A Monolithic Homotopy Continuation Algorithm with Application to Computational Fluid Dynamics

David A. Brown and David W. Zingg

University of Toronto Institute for Aerospace Studies, Toronto, Ontario, M3H 5T6, Canada

Abstract

A new class of homotopy continuation methods is developed suitable for globalizing quasi-Newton methods for large sparse nonlinear systems of equations. The new continuation methods, described as monolithic homotopy continuation, differ from the classical predictor-corrector algorithm in that the predictor and corrector phases are replaced with a single phase which includes both a predictor and corrector component. Conditional convergence and stability are proved analytically. Using a Laplacian-like operator to construct the homotopy, the new algorithm is shown to be more efficient than the predictor-corrector homotopy continuation algorithm as well as the widely-used pseudo-transient continuation algorithm for some inviscid and turbulent, subsonic and transonic external aerodynamic flows over the ONERA M6 wing and the NACA 0012 airfoil using a parallel implicit Newton-Krylov finite-difference flow solver.

Keywords: homotopy, continuation, globalization, Newton-Krylov, computational fluid dynamics

1. Introduction

Newton's method is a popular fixed-point iterative method for solving systems of nonlinear equations due to its quadratic convergence rate. When combined with an iterative Krylov solver such as the Generalized Minimal Residual method (GMRES) for solving the linear systems of equations that arise, the method becomes an inexact Newton method, but the convergence rate can still be super-linear or even quadratic [10]. However, it is well-known that Newton's method will usually not converge without a suitable initial guess. Obtaining the initial guess is referred to as globalization and is performed using a continuation method. Thus, solving a nonlinear system of equations with an inexact Newton method generally consists of two phases: a globalization phase and an inexact Newton phase.

When solving a large sparse nonlinear algebraic system of equations, such as those arising in computational fluid dynamics (CFD) when the steady flow equations are discretized in space, the globalization phase can be computationally very expensive, often more expensive than the inexact Newton phase. At the current state of CFD development and computer technology, many CFD problems of interest remain computationally demanding so the efficiency of the nonlinear equation solver is of engineering importance for CFD practitioners. In addition, the algorithm may not always converge to the solution; some algorithms may stall or become unstable. The reliability of the algorithm to converge to the solution is known as robustness and is also of importance. Many algorithms contain tunable parameters to allow for a user to balance speed and

robustness. However, regular tuning of parameters is undesirable and is also a consideration for algorithm design: parameters should be few, intuitive, and should not need regular retuning.

The most common continuation method for globalizing steady CFD problems is pseudo-transient continuation (PTC) [4, 8, 23, 24]. This is a method which imitates physical time marching, though time-accuracy is not required. A viable alternative to PTC is homotopy continuation. Homotopy continuation methods are discrete algorithms based on continuous deformations known as homotopies. The most common application of homotopy continuation in the literature has been for specialized study of systems with certain mathematical properties. For example, homotopy continuation can be used to study systems where multiple solutions exist [31, 35] or where solutions may be unstable [34, 22] in a more methodical and reliable way than can be done with a method such as PTC. Homotopy continuation has also been applied to facilitate the solution to CFD problems at high Reynolds numbers by solving the same problem at a lower Reynolds number and gradually increasing the Reynolds number [5].

Hicken and Zingg [20] and Hicken et al. [18] designed a homotopy continuation algorithm which they found could be used as a general globalization method for a Newton-Krylov external aerodynamic flow solver developed by Hicken and Zingg [19] and Osusky and Zingg [29]. They found that the continuation method, which they termed *dissipation-based continuation* since the homotopy was constructed based on a dissipation operator, was more efficient than PTC in some cases, especially inviscid flows, but was not as competitive for turbulent flows. To model the turbulent flows, they used the Reynolds-averaged Navier-Stokes equations with Spalart-Allmaras [33] turbulence model (RANS-SA). We subsequently made several improvements to their formulation, including the implementation of a classical predictor-corrector approach adapted to the Newton-Krylov algorithm [3]. For the cases that we investigated, we found that the homotopy continuation method performed better than PTC for some three-dimensional inviscid and laminar cases, as well as some two-dimensional RANS cases, but did not perform as well for some three-dimensional RANS cases.

