldsymbol{t}(\boldsymbol{a}_0) - s,$$

which when solving the equation $\tilde{\boldsymbol{F}}_C(\boldsymbol{a}) = \boldsymbol{0}$ implicitly ensures that the solution $\boldsymbol{a}$ of this equation lies on the line orthogonal to the tangent $\boldsymbol{t}_(a_0)$, passing through the predictor point $\widehat{\boldsymbol{a}}_0 := \boldsymbol{a}_0 + s \cdot \boldsymbol{t}(\boldsymbol{a}_0)$. This prescribed orthogonality is the key element of the pseudo-arclength continuation method, allowing for pathfollowing beyond turning points. For the system of equations $\tilde{\boldsymbol{F}}_C = \mathbf{0}$, the built-in solver methods in `bvpsuite` can be used. Starting from the predictor point $\widehat{\boldsymbol{a}}_0$, the iteration converges towards the point $\boldsymbol{a}_1$ further on the solution path, satisfying the user-prescribed tolerances. To allow for better approximations, the mesh adaptation of `bvpsuite` can be applied for pathfollowing problems as well. This offers a tool ensuring the approximations to satisfy the prescribed tolerances and thus move further along the path.

For the second pathfollowing step, starting from $\boldsymbol{a}_1$, the tangent $\boldsymbol{t}(\boldsymbol{a}_1)$ in this point is computed and the step-length control strategy helps determining a predictor point within the convergence domain of the Fast Frozen Newton method. This control strategy follows closely the strategy proposed in [6], adapting the strategy to the case of the simplified Newton method, on which the Fast Frozen Newton method is based, see [9] for details. Assuming the two Lipschitz-type bounds

$$\begin{aligned} \left\|\tilde{\boldsymbol{F}}'_C(\boldsymbol{a}_0)^{-1}(\tilde{\boldsymbol{F}}'_C(\boldsymbol{a}) - \tilde{\boldsymbol{F}}'_C(\boldsymbol{a}_0))\right\| &\le \omega_0 \|\boldsymbol{a} - \boldsymbol{a}_0\|, \\ \left\|\tilde{\boldsymbol{F}}'_C(\boldsymbol{a})^{-1}\big(\tilde{\boldsymbol{F}}'_C(u + s\delta_2 \boldsymbol{t}(u) - \tilde{\boldsymbol{F}}'_C(u)\big)\boldsymbol{t}(u)\right\| &\le \delta_2\, \omega_t \|\boldsymbol{t}(u)\|_2^2, \end{aligned}$$

where $0 \le \delta_2 \le 1$ and ω_0, ω_t are constants, and $u, u + s\delta_2\boldsymbol{t}(u)$ are elements of the domain of $\tilde{\boldsymbol{F}}_C$. The maximal steplength, allowing for convergence from the predictor point back to the path, depends on the a-priori unknown quantities ω_0 and ω_t. These are estimated from known values and from there, a prediction formula based on the step-length s_0 is deduced for the step-length s_1, namely

$$s_1 := \left(\frac{2}{c_{s_0}^2} \frac{\theta_{\max}}{\theta_0} \frac{\|\overline{\Delta \widehat{\boldsymbol{a}}_0}\|}{\|\widehat{\boldsymbol{a}}_0 - \boldsymbol{a}_1\|} \right)^{\frac{1}{2}} s_0,$$

where c_{s_0} is the scalar product between $\boldsymbol{t}(\boldsymbol{a}_0)$ and $\boldsymbol{t}(\boldsymbol{a}_1)$, $\overline{\Delta \widehat{\boldsymbol{a}}_0}$ is the first simplified Newton increment in the iteration from the predictor point $\widehat{\boldsymbol{a}}_0$ to $\boldsymbol{a}_1$, and θ_0 is the quotient of the norm of the second simplified Newton increment divided by the norm of the first. Finally, $\theta_{\max}$ is a user-specified threshold that must satisfy $\theta_{\max} \le 1/4$, and allows to control the convergence in the first step of the corrector iteration.

This formula is used to find a predictor point $\widehat{\boldsymbol{a}}_1$ on the tangent $\boldsymbol{t}(\boldsymbol{a}_1)$. This prediction formula relies on the maximal step-length, which allows convergence from the predictor point back to the solution path, in the ideal case. Therefore, for practical reasons some safety measures were implemented, verified during and at the end of the computation of each continuation step, to reduce the step-length if necessary. Details of the step-length control strategy and its implementation can be found in [9].

In pathfollowing problems, often the evolution of a characteristic value of the solution under variation of the pathfollowing parameter λ is of interest. For the tracking of such a development and the other features which can be performed with `bvpsuite`, we refer to the manual [2].

3. Numerical examples

3.1. Comparison of the two methods

We chose three examples to demonstrate the properties of the two numerical continuation methods. These examples are ordered in an ascending manner according to their complexity.

