Implementation of an Arc-Length Method in PETSc

Zachary R. Atkins*
May 8, 2024

Abstract

Traditional Newton methods with load and displacement control are unable to solve problems with limit points, bifurcations, and snap-through instabilities. Arc-length continuation methods are a powerful tool for solving these problems, but are not natively supported in PETSc (Portable, Extensible Toolkit for Scientific Computation). This work implements two arc-length continuation methods in PETSc: Crisfield's method with partial corrections and the normal-plane constraint method. Due to the composable nature of PETSc solvers, these methods can be used as a replacement for pseudo-time stepping approaches to static problems or as a composed solver for quasistatic and dynamic problems with time integration. To support the latter, this work implements a novel method for bounding the load parameter with a hybrid arc-length and Newton method in the final increment. The implementations are verified using a 3D large deformation buckling test, where they outperform the standard Newton method with pseudo-time stepping in terms of convergence and accuracy.

1 Introduction

PETSc (Portable, Extensible Toolkit for Scientific Computation) is a widely-used software library for the parallel solution of partial differential equations [2]. It provides a variety of composable linear and nonlinear solvers, as well as tools for mesh management and parallel I/O. One of the most powerful features of PETSc is its ability to solve problems on unstructured meshes, which are often used in complex geometries, as well as its built-in scalable framework, allowing the same program to be run on laptops and cutting-edge supercomputers. However, the current version of PETSc does not provide a built-in arc-length continuation method, limiting its utility for many solid mechanics applications. These include problems with bifurcations [6, 4, 10], fracture and snap-through instabilities [17, 5], buckling [11, 12], strain-softening [4], and other phenomena that are difficult to capture with standard Newton-style solvers. Beyond solid mechanics, arc-length continuation has had success in other numerical simulations, such as electrodynamics [18], shock-wave propagation [15], and thermomechanics [1].

This work implements two approaches to arc-length continuation in PETSc as nonlinear solvers (SNES): Crisfield's method [6] and the normal-plane constraint method [10, 11]. Both methods are suitable to large scale problems, as they preserve the symmetry and structure of the tangent stiffness matrix, allowing for efficient solution of the linear systems. They differ in complexity, accuracy, and computational cost, but each have their own advantages and disadvantages. A comprehensive review of arc-length continuation methods and extensions thereof is provided in [12]. This work implements some of the extensions proposed in the literature, such as partial corrections to the load parameter [19, 12] and the ability to choose between a spherical and cylindrical constraint surface [12].

To verify the implementation, we extend upon the large deformation nonlinear buckling test in

PETSc SNES Tutorials `ex16`[1] to support arc length continuation of both point and body loading of an arch which undergoes a snap-through instability. Additionally, we implement a 3D version of the Lee frame snap-through test [7] to verify the ability of the arc length continuation methods to capture snap-through instabilities and reversed loading paths.

2 Arc Length Continuation Method

Consider the general nonlinear system of equations in residual form for $\boldsymbol{x} \in \mathbb{R}^n$ and $R\colon \mathbb{R}^n \to \mathbb{R}^n$,

$$\boldsymbol{R}(\boldsymbol{x}) = \boldsymbol{0}. \tag{1}$$

The arc-length continuation method solves eq. (1) by parameterizing part of the residual function $\boldsymbol{R}$ with the load parameter λ, yielding the equilibrium equations and the constraint equation as

$$\boldsymbol{R}(\boldsymbol{x}, \lambda) = \boldsymbol{F}^{\text{int}}(\boldsymbol{x}) - \boldsymbol{F}^{\text{ext}}(\boldsymbol{x}, \lambda) = \boldsymbol{0}, \tag{2}$$

$$f(\Delta\boldsymbol{x}, \Delta\lambda) = \|\Delta\boldsymbol{x}\|^2 + \psi^2\Delta\lambda^2 - L^2 = 0, \tag{3}$$

where L is the arc length step size, $\psi^2 \geq 0$ is a consistency parameter controlling the shape of the constraint surface, $\Delta\boldsymbol{x}$ and $\Delta\lambda$ are the incremental updates from the last equilibrium state, and $\boldsymbol{F}^{\text{int}}$ and $\boldsymbol{F}^{\text{ext}}$ are the internal and external forces, respectively. Often, $\boldsymbol{F}^{\text{ext}}(\boldsymbol{x}, \lambda)$ is linear in λ, which can be thought of as applying the external force in proportional load increments. One obvious use-case for this is in solving a nonlinear system with a constant right-hand-side vector, where the load parameter λ is used to scale the right-hand-side vector. Generally, however, $\boldsymbol{F}^{\text{ext}}(\boldsymbol{x}, \lambda)$ may depend non-linearly on λ or $\boldsymbol{x}$, or both.

Since the constraint equation eq. (3) has a step size L, it may be necessary to take multiple increments in order to reach the desired load parameter λ. Thus, these equations are solved via an incremental-iterative method by updating the solution $\boldsymbol{x}$ and the load parameter λ. The method assumes that at the beginning of each increment n, the solution exactly satisfies the equilibrium equations eq. (2). Then, the updated solution $\boldsymbol{x}_n$ and load parameter λ_n are found via a series of candidate solutions $\boldsymbol{x}_n^{(k)}$ and $\lambda_n^{(k)}$ iteratively updated through a Newton-Raphson method. We denote by $\Delta\boldsymbol{x}_n$ and $\Delta\lambda_n$ the accumulated updates over the current load increment n and by $\delta\boldsymbol{x}^{(k)}$ and $\delta\lambda^{(k)}$ the incremental updates at each iteration k.