Homotopy continuation algorithms based on predictor-corrector methods are prevalent in the homotopy literature; we refer to Allgower and Georg [1] and the references therein. In a context where the homotopy continuation algorithm is needed to be as efficient as possible, predictor-corrector algorithms can be wasteful because, in order to ensure that the algorithm is convergent, parameters are chosen such that the corrector phase is often over-solved, by which we mean that much of the work done in the corrector phase leads to only marginal improvement to the quality of the predictor update. In addition, depending on the predictor method and corrector method employed, both of these phases may require the solution to linear systems of equations and the algorithm can be made more efficient if these linear solves are combined. These potential gains in computational efficiency have been the motivation for developing the monolithic homotopy (MH) continuation algorithm. The MH algorithm was developed based on a principle known as dynamic inversion [14, 15, 16], which is the mathematically analogous problem of inexactly predicting an implicitly-defined trajectory traced by a dynamic system. In addition to the gains in efficiency, combining the predictor and corrector phases into a single phase reduces the number of user parameters, simplifying user control. The objective of this paper is to present the monolithic homotopy continuation algorithm, to characterize its performance on some compressible flow problems, and to compare its performance to both a typical PTC algorithm and a homotopy continuation algorithm based on a predictor-corrector method.

2. Flow Solver

The flow solver to which the monolithic homotopy continuation algorithm is applied is a Newton-Krylov-Schur parallel implicit flow solver based on a finite-difference [26] discretization applicable to multi-block structured grids. The finite-difference discretization is based on the SBP-SAT [6, 9, 12, 25] approach, which uses Summation-By-Parts (SBP) operators to represent the discrete derivatives and Simultaneous Approximation Terms (SATs) to enforce the boundary conditions and couple the flow equations at block interfaces. The flow solver originated as an inviscid flow solver due to Hicken and Zingg [19] and was extended to the RANS-SA equations by Osusky and Zingg [29]. The flow solver can be used for both subsonic and transonic operating conditions. For transonic cases, a first-order dissipation operator is included with a pressure sensor [21] for shock capturing. To parallelize the flow solver, the domain is decomposed into blocks. Parallel preconditioning of the Krylov solver is performed using the Schur complement method [32] with block incomplete lower-upper (ILU) preconditioning applied to the domain blocks. The specific type of ILU factorization used in the current study is known as ILU(p) [32], where p is the fill level. The ILU(p) factorization is constructed based on an approximate Jacobian matrix using nearest neighbour nodes only. Since the Schur complement preconditioner can vary slightly throughout the Krylov solution process, a flexible variant of the Krylov solver GMRES is used, which is termed Flexible Generalized Minimal Residual, or FGMRES [32].

3. Jacobian-Free Newton-Krylov Method

Consider a nonlinear algebraic system of equations, represented by

$$\mathcal{F}(\mathbf{q}) = \mathbf{0}, \tag{1}$$

$$\mathcal{F} : \mathbb{R}^N \rightarrow \mathbb{R}^N, \ \mathbf{q} \in \mathbb{R}^N.$$

The update due to Newton's method, when applied to this system of equations, is calculated by solving the linear system of equations

$$\nabla\mathcal{F}^{(n)}(\mathbf{q}) \ \Delta\mathbf{q}^{(n)} = -\mathcal{F}\left(\mathbf{q}^{(n)}\right), \tag{2}$$

$$\Delta\mathbf{q}^{(n)} \equiv \mathbf{q}^{(n+1)} - \mathbf{q}^{(n)},$$

where $\nabla\mathcal{F}^{(n)} : \mathbb{R}^N \rightarrow \mathbb{R}^N$ is the Jacobian of $\mathcal{F}(\mathbf{q})$, defined as

$$\nabla\mathcal{F}_{i,j}(\mathbf{q}) \equiv \frac{\partial\mathcal{F}_i(\mathbf{q})}{\partial\mathbf{q}_j}, \tag{3}$$

which can be represented by a square matrix. Since the linear system (2) is being solved to some relative tolerance $\tau_1^{(n)} \in \mathbb{R}$, the actual Newton step is taken inexactly, and the update $\Delta\mathbf{q}$ does not satisfy equation (2) but does satisfy the inequality

$$\left\|\mathcal{F}\left(\mathbf{q}^{(n)}\right) + \nabla\mathcal{F}^{(n)}(\mathbf{q}) \ \Delta\mathbf{q}^{(n)}\right\| \leq \tau_1^{(n)} \left\|\mathcal{F}\left(\mathbf{q}^{(n)}\right)\right\|. \tag{4}$$

Newton's method is not expected to converge unless a suitable initial guess of the solution is obtained, typically from a globalization algorithm. It can be advantageous to augment Newton's method with a line search algorithm in order to improve the success rate in case of insufficient

globalization. From some experimentation we have found that including a line search capability in the Newton phase can slightly improve the success rate of the algorithm but is not a requirement in that we did not observe any cases where it was only possible to obtain a converged solution when the line search was active. Since augmenting Newton's method with line search can provide a similar benefit to any of the continuation algorithms investigated, it is not expected to have a dramatic impact on the outcome of the performance comparisons presented in Section 10 and so we choose to omit it for simplicity.