Example 3.1. *Bratu's Problem* is an elliptic partial differential equation recurring in applications of science and engineering [10]. The one-dimensional Bratu problem is given by

$$\begin{cases} z''(t) + \lambda\, e^{z(t)} = 0, & \text{for } t \in (0,1)\,, \\ z(0) = 0, \quad z(1) = 0. \end{cases}$$

Pathfollowing with Gauss-Newton was initialized on a mesh composed of 31 equidistant mesh points with two Gaussian collocation points in each subinterval. Relative and absolute tolerances of 10^{-6} for the Newton method were prescribed to compute the initial solution profile on the first iteration. Similarly, relative and absolute tolerances of 10^{-6} were set for the mixed tolerance condition of the mesh adaptation strategy. The maximal absolute error is of the order of magnitude of 10^{-4}. The CPU time (in seconds) with grid generation is significantly larger due to repeated calls of the

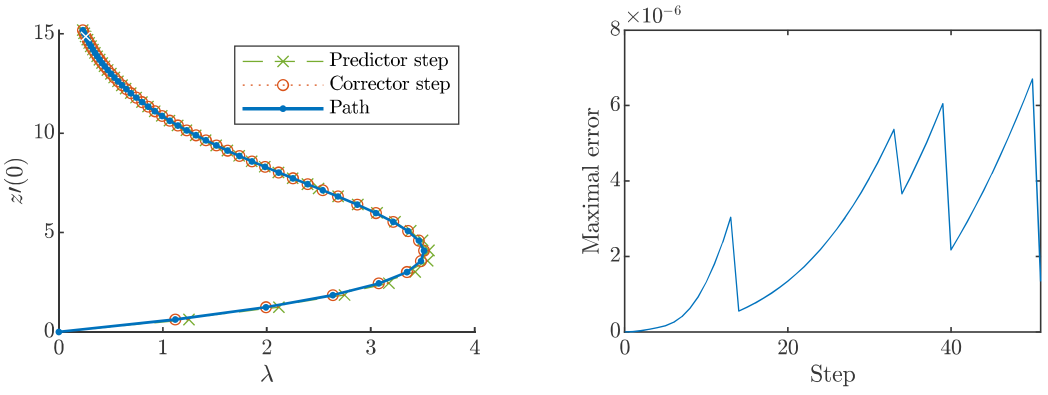


Figure 1: Example 3.1 with Gauss-Newton: Numerical continuation results for Example 3.1. Plot of $z'(0)$ vs. λ (left), and maximal error in each step (right).

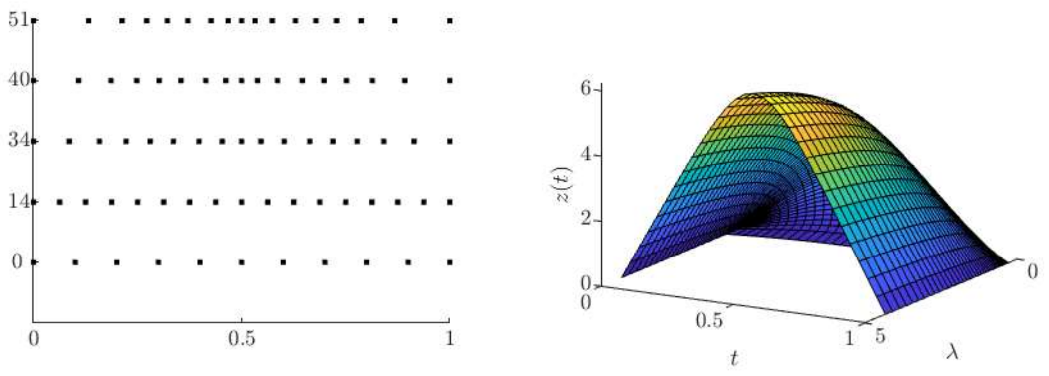


Figure 2: Example 3.1 with Gauss-Newton: Evolution of the mesh for Example 3.1 over 50 steps (left), and evolution of the solution (right).

Gauss-Newton solver module and the refinement of the solution mesh. Figure 1 shows the path of the dependent parameter $z'(0)$ versus the free parameter λ obtained via the modified tangent continuation method. For this example, we illustrate how the solution mesh is updated for each step of the continuation. Figure 2 shows the evolution of the mesh density over 50 continuation steps.

For the pathfollowing module implemented in `bvpsuite`, the absolute and relative tolerances of the Newton iteration are set to 10^{-6}, the absolute and relative tolerances of the mesh adaptation to 10^{-4}. The computations are started with a mesh of 51 equidistant points on the interval $[0, 1]$ and 3 Gaussian collocation points in each subinterval. The evolution of $z'(0)$ under variation of λ, starting at $\lambda = 0$ in positive direction, is followed until $z'(0) = 50$. In order to illustrate how $\theta_{\max}$ influences the pathfollowing behavior, this example is computed for the values $\theta_{\max} = 10^{-1}$ and $\theta_{\max} = 10^{-3}$. The resulting paths and the maximal error at the mesh points in each step are displayed in Figure 3 and Figure 4, respectively. In both cases, mesh adaptation was activated but not needed.

When $\theta_{\max}$ is chosen to be small, the evaluation points along the path are concentrated. Along the run in both cases, an adaptation of the step-length around the turning point and beyond is visible. In the case $\theta_{\max} = 10^{-1}$, the computation reaches $z'(0) = 50$ after 46 continuation steps and in the case of $\theta_{\max} = 10^{-3}$ after 420 continuation steps. But when comparing the evolution of the maximal error, the choice of $\theta_{\max}$ does not seem to have much influence.