The most straightforward approach to applying a Newton-Raphson method is to consider the augmented $(n+1)$-dimensional space of $\boldsymbol{x}$ and λ and apply a Newton-Raphson method to solve the linearized system of equations,

$$\begin{bmatrix} \boldsymbol{K}_n^{(k-1)} & -\boldsymbol{Q}_n^{(k-1)} \\ (f_{\Delta\boldsymbol{x}})_n^{(k-1)} & (f_{\Delta\lambda})_n^{(k-1)} \end{bmatrix} \begin{bmatrix} \delta\boldsymbol{x}_n^{(k)} \\ \delta\lambda_n^{(k)} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{R}(\boldsymbol{x}_n^{(k-1)}) \\ f(\Delta\boldsymbol{x}_n^{(k-1)}, \Delta\lambda_n^{(k-1)}) \end{bmatrix}, \tag{4}$$

where $\boldsymbol{K}_n^{(k-1)}$ is the tangent stiffness matrix, $\boldsymbol{Q}_n^{(k-1)}$ is the tangent load vector, and $(f_{\Delta\boldsymbol{x}})_n^{(k-1)}$ and $(f_{\Delta\lambda})_n^{(k-1)}$ are the derivatives of the constraint equation with respect to $\Delta\boldsymbol{x}$ and $\Delta\lambda$, respectively [8]. The tangent stiffness matrix K and tangent load vector $\boldsymbol{Q}$ are defined as

$$\begin{aligned} \boldsymbol{K}_n^{(k-1)} &= \boldsymbol{K}(\boldsymbol{x}_n^{(k-1)}, \lambda_n^{(k-1)}) = \partial\boldsymbol{F}(\boldsymbol{x}_n^{(k-1)}, \lambda_n^{(k-1)})/\partial\boldsymbol{x} \\ \boldsymbol{Q}_n^{(k-1)} &= \boldsymbol{Q}(\boldsymbol{x}_n^{(k-1)}, \lambda_n^{(k-1)}) = -\partial\boldsymbol{F}(\boldsymbol{x}_n^{(k-1)}, \lambda_n^{(k-1)})/\partial\lambda = \partial\boldsymbol{F}^{\text{ext}}(\boldsymbol{x}_n^{(k-1)}, \lambda_n^{(k-1)})/\partial\lambda. \end{aligned}$$

[1] See `https://petsc.org/release/src/snes/tutorials/ex16.c.html`

In the following, we will drop the $(\cdot)_n^{(k-1)}$ unless necessary for clarity. In the simplest case, $\boldsymbol{Q}$ is a constant vector in λ, which corresponds to proportional loading. Our implementation allows for the user to provide a general function which computes $\boldsymbol{Q}(\boldsymbol{x}, \lambda)$ as well as automatically adds the RHS vector, if provided.

While the monolithic formulation results in a nonsingular system, it also results in a non-symmetric and non-banded stiffness matrix, which can be computationally expensive to solve [8]. However, these issues can be mitigated via a splitting of the solution update into two components first proposed in [3] for a generalized displacement control method. Specifically, we split the solution update into two components

$$\delta \boldsymbol{x} = \delta s \delta \boldsymbol{x}^R + \delta\lambda \delta \boldsymbol{x}^Q, \tag{5}$$

where $\delta s = 1$ unless partial corrections are used (discussed more below). Each of $\delta \boldsymbol{x}^R$ and $\delta \boldsymbol{x}^Q$ are found via solving a linear system with the tangent stiffness matrix $\boldsymbol{K}$,

$$\begin{aligned} \boldsymbol{K}\delta \boldsymbol{x}^R &= -\boldsymbol{R}(\boldsymbol{x}, \lambda) \\ \boldsymbol{K}\delta \boldsymbol{x}^Q &= \boldsymbol{Q}(\boldsymbol{x}, \lambda), \end{aligned}$$

where $\boldsymbol{Q}$ is the tangent load vector as defined above.

During the first iteration of each increment, the *predictor* stage, the update to the load parameter $\delta\lambda$ is chosen such that the constraint equation is exactly satisfied. In subsequent iterations, the *corrector* stage, there are multiple approaches to choosing $\delta\lambda$, discussed in detail below.

2.1 Quadratic Constraint Surface and Linearized Equilibrium Line

Considering the augmented $n+1$ dimensional space of $\boldsymbol{x}$ and λ, the constraint equation eq. (3) can be reframed as a quadratic surface centered at the prior equilibrium point $\boldsymbol{t}_{n-1} = \begin{bmatrix} \boldsymbol{x}_{n-1} \\ \lambda_{n-1} \end{bmatrix}$,

$$L^2 = \|\Delta \boldsymbol{x}\|^2 + \psi^2(\Delta\lambda)^2 = \left\| \boldsymbol{x}_n^{(k)} - \boldsymbol{x}_{n-1} \right\|^2 + \psi^2 \left(\lambda_n^{(k)} - \lambda_{n-1} \right)^2, \tag{6}$$

where L is the radius of the constraint surface, $\Delta \boldsymbol{x}$ and $\Delta\lambda$ are the accumulated updates over the current load increment, and ψ^2 determines the shape of the constraint surface. Generally, $\psi^2 > 0$ leads to a hyper-sphere constraint surface, while $\psi^2 = 0$ leads to a hyper-cylinder constraint surface. Additional parameters can be added to the constraint equation to allow for generalized hyper-ellipsoids as in [12], but we do not consider these here.