It is not always necessary to form and store the Jacobian matrix in a Newton-Krylov algorithm [24] since this matrix is not needed explicitly by a Krylov iterative method for solving linear systems, such as GMRES. What is needed by a Krylov solver are the Jacobian-vector products, which can be approximated by the following finite-difference approximation:

$$\nabla_{\mathbf{q}}\mathcal{F}\left(\mathbf{u}\right)\mathbf{v} \approx \frac{\mathcal{F}\left(\mathbf{u}+\epsilon\mathbf{v}\right)-\mathcal{F}\left(\mathbf{u}\right)}{\epsilon}, \tag{5}$$

where the following expression for $\epsilon \in \mathbb{R}$ is used in order to balance truncation error and rounding error [19, 28]:

$$\epsilon = \sqrt{\frac{N\delta}{\mathbf{v}^T\mathbf{v}}}, \tag{6}$$

with $\delta \in \mathbb{R}$ typically taken around 10^{-12}, which is about 10^3 times machine precision when using double-precision arithmetic.

An alternative way to approximate the Jacobian-vector products is to use the approximate Jacobian matrix used in forming the ILU(p) factorization in place of the exact Jacobian. This results in a less accurate approximation than the finite-difference formulation but comes at reduced cost. We have generally found it to be more cost-effective to use the approximate matrix-vector products based on the preconditioner matrix in the continuation phase for both PTC and the homotopy methods but not in the inexact Newton phase due to very slow convergence rate of the inexact Newton iterations. However, when using the homotopy algorithms, the additional accuracy due to the finite-difference approximation can sometimes be important in the continuation phase.

4. Pseudo-Transient Continuation

The update formula is given by the implicit Euler method with local time linearization [26]:

$$\left(\mathcal{T}^{(n)} + \nabla\mathcal{R}^{(n)}\right)\Delta\mathbf{q}^{(n)} = -\mathcal{R}\left(\mathbf{q}^{(n)}\right), \tag{7}$$

where $\mathcal{T}^{(n)} = \frac{1}{\Delta t}\mathcal{I}$, and $\mathcal{I}$ is the identity matrix. Since time-accuracy is not required in the context of globalization, Δt can take large values and vary spatially. This can be done for the sake of efficiency.

Many update formulae for Δt have been used over the years [4, 23] and some authors have taken more sophisticated approaches based on PTC to achieve improve convergence; see Ceze and Fidkowski [7], for example, who formulated the PTC update with physicality constraints as an optimization problem with the objective of designing a robust algorithm.

Despite the extensive literature on PTC algorithms, due to practical considerations the present study is limited to the update formula employed by Hicken and Zingg [19], which, based on

experimentation, has been found to be efficient in comparison with other algorithms such as the popular SER [27] algorithm. The time step Δt is evolved according to:

$$\Delta t_i^{(n)} = \frac{\Delta t_{\text{ref}}^{(n)} \mathcal{J}_i}{1 + \mathcal{J}_i^{\frac{1}{D}}}, \;\; \Delta t_{\text{ref}}^{(n)} = a\,(b)^{m\left\lfloor \frac{n}{m} \right\rfloor}, \tag{8}$$

where $\mathcal{J}$ is the geometric Jacobian resulting from the coordinate transformation on the mesh, i is the grid point index, D is the number of spatial dimensions, and $\lfloor \cdot \rfloor$ is the floor operator[1]. The floor operator is present in the formula in the case where we choose to update the preconditioner every m iterations instead of every iteration. Hicken and Zingg [19] suggest $a = 0.1$, $b = 1.7$ for inviscid flows. In the studies in this paper, we have chosen more conservative parameter values of $a = 0.01$, $b = 1.4$ to increase the success rate over the wide range of operating conditions considered. For turbulent flows, Osusky and Zingg [29] have suggested $a = 0.001$, $b = 1.3$. Again, we have opted to use more conservative values for the performance studies to improve the robustness. For example, for one of the turbulent ONERA M6 cases we use $a = 0.001$, $b = 1.1$.