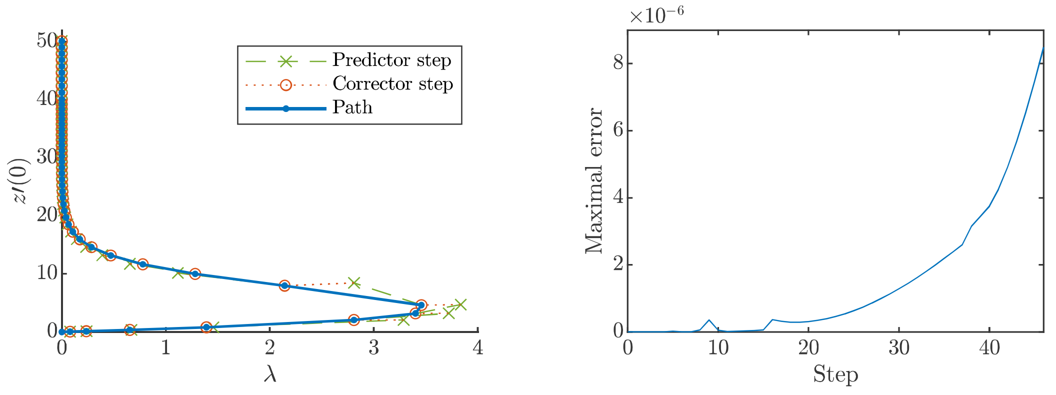


Figure 3: Example 3.1 in `bvpsuite`: Evolution of $z'(0)$ under variation of λ with $\theta_{\max} = 10^{-1}$ until $z'(0) = 50$ (left), and estimation of the maximal error over the mesh in each step (right).

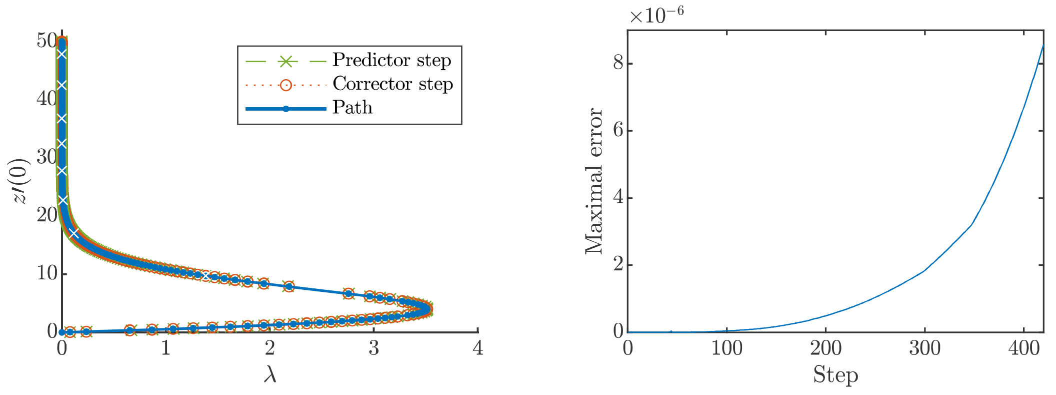


Figure 4: Example 3.1 in `bvpsuite`: Evolution of $z'(0)$ under variation of λ with $\theta_{\max} = 10^{-3}$ until $z'(0) = 50$, where the white crosses mark 50 continuation steps (left), and estimation of the maximal error on the mesh in each step (right).

Example 3.2. We consider the forced nonlinear oscillator ODE [11]

$$\begin{cases} \left(\dfrac{\lambda}{2\pi}\right)^2 z''(t) + \dfrac{1}{25}\left(\dfrac{\lambda}{2\pi}\right) z'(t) - \dfrac{1}{5}z(t) + \dfrac{8}{15}z(t)^3 = \dfrac{2}{5}\cos 2\pi t, \quad \text{for } t \in [0,1]\,, \\ z(0) = z(1), \quad z'(0) = z'(1). \end{cases}$$

This is a recommended benchmarking problem for testing numerical continuation methods due to the complex dependence of the harmonic solution on the frequency parameter λ [12].

The numerical simulation was initialized on a mesh composed of 21 equidistant mesh points with two Gaussian collocation points in each subinterval. Relative and absolute tolerances of 10^{-3} for the Newton method were set to compute the initial solution profile in the first iteration. Similarly, relative and absolute tolerances of 10^{-3} were set for the mixed tolerance condition of the adaptive mesh strategy. The path of $z(0)$ is traced with respect to the inverse of the exciting frequency $\frac{1}{\lambda}$ to enlarge the loops of the main solution branch resulting from the turning points encountered along the path as shown in Figure 5. Also the error estimation of our numerical continuation method is shown.

For the pathfollowing module in `bvpsuite`, the absolute and relative tolerances of the Fast Frozen Newton iteration are set to 10^{-6}, the absolute and relative tolerances of the mesh adaptation to 10^{-4}. The computations are started with a mesh of 51 equidistant points on the interval $[0, 1]$ and 2 Gaussian collocation points in each subinterval. The characteristic value of the solution which is followed along the path is $z(0)$. Here, $\theta_{\max}$ was set to $5 \cdot 10^{-2}$, and the computation was started at the value $\lambda = 3$ in negative direction. Following the evolution of the path, the solution of the problem would exhibit a more and more oscillatory behavior. This triggers mesh adaptation, increasing the number of mesh points to 694. Altogether, 413 pathfollowing steps were executed.