We define the linearized equilibrium line to be the set of points for which the equilibrium equations eq. (2), linearized around the previous iterative solution $\boldsymbol{t}^{(k-1)} = \begin{bmatrix} \boldsymbol{x}^{(k-1)} \\ \lambda^{(k-1)} \end{bmatrix}$, are satisfied. This line is defined by the point $\boldsymbol{t}^{(k-1)} + \begin{bmatrix} \delta \boldsymbol{x}^R \\ 0 \end{bmatrix}$ and the vector $\boldsymbol{t}^Q = \begin{bmatrix} \delta \boldsymbol{x}^Q \\ 1 \end{bmatrix}$. This provides a more intuitive meaning to $\boldsymbol{\delta} x^Q$ and $\boldsymbol{\delta} x^R$ as the slope of the equilibrium line and the projected distance of the candidate solution from the equilibrium line, respectively.

The arc length continuation method can be reframed as seeking the intersection of this linearized equilibrium line with the quadratic constraint surface. Importantly, the constraint surface is constant across the entire increment, but the linearized equilibrium line changes at each iteration when a standard Newton-Raphson method is used, as in our implementation. There are additional approaches in [12] which change the constraint surface at each iteration to improve convergence properties, but these are not considered here.

2.2 Choosing the Load Parameter

The load parameter update $\delta\lambda$ is chosen such that the candidate solution $\boldsymbol{t}^{(k)} = \begin{bmatrix}\boldsymbol{x}^{(k)} \\ \lambda^{(k)}\end{bmatrix}$ lies on the constraint surface. We can find such updates by substituting $\Delta\boldsymbol{x}^{(k)}$ and $\Delta\lambda^{(k)}$ into the constraint equation eq. (6) and solving for $\delta\lambda$:

$$L^2 = \left\|\Delta\boldsymbol{x}_n^{(k)}\right\|^2 + \psi^2\left(\Delta\lambda_n^{(k)}\right)^2 = \left\|\Delta\boldsymbol{x}_n^{(k-1)} + \delta s\,\delta\boldsymbol{x}^F + \delta\lambda^{(k-1)}\delta\boldsymbol{x}^Q\right\|^2 + \psi^2\left(\Delta\lambda_n^{(k-1)} + \delta\lambda\right)^2$$

Expanding this yields a quadratic in $\delta\lambda$,

$$a(\delta\lambda)^2 + b\,\delta\lambda + c = 0, \tag{7}$$

where the coefficients a, b, and c are given by,

$$\begin{aligned}
a &= \left\|\delta\boldsymbol{x}^Q\right\|^2 + \psi^2 \\
b &= 2\delta\boldsymbol{x}^Q \cdot \left(\Delta\boldsymbol{x}^{(k-1)} + \delta s\,\delta\boldsymbol{x}^R\right) + 2\psi^2\Delta\lambda^{(k-1)} \\
c &= \left\|\Delta\boldsymbol{x}^{(k-1)} + \delta s\,\delta\boldsymbol{x}^R\right\|^2 + \psi^2\left(\Delta\lambda^{(k-1)}\right)^2 - L^2.
\end{aligned}$$

This quadratic equation has two (possibly complex) roots, which can be found as

$$\delta\lambda^{(k)} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} = \begin{cases}\eta_1 \\ \eta_2\end{cases} \tag{8}$$

This formulation is the basis for Crisfield's method for choosing the load parameter update [6].

2.2.1 Predictor Load Parameter Update

For the first iteration from an equilibrium point, the quadratic equation eq. (8) can be simplified further. Since we assume that the solution exactly satisfies the equilibrium equations at the beginning of each increment, $\Delta\boldsymbol{x}_n^{(k-1)} = \delta\boldsymbol{x}^{R,(j)} = \boldsymbol{0}$ and $\Delta\lambda_n^{(k-1)} = 0$. Thus, $b = 0$ and the equation simplifies to a pair of real roots:

$$\delta\lambda_n^{(0)} = \pm\frac{L}{\sqrt{\left\|\delta\boldsymbol{x}^Q\right\|^2 + \psi^2}}. \tag{9}$$

Following [12], we choose the positive root for the first increment and choose the root based on the prior increment for $n > 0$ as

$$\operatorname{sign}(\delta\lambda^{(k)}) = \operatorname{sign}\left(\delta\boldsymbol{x}^Q \cdot \Delta\boldsymbol{x}_{n-1} + \psi^2\Delta\lambda_{n-1}\right). \tag{10}$$

This choice is proven to be the correct forward direction along the equilibrium path in the neighborhood of $\boldsymbol{t}_{n-1}$ [12].

2.2.2 Corrector Load Parameter Update — Crisfield's Method

Crisfield's method uses the quadratic equation eq. (8) to choose the load parameter update at each iteration. The correct root is chosen to be the one which results in the candidate solution $\boldsymbol{t}^{(k)}$ closest to prior iterative solution $\boldsymbol{t}^{(k-1)}$. The approach proposed by [12] is evaluate the relative proximity of the candidate solutions by maximizing

$$b_0 = \delta \boldsymbol{x}^Q \cdot \Delta \boldsymbol{x}^{(k-1)} + \psi^2 \Delta\lambda^{(k-1)},$$

leading to the choice

$$\delta\lambda^{(k)} = \begin{cases} \eta_1 & \text{if } b_0\eta_1 > b_0\eta_2 \\ \eta_2 & \text{otherwise.} \end{cases} \tag{11}$$

This method encounters issues, however, if the linearized equilibrium line and constraint surface do not intersect, which can occur if there is a particularly large linearized error. In this case, the roots are complex. One option is to reduce the step size, but this failed to resolve the issue of complex roots in our testing. Instead, one can use a partial correction by choosing δs such that the quadratic equation has a single real root [19, 12]. Geometrically, this is selecting the point on the constraint surface closest to the linearized equilibrium line.