PTC is terminated and the inexact Newton phase initiated when the relative residual

$$\mathcal{R}_{\text{rel}}^{(n)} \equiv \frac{\left\| \mathcal{R}\left(\mathbf{q}^{(n)}\right) \right\|}{\left\| \mathcal{R}\left(\mathbf{q}^{(0)}\right) \right\|} \tag{9}$$

is reduced below some user-specified tolerance τ_{rel}. Typical values of τ_{rel} are $\tau_{\text{rel}} \approx 10^{-1}$ to 10^{-2} for the Euler equations and $\tau_{\text{rel}} \approx 10^{-3}$ to 10^{-4} for the RANS-SA equations.

When the PTC method is used for globalization, the inexact Newton update is also modified to take the form of the implicit Euler update (8). The time step matrix is carried over from the last iteration of the PTC phase but the reference time step Δt_{ref} is replaced with the successive evolution relaxation used by Hicken and Zingg [19], which was adapted from Mulder and van Leer [27]. The update is given as

$$\Delta t_{\text{ref}}^{(n)} = \max\left[\alpha \left(R_d^{(n)} \right)^{-\beta}, \Delta t_{\text{ref}}^{(n-1)} \right], \tag{10}$$

$$\alpha = a b^{m\left\lfloor \frac{n_{\text{Newt}}}{m} \right\rfloor} \left(R_d^{(n_{\text{Newt}})} \right)^{\beta}, \;\; R_d^{(n)} \equiv \frac{\left\| \mathcal{R}^{(n)} \right\|}{\left\| \mathcal{R}^{(0)} \right\|},$$

where n_{Newt} is the first inexact Newton iteration. This update formula tends to increase Δt_{ref} rapidly in the inexact Newton phase so that the contribution of $\mathcal{T}$ quickly becomes small, so this modification can be seen as a short transition region between the pseudo-transient phase and inexact Newton phase.

A pseudo-code of the PTC algorithm is provided as Algorithm 1.

5. Convex Homotopy Continuation

Consider a nonlinear system of equations $\mathcal{R}(\mathbf{q}) = \mathbf{0}$, $\mathcal{R} : \mathbb{R}^N \to \mathbb{R}^N$ as well as the so-called *convex homotopy* [1] which is defined as the (presumably) continuous solution $\mathbf{q}(\lambda)$ to

$$\mathcal{H}(\mathbf{q}, \lambda) = (1 - \lambda)\,\mathcal{R}(\mathbf{q}) + \lambda \mathcal{G}(\mathbf{q}) = \mathbf{0}, \tag{11}$$

[1] $\lfloor x \rfloor = y$, where y is the largest integer less than or equal to x.

Algorithm 1: Pseudo-transient continuation (PTC) algorithm

Choose a starting guess for $\mathbf{q}$ (e.g. free-stream)
while $||\mathcal{R}|| > \tau_{\text{rel}}$ *and* $||\mathcal{R}|| > \tau_{\text{abs}}$ **do**
  Get $\mathcal{R}$ and $||\mathcal{R}||$
  Set the reference time step $\Delta t_{\text{ref}} \leftarrow a \cdot b^{\lfloor n/m \rfloor}$
  Set the time step diagonal matrix $\mathcal{T}[:] \leftarrow \left(1 + \mathcal{J}^{1/D}[:]\right) / (\Delta t_{\text{ref}} \mathcal{J}[:])$
  Form and factor the preconditioner based on the matrix $\mathcal{T} + \nabla\mathcal{R}$
  Solve $[\mathcal{T} + \nabla\mathcal{R}]\,\Delta\mathbf{q} = -\mathcal{R}$ inexactly for $\Delta\mathbf{q}$ and update $\mathbf{q} \leftarrow \mathbf{q} + \Delta\mathbf{q}$
end

$$\mathcal{H} : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N,\ \mathcal{G} : \mathbb{R}^N \to \mathbb{R}^N,\ \mathcal{R} : \mathbb{R}^N \to \mathbb{R}^N,\ \lambda \in \mathbb{R}.$$