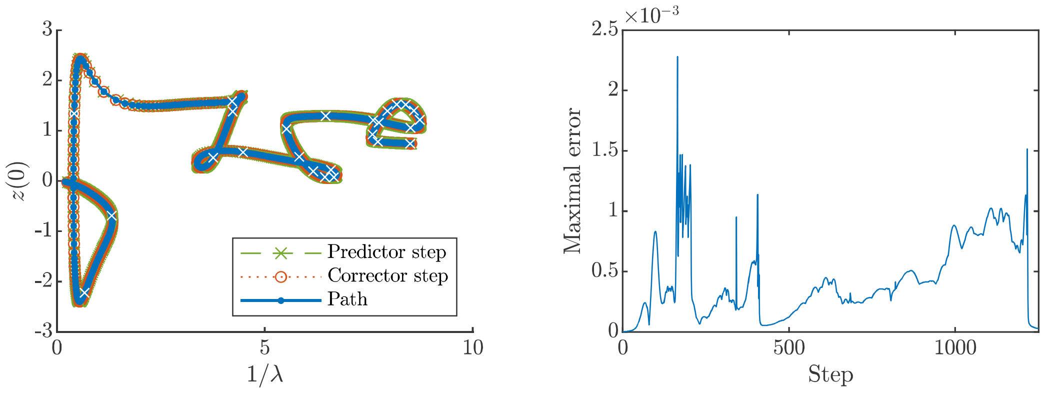


Figure 5: Example 3.2 with Gauss-Newton: Evolution of $z(0)$ under variation of λ, with the white crosses marking 50 continuation steps (left), and estimation of the maximal error (right).

The results are displayed in Figure 6. The evolution of the mesh is displayed in Figure 7. See also Figure 8 for the evolution of the solution along the path.

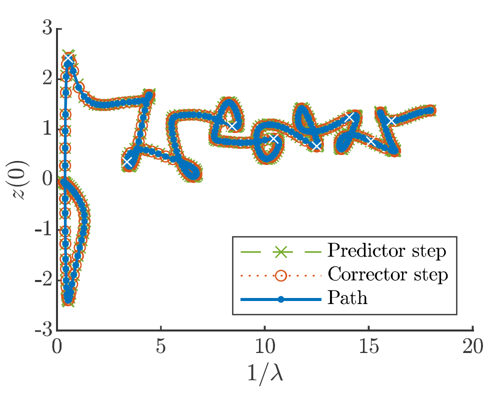
Due to the large number of bifurcation points, this example is an interesting challenge for any pathfollowing code. The pathfollowing module in `bvpsuite` managed to get as far as it did due to the various verifications which are performed in each step along the path, controlling different key values of the computation, see [9] for details. This allows an automated computation of the path for this example.

Example 3.3. The following singular parameter-dependent third-order BVP describes the flow in a tube with accelerating surface velocity (referred to as "axisymmetric" flow) [13, 14]

$$\begin{cases} t^2 z^{(3)}(t) - tz''(t) + z'(t) - t^3 p - \lambda\left(tz'(t)^2 - tz(t)z''(t) + z(t)z'(t)\right) = 0, & \text{for } t \in (0,1)\,, \\ z(0) = z'(0) = 0, \quad z(1) = 0, \quad \text{and} \quad z'(1) = 1. \end{cases}$$

The initial solution profile provided in [13] was used for our collocation method at the beginning of the continuation algorithm. The numerical simulation was initialized on a mesh composed of 31 equidistant mesh points with two Gaussian collocation points in each subinterval. Relative and absolute tolerances of 10^{-4} for the Newton method were prescribed to compute the initial solution profile on the first iteration. Similarly, relative and absolute tolerances of 10^{-4} were set for the mixed tolerance condition of the adaptive mesh strategy. The evolution of the pathfollowing parameter and the solution are shown in Figure 9.

For the pathfollowing module implemented in `bvpsuite`, the constant function equal to 1 is chosen as initial profile for the approximation of the solution in the case $\lambda = 0$. The continuation is started in positive direction. The evolution of the parameter p is observed. The evolution is followed until the value $p = -60$ is reached. In the solver settings, a starting mesh with 101 equidistant mesh points, and 3 Gaussian collocation points in each subinterval. The relative and absolute tolerances for the nonlinear solver were set to 10^{-6} and the relative and absolute tolerances for the mesh adaptation to 10^{-4}. $\theta_{\max}$ was set to 10^{-2}. The run finishes after 15 tangent continuation steps, without any mesh adaptation. The result is displayed in Figure 10. In Figure 11, the approximation to the solution of the problem with λ being equal to 0, 5 and 10 are shown.



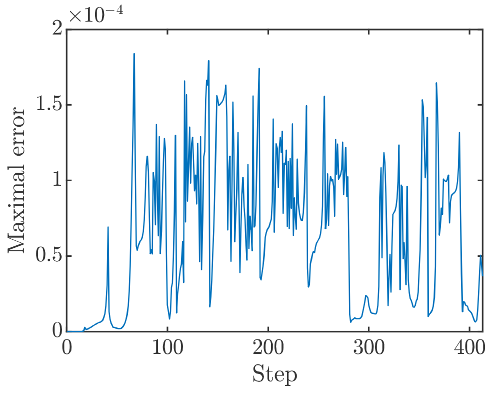


Figure 6: Example 3.2 in `bvpsuite`: Evolution of $z(0)$ under variation of λ, with the white crosses marking 50 continuation steps (left), and estimation of the maximal error (right).

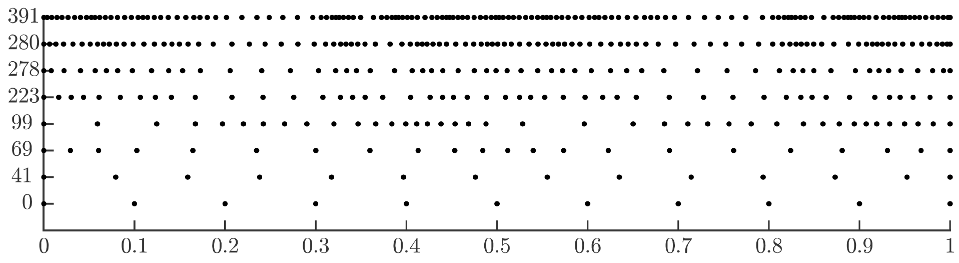


Figure 7: Example 3.2 in `bvpsuite`: Evolution of the mesh adjusted by the mesh adaptation algorithm in `bvpsuite` along the path shown in Figure 6, where on the vertical axis the step number in which the mesh was adapted is shown.