For completeness, we include the procedure from [12]. The proper choice of δs can be found by writing the coefficients of eq. (7) as

$$a = a_0, \qquad b = b_0 + b_1\delta s, \qquad c = c_0 + c_1\delta s + c_2\delta s^2,$$

where,

$$\begin{aligned} a_0 &= \|\delta \boldsymbol{x}^Q\|^2 + \psi^2 & c_0 &= \|\Delta \boldsymbol{x}^{(k-1)}\|^2 + \psi^2(\Delta\lambda^{(k-1)})^2 - L^2 \\ b_0 &= 2\delta \boldsymbol{x}^Q \cdot \Delta \boldsymbol{x}^{(k-1)} + 2\psi^2\Delta\lambda^{(k-1)} & c_1 &= 2\delta \boldsymbol{x}^R \cdot \Delta \boldsymbol{x}^{(k-1)} \\ b_1 &= 2\delta \boldsymbol{x}^Q \cdot \delta \boldsymbol{x}^R & c_2 &= \|\delta \boldsymbol{x}^R\|^2. \end{aligned}$$

Then, it follows that $b^2 - 4ac > 0$ if and only if $a_s\delta s^2 + b_s\delta s + c_2 > 0$, where

$$a_s = b_1^2 - 4a_0c_2, \qquad b_s = 2b_1b_0 - 4a_0c_1, \qquad c_s = b_0^2 - 4a_0c_0.$$

Thus, the partial correction is chosen as the maximum root of $a_s\delta s^2 + b_s\delta s + c_2 = 0$,

$$\delta s = \frac{-b_s + \sqrt{b_s^2 - 4a_sc_2}}{2a_s}. \tag{12}$$

Using this partial correction, the load parameter update is then chosen as the single real root of

$$a_0(\delta\lambda)^2 + (b_0 + b_1\delta s)\delta\lambda + c_0 + c_1\delta s + c_2\delta s^2 = 0. \tag{13}$$

2.2.3 Corrector Load Parameter Update — Normal Hyperplane

This option is simpler and computationally less expensive, but may fail to converge, as the constraint equation is not satisfied at every iteration. This approach was originally proposed by Ramm in [10] as a means to avoid the need to find the roots of eq. (7) at each iteration. The implementation we use is based on primarily on [14].

Rather than requiring the constraint to be satisfied at each iteration, the update $\delta\lambda$ is chosen such that the update is within the normal hyper-surface to the equilibrium curve in the augmented

space. To this end, the updated tangent vector to the equilibrium curve is defined as the current incremental update $\hat{\boldsymbol{t}}^{(k-1)} = \begin{bmatrix} \Delta \boldsymbol{x}^{(k-1)} \\ \Delta\lambda^{(k-1)} \end{bmatrix}$. Then, the update $\delta\lambda$ is chosen such that $\begin{bmatrix} \delta \boldsymbol{x} \\ \delta\lambda \end{bmatrix}$ is orthogonal to $\hat{\boldsymbol{t}}^{(k-1)}$. Mathematically, this is equivalent to choosing $\delta\lambda$ such that

$$\begin{aligned} 0 &= \Delta \boldsymbol{x}^{(k-1)} \cdot \delta \boldsymbol{x} + \Delta\lambda^{(k-1)} \delta\lambda \\ &= \Delta \boldsymbol{x}^{(k-1)} \cdot (\delta\lambda \delta \boldsymbol{x}^{Q} + \delta s \delta \boldsymbol{x}^{R}) + \Delta\lambda^{(k-1)} \delta\lambda. \end{aligned}$$

Solving for $\delta\lambda$, and noting that $\delta s = 1$, yields

$$\delta\lambda = -\frac{\Delta \boldsymbol{x}^{(k-1)} \cdot \delta \boldsymbol{x}^{R}}{\Delta \boldsymbol{x}^{(k-1)} \cdot \delta \boldsymbol{x}^{Q} + \Delta\lambda^{(k-1)}}. \tag{14}$$

The original proposed version of this method used the initial tangent vector $\hat{\boldsymbol{t}}^{(0)}$, but this is far less robust than the updated tangent approach [14].

2.3 Bounding the Load Parameter

To support generalized loading conditions, we assume that $\lambda \in [\lambda_{\min}, \lambda_{\max}]$, where, by default, $\lambda_{\min} = 0$ and $\lambda_{\max} = 1$. This formulation assumes that the loading can be defined as a parametric function of λ. To allow for this nonlinear solver to be used as the internal solver for a time-stepping method, we must ensure that $\lambda = \lambda_{\max}$ exactly at the end of the final load increment. This can be accomplished by clamping λ to $[\lambda_{\min}, \lambda_{\max}]$ before computing the solution update. This operation is equivalent to choosing the candidate solution on the linearized equilibrium line near to its intersection with the constraint surface (but not necessarily on it); in other words, the constraint surface is dynamically resized to exactly satisfy $\lambda = \lambda_{\max}$. Practically, this results in a hybrid of arc length continuation and normal Newton iterations during the final load increment. This method still ensures quadratic convergence of the Newton iterations, while ensuring that the final load parameter is exactly $\lambda_{\max}$. To our knowledge, this is a novel approach for ensuring the solution from the arc length method is directly comparable to a standard Newton method.