Since the homotopy can be constructed by any means of our choosing, it is sensible to use the convex formulation, as it is linear with respect to λ, and λ is conveniently bounded in the interval $[0, 1]$. Interpreting the homotopy as a curve existing in $\mathbb{R}^N$, a continuation method, called convex homotopy continuation (CHC) can be developed from this homotopy by discretizing in λ to form a sequence of nonlinear equations:

$$\mathcal{H}(\mathbf{q}, \lambda_k) = (1 - \lambda_k)\mathcal{R}(\mathbf{q}) + \lambda_k \mathcal{G}(\mathbf{q}) = \mathbf{0}, \tag{12}$$

$$k \in [0, m],\ \lambda_k \in \mathbb{R},\ \lambda_0 = 1,\ \lambda_m = 0,\ \lambda_{k+1} < \lambda_k,$$
$$\mathcal{H} : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N,\ \mathcal{G} : \mathbb{R}^N \to \mathbb{R}^N,\ \mathcal{R} : \mathbb{R}^N \to \mathbb{R}^N.$$

Solving $\mathcal{H}(\mathbf{q}, \lambda) = \mathbf{0}$ for sequentially increasing k is referred to as *traversing*.

In this paper it is assumed that the curve is monotonically decreasing. This is consistent with our experience with the specific homotopies which we have generated, such as the homotopy generated using the dissipation operator as described in Section 6. However, to our knowledge, in the absence of an analytical expression for the homotopy curve, it is not possible to know prior to numerical investigation whether or not a homotopy contains bifurcation points. One common type of bifurcation is a Hopf bifurcation in which the curve begins progressing in the opposite direction with respect to λ. For example, λ, which is normally decreasing, may at some point begin to increase as the curve is traced. Such points pose several challenges, one of which is that they are irregular points. That is, the Jacobian $\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}, \lambda)$ is singular at such points. Tracing curves with bifurcation points has many applications and has been studied extensively in the homotopy literature - see Allgower and Georg [1] and the references therein or Wales et al. [34] to give a more recent example. However, tracing curves featuring bifurcations is more challenging for problems on the scale that we are studying and our codes do not currently possess this capability.

6. The Dissipation Operator as Homotopy System

For all studies in this paper, the homotopy system $\mathcal{G}(\mathbf{q})$ is taken as the dissipation operator with far-field boundary conditions. As mentioned in the introduction, Hicken et al. [18] used a second-difference dissipation operator as the homotopy system. This operator was augmented with pseudo boundary conditions by Brown and Zingg [3], which is the form employed in this

paper.

For simplicity, we give only the version of the operator that would correspond to a scalar one-dimensional problem. Extension to higher dimensional systems is straightforward; see, for example, Pulliam [30] for the dissipation operator in two spatial dimensions. Explicitly, the one-dimensional scalar version of the operator is:

$$\mathcal{D}^{(2)} = D^T C D, \tag{13}$$

$$D : \mathbb{R}^N \to \mathbb{R}^{N-1},\ C : \mathbb{R}^{N-1} \to \mathbb{R}^{N-1},$$

$$D = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{bmatrix},\ C = \begin{bmatrix} d_{1\frac{1}{2}} & & & \\ & d_{2\frac{1}{2}} & & \\ & & \ddots & \\ & & & d_{N-\frac{1}{2}} \end{bmatrix},$$

$$d_{i+\frac{1}{2}} = \frac{1}{2}\left(d_i + d_{i+1}\right),\ d_i = \frac{1}{\Delta x_i}\left(|u_i| + a_i\right),$$

where $\Delta x_i \in \mathbb{R}$ is the local grid spacing, $u_i \in \mathbb{R}$ is the local fluid velocity, and $a_i \in \mathbb{R}$ is the local sound speed.

That D defines a mapping from $\mathbb{R}^N$ to $\mathbb{R}^{N-1}$ is sufficient to identify that the range of $\mathcal{D}^{(2)}$ is spanned by at most $N-1$ linearly independent vectors. Therefore $\mathcal{D}^{(2)}$ is singular and in the context of convex homotopy it is a necessary condition for regularity that $\nabla\mathcal{G}(\mathbf{q})$ be non-singular. This can be achieved by augmenting $\mathcal{D}^{(2)}$ with pseudo boundary conditions.