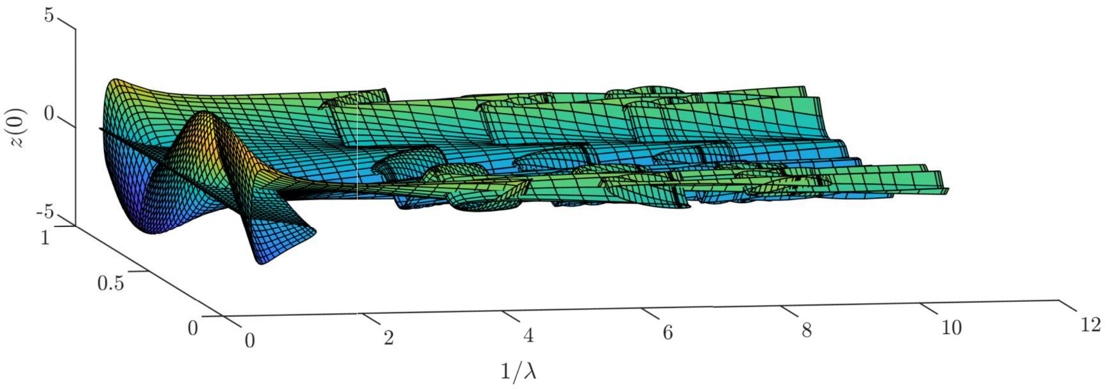
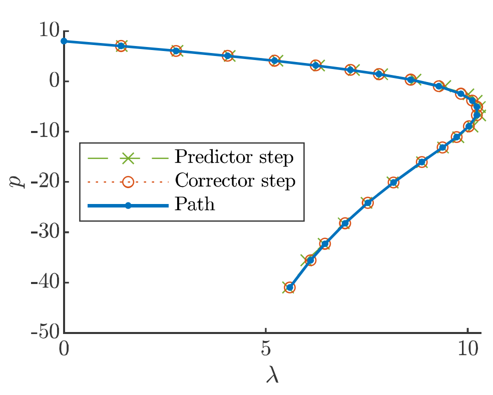


Figure 8: Example 3.2 in `bvpsuite`: Evolution of the approximation of the solution of the BVP on the interval $[0, 1]$.



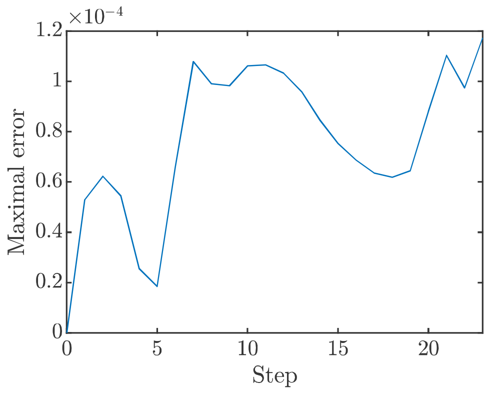
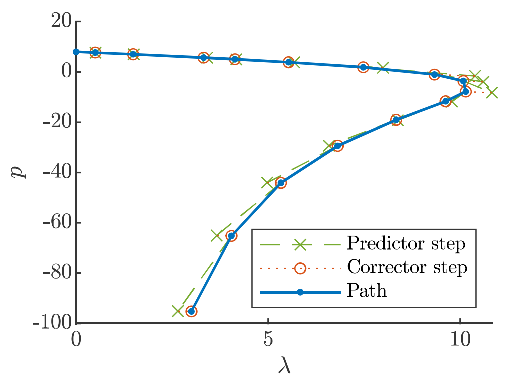


Figure 9: Example 3.3 with Gauss-Newton: Evolution of p under variation of λ (left), and the estimation of the maximal error (right).



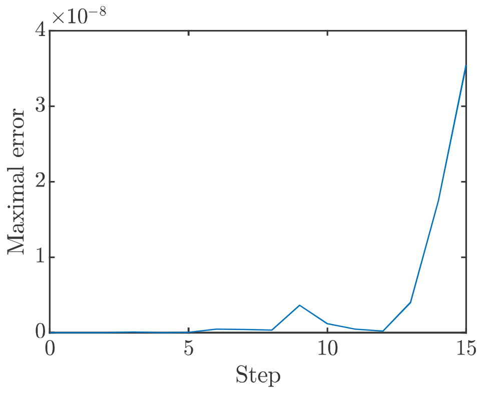


Figure 10: Example 3.3 in `bvpsuite`: Evolution of p until $p = -60$ (left), and estimation of the maximal error (right).