3 Verification

Both arc-length continuation methods were implemented in PETSc as nonlinear solvers (SNES) and verified using two large deformation nonlinear buckling tests. The implementation is available in the PETSc repository[2] and the verification tests are available on GitHub[3]. The implementation allows for generalized external forces via a user-provided function to compute the tangent load vector $\boldsymbol{Q}$, as well as natively supports an arbitrary level of parallelism through any of the PETSc discretization management interfaces.

To verify our implementation, we extended the nonlinear arch buckling test in SNES Tutorials `ex16` to support arc length continuation of both point and body loading. This example uses a 3D St. Venant-Kirchhoff constitutive model with finite deformations.

[2]See `https://gitlab.com/petsc/petsc/-/merge_requests/7502`.
[3]See `https://github.com/zatkins-dev/arc-length-verification`.

3.1 Constitutive Equations

The constitutive equations for the Saint Venant-Kirchhoff model are given by a linear relationship between the Piola-Kirchhoff stress tensor $\boldsymbol{S}$ and the Green-Lagrange strain tensor $\boldsymbol{E}$,

$$\boldsymbol{S} = 2\mu\boldsymbol{E} + \lambda \operatorname{tr}(\boldsymbol{E})\mathbf{1}, \tag{15}$$

$$\boldsymbol{E} = \frac{1}{2}\left(\boldsymbol{F}^T\boldsymbol{F} - \mathbf{1}\right), \tag{16}$$

$$\boldsymbol{F} = \nabla_{\boldsymbol{X}}\boldsymbol{u} + \mathbf{1}, \tag{17}$$

where $\boldsymbol{F}$ is the deformation gradient, $\mathbf{1}$ is the second-order identity tensor, $\nabla_{\boldsymbol{X}}\boldsymbol{u}$ is the gradient of the displacement field $\boldsymbol{u}$ with respect to the initial configuration $\boldsymbol{X} = \boldsymbol{x} - \boldsymbol{u}$, and μ and λ are the Lamè constants [16]. This constitutive model has limited utility for large strains, but is commonly used for analysis of snap-through behavior for beam and shell elements [16].

3.2 Weak Form and FEM Discretization

The weak form of the equilibrium equations is given by: find $\boldsymbol{u} \in V \subset H^1(\Omega_0)$ such that

$$\underbrace{\int_{\Omega_0} \nabla\boldsymbol{v} : (\boldsymbol{F}\cdot\boldsymbol{S})\ \mathrm{d}\Omega_0}_{\boldsymbol{F}^{\text{int}}} - \underbrace{\int_{\Omega_0} \boldsymbol{v}\cdot\boldsymbol{f}_{dist}\ \mathrm{d}\Omega_0}_{\text{Body Loading}} - \underbrace{\int_{\Gamma_p} \boldsymbol{v}\cdot\boldsymbol{f}_p\ \mathrm{d}A}_{\text{Applied force BCs}} = 0, \quad \forall\boldsymbol{v} \in V, \tag{18}$$

where f_{dist} is the distributed body force and $\boldsymbol{f}_p$ is the point force applied on the boundary $\Gamma_p \subset \partial\Omega_0$. The left-most term is the internal force and the remaining terms are the external forces, which are treated as linear in the load parameter λ. The weak form is discretized using a standard Galerkin finite element method with linear hexahedral elements. The PETSc example already implemented the discretization and the tangent stiffness matrix, so we exclude their description for brevity.

3.3 Results

3.3.1 Arch Buckling Test

The arch buckling test was already implemented into the PETSc example and originates from a 2D problem in [16]. This test uses a 60 degree arch with a inner radius of 100 mm, an outer radius of 103 mm and a thickness of 3 mm. The arch uses a St. Venant-Kirchhoff material model with $\nu = 0.48$ and $E = 40$ GPa. We use a Q1 (8 node) hexahedral element mesh with a total of 50, 3, and 2 elements in the x, y, and z directions, respectively. The arch is loaded with a downward point force of 12 kN at the nodes at the center of the arch. Each end of the arch is fixed along the $y+$ edge and is free to pivot around the z axis. The z degrees of freedom are all constrained to be zero to better replicate the 2D problem. See fig. 1 for a schematic of the arch.