Pseudo boundary conditions for $\mathcal{D}^{(2)}$ are formulated using the SAT approach to be consistent with the application of the boundary conditions for the discrete flow equations. The scalar one-dimensional version of the operator is analyzed for clarity of presentation. By analogy to the diffusion equation, the operator, including boundary conditions, is assumed to be of the following form:

$$\mathcal{G}(\mathbf{q}) = \mathcal{D}^{(2)}\mathbf{q} + \Sigma(\mathbf{q}), \tag{14}$$

$$\Sigma(\mathbf{q}) = \mathrm{diag}\left(\sigma_{\mathrm{L}}\left(\mathbf{q}_1 - q_{\mathrm{L}}\right), 0, \ldots, 0, \sigma_{\mathrm{R}}\left(\mathbf{q}_N - q_{\mathrm{R}}\right)\right),$$

$$\Sigma : \mathbb{R}^N \to \mathbb{R}^N,\ \sigma_{\mathrm{L}}, \sigma_{\mathrm{R}}, q_{\mathrm{L}}, q_{\mathrm{R}} \in \mathbb{R}.$$

At a domain boundary, q_{L} and q_{R} are pseudo boundary conditions. At block interfaces, they are the flow values at the same point in physical space but corresponding to the adjacent block.

Following some analysis of the Jacobian of the operator, and by comparison with the well-known diffusion operator, it is inferred that the conditions $\sigma_{\mathrm{L}} = d_1$ and $\sigma_{\mathrm{R}} = d_N$ are appropriate conditions for the scalar one-dimensional version of the operator, since these conditions ensure invertibility of $\nabla\mathcal{G}(\mathbf{q})$ and give a consistent representation of the operator on different grid topologies. The extension to three-dimensional vector-valued systems is accomplished by similarly constructing the SATs in each spatial coordinate direction and for each equation. In this paper, the pseudo boundary conditions q_{L} and q_{R} are taken as far-field conditions at all domain boundaries so that the vector containing all far-field values is a solution to $\mathcal{G}(\mathbf{q}) = \mathbf{0}$.

7. Predictor-Corrector Algorithm

The objective of the predictor phase is to obtain a suitable starting guess for the $k+1$st sub-problem using the estimated solution at the k-th sub-problem, a trajectory, and a distance (step-length) to travel along that trajectory. An Euler predictor update at the kth step is given by

$$\mathbf{u}_{k+1}^{(0)} = \mathbf{u}_k^{(p_k)} + h_k \mathbf{d}_k, \tag{15}$$

$$h_k \in \mathbb{R}_+,\ \mathbf{u}_k \in \mathbb{R}^{N+1},\ \mathbf{d}_k \in \mathbb{R}^{N+1},$$

where $\mathbf{u}_k \in \mathbb{R}^{N+1}$, $\mathbf{u}_k = [\mathbf{q}_k; \lambda_k]$, $h_k \in \mathbb{R}$ is the step-length, $\mathbf{d}_k \in \mathbb{R}^{N+1}$ is the step direction, and $p_k \in \mathbb{Z}$ is the number of iterations required to converge the k-th sub-problem.

In this study, the step direction $\mathbf{d}_k$ is taken as an inexact estimate of the tangent vector. As in Brown and Zingg [3], an expression for the tangent is given by

$$t = \frac{\tau}{\|\tau\|},\ \ \tau = \begin{pmatrix} \mathbf{z} \\ -1 \end{pmatrix},\ \ \mathbf{z} = \left[\nabla_{\mathbf{q}} \mathcal{H}(\mathbf{q}, \lambda)\right]^{-1} \frac{\partial}{\partial \lambda} \mathcal{H}(\mathbf{q}, \lambda), \tag{16}$$

$$t \in \mathbb{R}^{N+1},\ \ \mathbf{z} \in \mathbb{R}^{N},\ \ \tau \in \mathbb{R}^{N+1},$$

where $\frac{\partial}{\partial \lambda} \mathcal{H}(\mathbf{q}, \lambda) = \mathcal{G}(\mathbf{q}) - \mathcal{R}(\mathbf{q})$ for convex homotopy.

The objective of the corrector phase is to solve the nonlinear sub-problem at λ_k. The sub-problem $\mathcal{H}(\mathbf{q}_k, \lambda_k) = \mathbf{0}$ is solved inexactly using the inexact Newton method. Newton iterations are performed until the relative residual $\left\|\mathcal{H}\left(\mathbf{q}_k^{(n)}, \lambda_k\right)\right\| / \left\|\mathcal{H}\left(\mathbf{q}_k^{(0)}, \lambda_k\right)\right\|$ is reduced below some user-specified tolerance $\mu_k \in \mathbb{R}$. Typically, $\mu_k \in [0.1, 0.5]$.