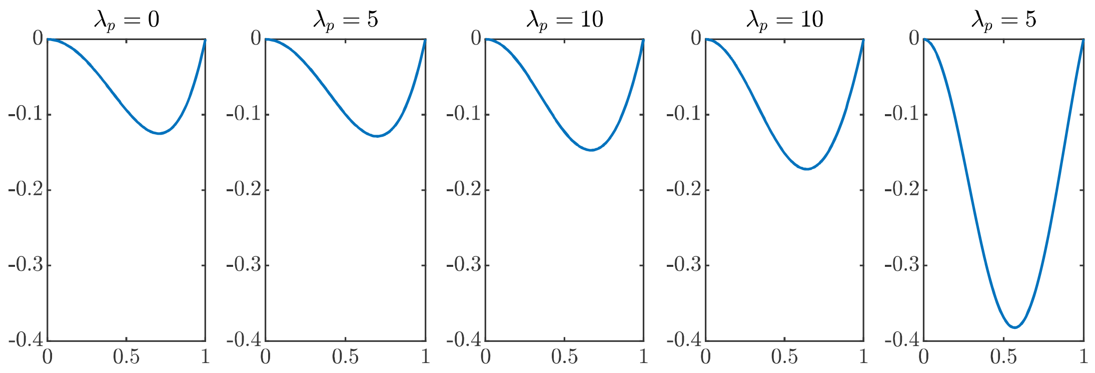


Figure 11: Example 3.3 in `bvpsuite`: Approximation to the solution of the BVP for $\lambda = 0, 5, 10$ along the path from Figure 10.

3.2. Metal ball under hydrostatic pressure

This example shall demonstrate the difficulty of computing paths where bifurcation occurs and the paths are not isolated, in contrast to Example 3.2. Consider the BVP

$$\begin{aligned}
&\delta \Bigg(z_1''(t) + z_1'(t)\cot(t) + \cot(t)^2 \frac{\cos(t-z_1(t))}{\cos(t)} \frac{\sin(t-z_1(t)) - \sin(t)}{\cos(t)} \\
&\qquad -0.3 \frac{\cos(t-z_1(t)) - \cos(t)}{\sin(t)} \Bigg) \\
&\quad = -z_2(t) \frac{\sin(t-z_1(t))}{\sin(t)} - 4\lambda \frac{\cos(t-z_1(t))}{\sin(t)} z_3(t), \qquad \text{(4a)} \\
&\delta \left(z_2''(t) + z_2'(t)\cot(t) - z_2(t) \left(\cot(t)^2 \frac{\cos(t-z_1(t))^2}{\cos(t)^2} - 0.3(1 - z_1'(t)) \frac{\sin(t-z_1(t))}{\sin(t)} \right) \right) \\
&\quad = \frac{\cos(t-z_1(t)) - \cos(t)}{\sin(t)} + \delta \Bigg(- 4\lambda \cot(t) z_3(t) \\
&\qquad \cdot \left(\frac{\sin(2(t-z_1(t)))}{\sin(2t)} + 0.3(1 - z_1'(t)) \frac{\cos(t-z_1(t))}{\cos(t)} \right) \\
&\qquad + 4\lambda \frac{(\sin(t)^2 (1 - z_1'(t)) \cos(t-z_1(t)) + 2\sin(t)\cos(t)\sin(t-z_1(t)))}{\sin(t)} \Bigg), \qquad \text{(4b)} \\
&z_3'(t) = \cos(t-z_1(t))\sin(t), \qquad \text{for } t \in [0,\pi] \qquad \text{(4c)} \\
&z_1(0) = z_1(\pi) = 0, \quad z_2(0) = z_2(\pi) = 0, \quad z_3(0) = 0, \qquad \text{(4d)}
\end{aligned}$$

where $\delta = 0.00369$.

This BVP describes the deformation of a metal ball under hydrostatic pressure. We consider now the plot of the evolution of the pressure λ versus the measure for the deformation $\|z_1\|_\infty$. For pressures larger than the critical pressure $\lambda = 1$, stable branches can be observed, where the deformations of the metal ball due to the increased pressure are irreversible. For pressures smaller than the critical pressure $\lambda = 1$, the branch where no deformation occurs, can be observed as well as branches where deformations occur. Under small perturbations, these disappear and the ball jumps back into its original undeformed state.

The problem is a singular BVP and therefore it is more challenging than the previous ones. In order to reproduce the computations performed in [8], the problem (4) is rewritten as an eigenvalue problem to find an initial profile to start the pathfollowing close to the critical pressure $\lambda = 1$. The absolute and relative tolerances of the Newton iteration are set to 10^{-6}, the absolute and relative tolerances of the mesh adaptation to 10^{-4}. The computations are started on a mesh of 101 equidistant points on the interval $[0,\pi]$, and 3 Gaussian collocation points in each subinterval. The characteristic value of the solution which is followed along the path is $\|z_1\|_\infty$. Also, $\theta_{\max}$ is set to $5 \cdot 10^{-2}$ and the starting value for λ to 0.98. The path is followed in decreasing direction of λ. With mesh adaptation augmenting the mesh to 272 mesh points along the run, the computation is stopped above $\|z_1\|_\infty = 4$ after 95 continuation steps. The results are displayed in Figure 12.

A second computation with the same tolerance settings and $\theta_{\max}$ as above, is started at $\lambda = 0$ with the constant function 0 as initial profile. The evolution of $\|z_1\|_\infty$ is followed with increasing λ. A mesh with 51 equidistant points and 3 Gaussian collocation points in each subinterval is chosen. With some mesh adaptation moving the 51 mesh points on the interval $[0,\pi]$ such that the tolerances are satisfied along the run, the computation is stopped above $\lambda = 0$, after 119 continuation steps. The results are displayed in Figure 13.