In our experiment, we verify the arc length continuation method by comparing the results of the arc length continuation method to the standard Newton method using backward Euler time stepping. For the arc length methods, we use a step size of $L = 10$ and a consistency parameter of $\psi^2 = 1$ (spherical constraint surface). For the Newton method and time stepping, we use a time step of $\Delta t = 0.05$. Figure 2a shows the force-displacement curve for the arch under point loading. While both methods converge, the arc length methods converge more quickly, requiring 18 increments and 4 Newton iterations per increments for both Crisfield's method and the normal plane approximations. By comparison, the Newton method requires 20 increments and an average of

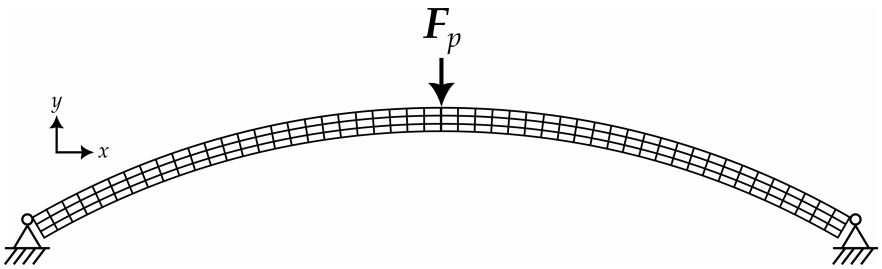
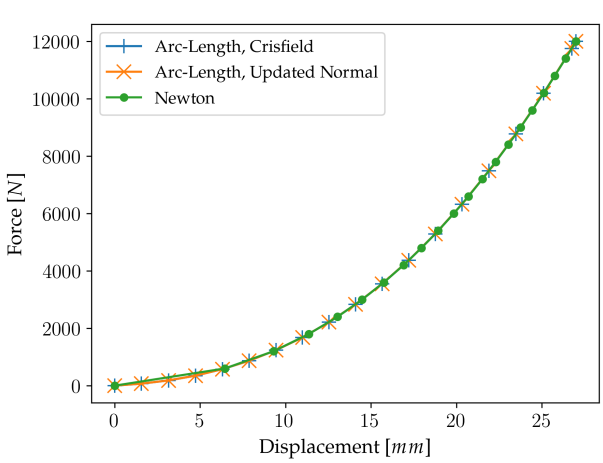
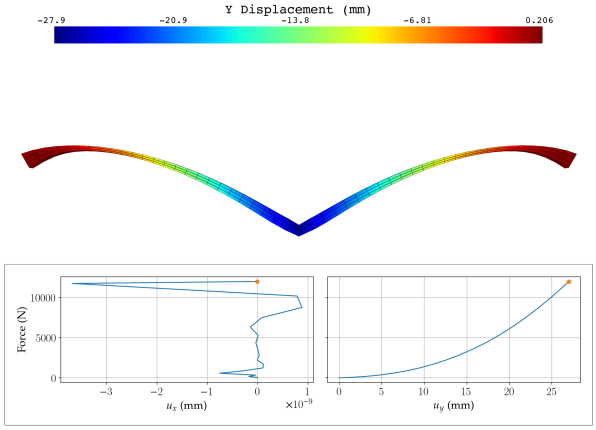


Figure 1: Schematic of the arch buckling test.



(a) Force-displacement curve for the top-center node of the arch buckling test.



(b) Deformed arch mesh at peak loading of 12 kN along the top-center nodes of the arch.

Figure 2: Results of the arch buckling test.

12.6 Newton iterations per increment. The first increment accounts for 103 of the total 251 Newton iterations, with the final 11 increments requiring 5 iterations each. This is likely due to the large jump along the force-displacement curve during the first increment of time stepping, which is not present in the arc length methods. In fact, the arc length methods resolve the force-displacement curve more regularly than the Newton method, despite having fewer increments. The final deformed mesh is shown in fig. 2b.

3.3.2 Lee Frame Snap-Through Test

The Lee frame snap-through test is a 2D problem from [7] which models snap-through instability in a L-shaped frame. This test has been used extensively to verify arc length continuation methods [12, 8, 7, 13, 9]. Each side of the frame is 120 cm long, with a thickness of 2 cm in the y direction and 3 cm in the z direction as shown in fig. 3. Typically, this problem is solved using isoparametric, geometrically-exact beam elements for which it has an analytical solution, but PETSc does not easily support these elements and their implementation is beyond the scope of this work. Instead, we use the same 3D St. Venant-Kirchhoff material model as in the arch buckling test. The material properties are $\nu = 0.3$ and $E = 720\,\text{MPa}$. Unlike in the prior test, we apply the downward force as a body load with a force density of $-1\,\text{kN/cm}^3$ in the y direction within the greyed element in fig. 3. The x and y displacements of the frame are fixed along the outside edges on the small right

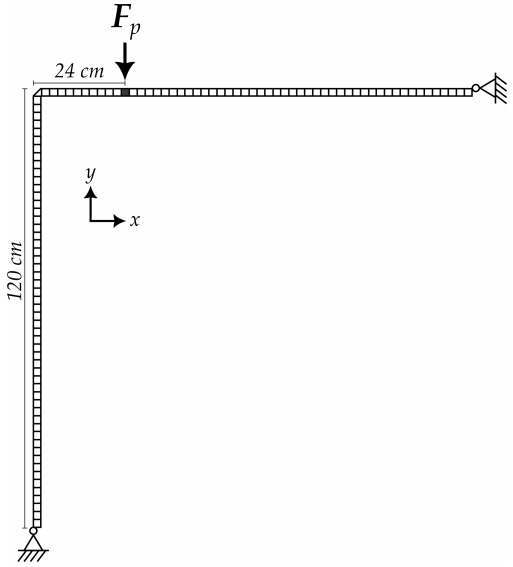
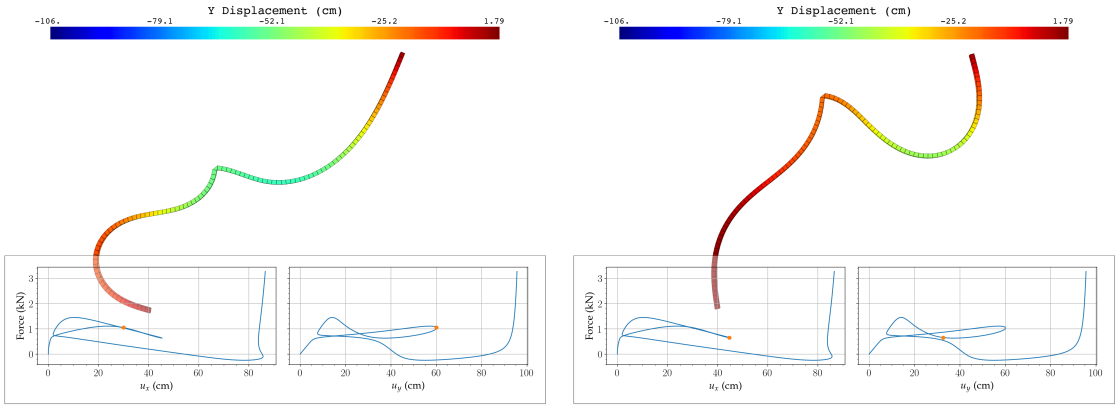