In this paper, step-length adaptation is performed by determining a factor $f \in \mathbb{R}$ at the end of each corrector phase by which the previous step-length h_{k-1} is divided. Once f is determined, the condition $f_{\min} \leq f \leq f_{\max}$ is enforced. In addition, the update to $\Delta\lambda$ is forced to satisfy $|\Delta\lambda|_{\min} \leq |\Delta\lambda| \leq |\Delta\lambda|_{\max}$. The step-length is adapted using the method of asymptotic expansions [1, 13] using two criteria: the distance $\delta \in \mathbb{R}$ between the predictor point and the corrector point and the angle $\phi \in \mathbb{R}$ between consecutive tangent vectors. Explicit expressions for these quantities are given by

$$\delta = \left\|\mathbf{q}_k^{(p_k)} - \mathbf{q}_k^{(0)}\right\|, \tag{17}$$

$$\phi = \arccos\left(t_k^{(p_k)} \cdot t_{k-1}^{(p_{k-1})}\right), \tag{18}$$

and the factor f is calculated as

$$f = \max\left\{\sqrt{\frac{\delta}{\tilde{\delta}}}, \frac{\phi}{\tilde{\phi}}\right\}. \tag{19}$$

A pseudo-code of the predictor-corrector algorithm is provided as Algorithm 2.

8. Monolithic Homotopy Continuation Algorithm

8.1. Dynamic Inversion Principle

The motivation for the monolithic homotopy (MH) continuation algorithm comes from the dynamic inversion principle. Dynamic inversion [14, 15, 16] is a process that was developed for

Algorithm 2: Predictor-corrector (PC) algorithm.

Set $\lambda = 1$ and solve $\mathcal{G}(\mathbf{q}) = \mathbf{0}$ if necessary
Predictor iterations: while $\lambda > 0$ **do**
  Corrector iterations: while $\|\mathcal{H}\|$ *is above some (relative) tolerance* **do**
    Get $\mathcal{H}$, $\|\mathcal{H}\|$, and $\frac{\partial}{\partial\lambda}\mathcal{H}$
    Form and factor the preconditioner based on the matrix $\nabla\mathcal{H}$
    Solve the linear system $\nabla\mathcal{H}\Delta\mathbf{q} = -\mathcal{H}$ and take a Newton step
  end
  Calculate the predictor direction
  Calculate the step-length
  Take a predictor step, updating λ and $\mathbf{q}$
end

tracking implicitly-defined trajectories to which the inverse mapping is not analytically available and is parameter-dependent. This method was developed for applications in the field of robotics, where the parameter is time.

Consider a regular Lipshitz-continuous parameter-dependent system of equations

$$\mathcal{F}(\mathbf{q}(\lambda), \lambda) = \mathbf{0} \tag{20}$$

$\mathcal{F} : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$, $(\mathbf{q}(\lambda), \lambda) \mapsto \mathcal{F}(\mathbf{q}(\lambda), \lambda)$ given some parameter $\lambda \in \mathbb{R}$ and assume that $\mathbf{q}_s(\lambda)$ is the curve which satisfies equation (20). If $\nabla_{\mathbf{q}}\mathcal{F}(\mathbf{q}_s(\lambda), \lambda)$ is positive-definite then the ODE

$$\dot{\mathbf{q}}(\lambda) + \mathcal{F}(\mathbf{q}(\lambda), \lambda) = \mathbf{0} \tag{21}$$

converges asymptotically to the curve $\mathbf{q}_s(\lambda)$ if $\mathbf{q}(\lambda)$ is sufficiently close to $\mathbf{q}_s(\lambda)$. The principle behind the dynamic inverse is that if the solution to the ODE (21) does not converge to $\mathbf{q}_s(\lambda)$ then it may be possible to introduce a so-called dynamic inverse $\mathcal{F}^* : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$, $(\mathbf{q}(\lambda), \lambda) \mapsto \mathcal{F}^*(\mathbf{q}(\lambda), \lambda)$ such that the solution to the ODE

$$\dot{\mathbf{q}}(\lambda) + \mathcal{F}^*\mathcal{F}(\mathbf{q}(\lambda), \lambda) = \mathbf{0} \tag{22}$$

is locally asymptotically convergent to $\mathbf{q}_s(\lambda)$. In the case where the ODE (21) is convergent, it may be possible to construct a dynamic inverse $\mathcal{F}^*(\mathbf{q}(\lambda), \lambda)$ such that the modified ODE (22) still converges but with a higher convergence rate. Formal definitions of the forward and reverse-mode dynamic inverses, in the context of convex homotopy, are given by