The two paths that were recovered represent the path of the single dimple solution and the double dimple solution, respectively. The shape of one half of the metal ball in each step along those two paths is computed by solving the BVP

$$x'(t) = \cos(t - z_1(t)), \quad y'(t) = \sin(t - z_1(t)), \qquad \text{for } t \in [0,\pi], \qquad \text{(5a)}$$

$$x(0) = 0, \quad y\left(\frac{\pi}{2}\right) = 0, \qquad \text{(5b)}$$

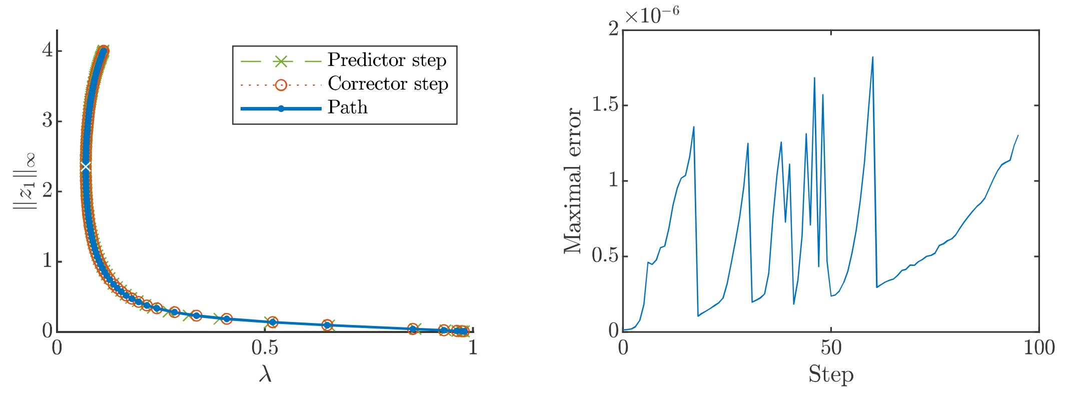


Figure 12: Shell buckling problem (4) in `bvpsuite`: Evolution of $\|z_1\|_\infty$ under variation of λ with $\theta_{\max} = 5 \cdot 10^{-2}$, until $\|z_1\|_\infty = 4$. This is the path of the single dimple solution (left). Estimation of the maximal error (right).

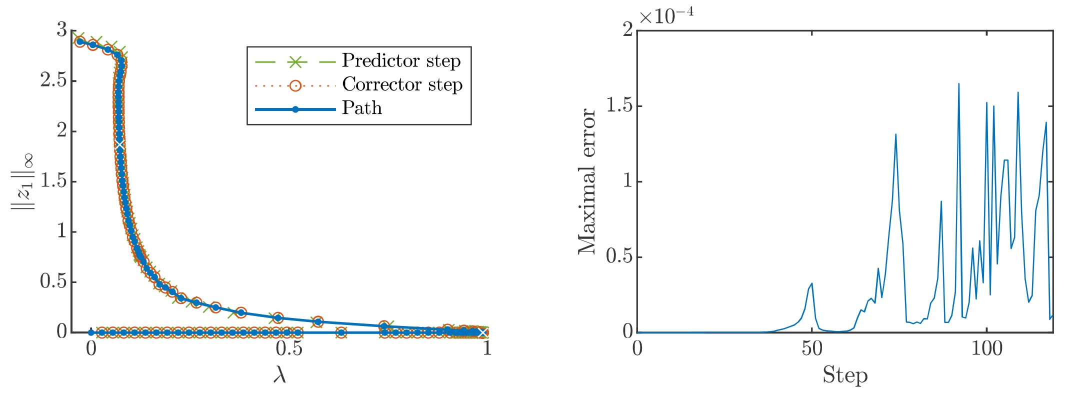


Figure 13: Shell buckling problem (4) in `bvpsuite`: Evolution of $\|z_1\|_\infty$ under variation of λ with $\theta_{\max} = 5 \cdot 10^{-2}$, until $\lambda = 0$. This is the path of the double dimple solution (left). Estimation of the maximal error (right).

with the function z_1 computed with `bvpsuite` in each step of the paths in Figures 12 and 13. The solutions x and y are then the coordinates of the right half of the ball, as displayed in Figure 15 in the case of the single dimple solution and in Figure 14 in the case of the double dimple solution.

The verification procedure for each step to be accepted, which is implemented in the pathfollowing module in `bvpsuite`, allowed the pathfollowing in this example, where near to the critical pressure $\lambda = 1$ many paths intersect and are close to each other. The pathfollowing module in `bvpsuite` was not designed to recognize bifurcation points, but in some examples the features implemented in the code allow the pathfollowing beyond bifurcation points.

4. Conclusion and outlook

The two presented pathfollowing methods differ in the corrector step, where in the first one the Gauss-Newton method was used, in the second one the algorithm relies on the Fast Frozen Newton method, which is the main approximation method in `bvpsuite`. Relying on the already implemented and well-tested algorithms of the `bvpsuite` package over the years, was the deciding factor for the method using the Fast Frozen Newton method to be released with the package `bvpsuite`. The pathfollowing module is compatible with the other problem types, for which `bvpsuite` is able to compute approximations to their solutions. These include linear and nonlinear BVPs, eigenvalue problems and index-1 differential algebraic systems of equations, either on finite or semi-infinite intervals. Furthermore, the pathfollowing module can be used with the error estimate and mesh adaptation implemented in `bvpsuite`, as well as the solver methods implemented for nonlinear problems. Lastly, many safety features and parameters adjustable by the user before the start of, as well as during, the pathfollowing computations, can enable the computation of more complicated examples. These have already proven themselves to be helpful in the computation of problems such as those shown in Section 3.1. We refer to the manual [2] for the instructions on how to use `bvpsuite` and the full description of its features.