Figure 3: Schematic of the Lee frame snap-through test.

and bottom faces, allowing it to freely pivot around either edge. The z degrees of freedom are all constrained to be zero to better replicate the 2D problem.

This test is more challenging than the arch buckling test, as it involves a snap-through instability and reversed loading path, as well as multiple bifurcation points. In order to avoid some bifurcations, we use a step size of $L = 10$ and a consistency parameter of $\psi^2 = 1$, leading to 495 load increments. Many of these increments are likely unnecessary, however, as the average number of Newton iterations per increment is 4, with a maximum of 9. The number of increments could be decreased by adapting the step size based on the convergence profile, which is a topic for future work. Figure 4 shows the force-displacement curve and deformed configuration for the Lee frame at several of the critical points along the loading path. As expected, the methods are able



(a) Step 110, load and displacement limit point.

(b) Step 171, load limit point.

(c) Step 255, displacement limit point.

(d) Step 421, load limit point.

Figure 4: Deformed mesh of the Lee frame snap-through test at various critical points.

to capture the snap-through instability and reversed loading path. Newton's method with time stepping was unable to pass the first bifurcation point, which can be seen as the crossing point in the force-displacement curve. Future work will implement beam elements to verify the results of this test against other continuation methods.

4 Conclusion

The two arc length continuation methods are implemented in PETSc as new nonlinear solvers, pending review and inclusion in the main repository. The methods are able to capture snap-through instabilities and reversed loading paths, as well as multiple bifurcation points, in ways previously impossible within PETSc. Additionally, the methods are able to solve some load-driven problems with fewer iterations and increments than the standard Newton method while better resolving the force-displacement curve. The methods are also able to handle problems with large jumps in the force-displacement curve, such as the arch buckling test, which the Newton method struggles with. While these methods may not be as efficient as the Newton method for some problems, they are able to solve problems that the Newton method cannot. By implementing them in PETSc, this work will allow for the solution of a wider range of nonlinear problems in a portable, highly parallelizable framework.

References

[1] Marco Amabili and Mohammad Reza Sareban Tajahmadi. "Thermal post-buckling of laminated and isotropic rectangular plates with fixed edges: Comparison of experimental and numerical results". In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 226.10 (Oct. 2012), pp. 2393–2401. ISSN: 0954-4062, 2041-2983. DOI: 10.1177/0954406211434496. URL: http://journals.sagepub.com/doi/10.1177/0954406211434496 (visited on 04/01/2024).

[2] Satish Balay et al. *PETSc/TAO Users Manual*. Tech. rep. ANL-21/39 - Revision 3.21. Argonne National Laboratory, 2024. DOI: 10.2172/2205494.

[3] Jean-Louis Batoz and Gouri Dhatt. "Incremental displacement algorithms for nonlinear problems". In: *International Journal for Numerical Methods in Engineering* 14.8 (1979). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.1620140811, pp. 1262–1267. ISSN: 1097-0207. DOI: 10.1002/nme.1620140811. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.1620140811 (visited on 04/28/2024).

[4] R. de Borst. "Computation of post-bifurcation and post-failure behavior of strain-softening solids". In: *Computers & Structures* 25.2 (Jan. 1, 1987), pp. 211–224. ISSN: 0045-7949. DOI: 10.1016/0045-7949(87)90144-1. URL: https://www.sciencedirect.com/science/article/pii/0045794987901441 (visited on 04/28/2024).

[5] Alberto Carpinteri and Federico Accornero. "Multiple snap-back instabilities in progressive microcracking coalescence". In: *Engineering Fracture Mechanics* 187 (Jan. 2018), pp. 272–281. ISSN: 00137944. DOI: 10.1016/j.engfracmech.2017.11.034. URL: https://linkinghub.elsevier.com/retrieve/pii/S0013794417310512 (visited on 04/01/2024).

[6] M. A. Crisfield. "A FAST INCREMENTAL/ITERATIVE SOLUTION PROCEDURE THAT HANDLES "SNAP-THROUGH"". In: *Computational Methods in Nonlinear Structural and Solid Mechanics*. Ed. by AHMED K. Noor and HARVEY G. McCOMB. Pergamon, Jan. 1, 1981, pp. 55–62. ISBN: 978-0-08-027299-3. DOI: 10.1016/B978-0-08-027299-3.50009-1. URL: https://www.sciencedirect.com/science/article/pii/B9780080272993500091 (visited on 04/01/2024).

[7] Seng-Lip Lee, Francisco S. Manuel, and Edwin C. Rossow. "Large Deflections and Stability of Elastic Frame". In: *Journal of the Engineering Mechanics Division* 94.2 (1968), pp. 521–548. DOI: 10.1061/JMCEA3.0000966. eprint: https://ascelibrary.org/doi/pdf/10.1061/JMCEA3.0000966. URL: https://ascelibrary.org/doi/abs/10.1061/JMCEA3.0000966.

[8] Sofie E. Leon et al. "A Unified Library of Nonlinear Solution Schemes". In: *Applied Mechanics Reviews* 64.4 (July 1, 2011), p. 040803. ISSN: 0003-6900, 2379-0407. DOI: 10.1115/1.4006992. URL: https://asmedigitalcollection.asme.org/appliedmechanicsreviews/article/doi/10.1115/1.4006992/463373/A-Unified-Library-of-Nonlinear-Solution-Schemes (visited on 04/13/2024).

[9] E. Parente Jr. and L. E. Vaz. "Improvement of semi-analytical design sensitivities of non-linear structures using equilibrium relations". In: *International Journal for Numerical Methods in Engineering* 50.9 (2001). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.115, pp. 2127–2142. ISSN: 1097-0207. DOI: 10.1002/nme.115. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.115 (visited on 05/04/2024).

[10] E. Ramm. "Strategies for Tracing the Nonlinear Response Near Limit Points". In: *Nonlinear Finite Element Analysis in Structural Mechanics*. Ed. by W. Wunderlich, E. Stein, and K.-J. Bathe. Berlin, Heidelberg: Springer Berlin Heidelberg, 1981, pp. 63–89. ISBN: 978-3-642-81591-1 978-3-642-81589-8. DOI: 10.1007/978-3-642-81589-8_5. URL: http://link.springer.com/10.1007/978-3-642-81589-8_5 (visited on 04/01/2024).

[11] E. Riks. "An incremental approach to the solution of snapping and buckling problems". In: *International Journal of Solids and Structures* 15.7 (1979), pp. 529–551. ISSN: 00207683. DOI: 10.1016/0020-7683(79)90081-7. URL: https://linkinghub.elsevier.com/retrieve/pii/0020768379900817 (visited on 04/01/2024).

[12] Manuel Ritto-Corrêa and Dinar Camotim. "On the arc-length and other quadratic control methods: Established, less known and new implementation procedures". In: *Computers & Structures* 86.11 (June 1, 2008), pp. 1353–1368. ISSN: 0045-7949. DOI: 10.1016/j.compstruc.2007.08.003. URL: https://www.sciencedirect.com/science/article/pii/S0045794907002696 (visited on 04/09/2024).

[13] Karl Schweizerhof and Peter Wriggers. "Consistent linearization for path following methods in nonlinear FE Analysis". In: *Computer Methods in Applied Mechanics and Engineering* 59 (July 2, 1986), pp. 261–279. DOI: 10.1016/0045-7825(86)90001-0.

[14] J. C. Simo et al. "Finite deformation post-buckling analysis involving inelasticity and contact constraints". In: *International Journal for Numerical Methods in Engineering* 23.5 (May 1986), pp. 779–800. ISSN: 0029-5981, 1097-0207. DOI: 10.1002/nme.1620230504. URL: https://onlinelibrary.wiley.com/doi/10.1002/nme.1620230504 (visited on 04/01/2024).

[15] Xing Wang, Tian-Bao Ma, and Jian-Guo Ning. "A Pseudo Arc-Length Method for Numerical Simulation of Shock Waves". In: *Chinese Physics Letters* 31.3 (Mar. 2014), p. 030201. ISSN: 0256-307X, 1741-3540. DOI: 10.1088/0256-307X/31/3/030201. URL: https://iopscience.iop.org/article/10.1088/0256-307X/31/3/030201 (visited on 04/01/2024).

[16] Peter Wriggers. *Nonlinear finite element methods*. Berlin Heidelberg: Springer, 2008. 559 pp. ISBN: 978-3-540-71000-4 978-3-642-09002-8.

[17] Zhijian Wu, Li Guo, and Jun Hong. "Improved Staggered Algorithm for Phase-Field Brittle Fracture with the Local Arc-Length Method". In: *Computer Modeling in Engineering & Sciences* 134.1 (2022). Publisher: Tech Science Press, pp. 611–636. ISSN: 1526-1492, 1526-1506. DOI: 10.32604/cmes.2022.020694. URL: https://www.techscience.com/CMES/v134n1/49446 (visited on 04/01/2024).

[18] Xiao Bin Yang. "The Arc-Length Method from Controlling Load to Controlling Electric Potential". In: *Applied Mechanics and Materials* 33 (Oct. 2010), pp. 636–641. ISSN: 1662-7482. DOI: 10.4028/www.scientific.net/AMM.33.636. URL: https://www.scientific.net/AMM.33.636 (visited on 04/09/2024).

[19] Zhilong Zhou and D. W. Murray. "An incremental solution technique for unstable equilibrium paths of shell structures". In: *Computers & Structures* 55.5 (June 3, 1995), pp. 749–759. ISSN: 0045-7949. DOI: 10.1016/0045-7949(94)00474-H. URL: https://www.sciencedirect.com/science/article/pii/004579499400474H (visited on 04/28/2024).