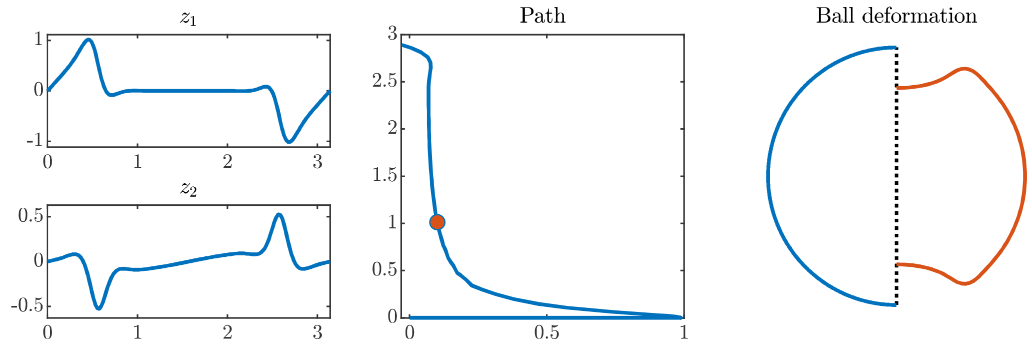


Figure 14: Shell buckling problem (4) in `bvpsuite`: A deformation state of the half of the ball along the path of the double dimple solution.

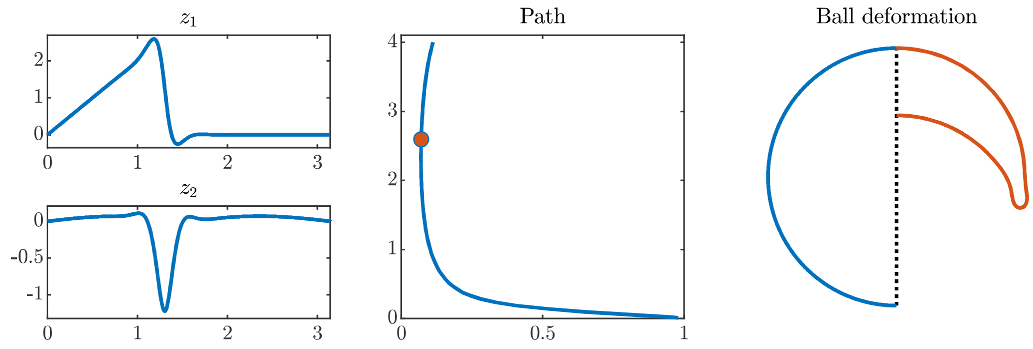


Figure 15: Shell buckling problem (4) in `bvpsuite`: A deformation state of the half of the ball along the path of the single dimple solution.

Acknowledgments

This project was supported by the Continuous Operational and Outcomes-based Partnership for Excellence in Research and Academic Training Enhancement (COOPERATE) financial support program facilitated by the University of the Philippines Office of International Linkages.

The authors would like to thank Alois Steindl for his valuable help in finding an initial profile for the shell buckling problem and identifying the correct paths.

References

[1] P. Deuflhard, A. Hohmann, Numerical Analysis in Modern Scientific Computing: An Introduction, Springer, 2003.

[2] M. Fallahpour, O. Koch, A. Sass, E. Weinmüller, Manual for `bvpsuite` – a MATLAB solver for singular BVPs in ODEs, EVPs and DAEs, can be retrieved as part of the `bvpsuite` package at https://www.asc.tuwien.ac.at/weinmueller/ (2019).

[3] C. De Boor, B. Swartz, Collocation at Gaussian Points, SIAM J. Numer. Anal. 10 (1973) 582–606.

[4] U. Ascher, R. Mattheij, R. Russell, Numerical Solution of Boundary Value Problems for Ordinary Differential Equations, SIAM, 1995.

[5] G. Pulverer, G. Söderlind, E. Weinmüller, Automatic grid control in adaptive BVP solvers, Numer. Algorithms 56 (2011) 61–92.

[6] P. Deuflhard, Newton Methods for Nonlinear Problems, Springer, 2004.

[7] E. Allgower, K. Georg, Introduction to Numerical Continuation Methods, SIAM, 2003. f

[8] G. Kitzhofer, O. Koch, E. Weinmüller, Pathfollowing for essentially singular boundary value problems with application to the complex Ginzburg-Landau equation, BIT 49 (2009) 141–160.

[9] W. Auzinger, M. Fallahpour, O. Koch, E. Weinmüller, Implementation of a pathfollowing strategy with an automatic step-length control: New MATLAB package bvpsuite 2.0, Vienna University of Technology, Technical Report ASC 31/2019.

[10] L. Bougoffa, Exact solutions of a generalized Bratu equation, Romanian Journal of Physics 62 (110) (2017) 1–5.

[11] G. Bader, P. Kunkel, Continuation and collocation for parameter-dependent boundary value problems, SIAM Journal on Scientific and Statistical Computing 10 (1) (1989) 72–88.

[12] R. Seydel, A continuation algorithm with step control, International Series of Numerical Mathematics 70 (1984) 480–494.

[13] H. Binous, B. Higgins, A simple method for tracking turning points in parameter space, Journal of Chemical Engineering of Japan 43 (2010) 1035–1042.

[14] J. Brady, A. Acrivos, Steady flow in a channel or tube with an accelerating surface velocity. an exact solution to the navier-stokes equations with reverse flow, Journal of Fluid Mechanics 112 (1981) 127–150.