Definition 1. *Let $\mathbf{q}_s(\lambda)$ be a regular homotopy defined implicitly by $\mathcal{H}(\mathbf{q}, \lambda) = \mathbf{0}$, $\mathcal{H} : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$. Let $\mathcal{H}^* : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$ be continuous in λ and Lipschitz continuous on the ball $\mathcal{B}_r = \left\{\Delta\mathbf{q} \in \mathbb{R}^N \middle| \, \|\Delta\mathbf{q}\| \leq r\right\}$, $r > 0$. Then $\mathcal{H}^*$ is called a* forward dynamic inverse *of $\mathcal{H}$ on $\mathcal{B}_r$ if there exists fixed $\beta \in \mathbb{R}$, $0 < \beta < \infty$, such that*

$$\Delta\mathbf{q}^T\mathcal{H}^*(\mathcal{H}(\mathbf{q}_s(\lambda) + \Delta\mathbf{q}, \lambda), \lambda) \geq \beta\|\Delta\mathbf{q}\|^2 \tag{23}$$

for all $\Delta\mathbf{q} \in \mathcal{B}_r$. Similarly, $\mathcal{H}^*$ is called a reverse mode dynamic inverse *of $\mathcal{H}$ on $\mathcal{B}_r$ if there exists fixed $\beta \in \mathbb{R}$, $0 < \beta < \infty$, such that*

$$\Delta\mathbf{q}^T\mathcal{H}^*(\mathcal{H}(\mathbf{q}_s(\lambda) + \Delta\mathbf{q}, \lambda), \lambda) \leq -\beta\|\Delta\mathbf{q}\|^2 \tag{24}$$

for all $\Delta\mathbf{q} \in \mathcal{B}_r$.

Remark 1. *If $\mathcal{H}^*$ is a (forward or reverse mode) dynamic inverse of $\mathcal{H}$ with constant β, then for any $\gamma \in \mathbb{R}$, $\gamma > 0$, $\gamma\mathcal{H}^*$ is a (forward or reverse mode) dynamic inverse of $\mathcal{H}$ with constant $\gamma\beta$.*

Remark 2. *If $\mathcal{H}^*$ is a forward dynamic inverse of $\mathcal{H}$, then $-\mathcal{H}^*$ is a reverse mode dynamic inverse of $\mathcal{H}$.*

The dynamic inverse $\mathcal{H}^*$ of Definition 1 is a local state- and parameter-dependent approximation to the inverse of the nonlinear system of equations $\mathcal{H}(\mathbf{q}, \lambda)$. Theorem 1 to follow presents the predictor portion $\mathcal{E}$, which is a local state- and parameter-dependent approximation to $\dot{\mathbf{q}}_s(\lambda)$, the rate of change of actual curve values $\mathbf{q}_s$ with respect to the parameter λ.

Theorem 1. *Let $\mathbf{q}_s(\lambda)$ be a regular homotopy defined implicitly by $\mathcal{H}(\mathbf{q}, \lambda) = \mathbf{0}$. Assume that $\mathcal{H}^* : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$; $(\mathbf{w}, \lambda) \mapsto \mathcal{H}^*(\mathbf{w}, \lambda)$ is a reverse mode dynamic inverse of $\mathcal{H}(\mathbf{q}, \lambda)$ on $\mathcal{B}_r = \left\{\Delta\mathbf{q} \in \mathbb{R}^N | \ \|\Delta\mathbf{q}\| \leq r\right\}$, $r > 0$, $0 < \beta < \infty$. Let $\mathcal{E} : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$; $(\mathbf{q}, \lambda) \mapsto \mathcal{E}(\mathbf{q}, \lambda)$ be locally Lipschitz in $\mathbf{q}$ and piecewise continuous in λ. Assume that for some fixed $\omega \in (0, \infty)$, $\mathcal{E}(\mathbf{q}, \lambda)$ satisfies*

$$-\frac{1}{2}\omega \|\Delta\mathbf{q}\|^2 \leq \Delta\mathbf{q}^T \left[\mathcal{E}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda) + \dot{\mathbf{q}}_s(\lambda)\right] \leq \frac{1}{2}\omega \|\Delta\mathbf{q}\|^2 \tag{25}$$

for all $\Delta\mathbf{q} \in \mathcal{B}_r$. Let $\mathbf{q}'_s(\lambda)$ denote the solution to the system

$$-\dot{\mathbf{q}} = \gamma\mathcal{H}^*\left(\mathcal{H}(\mathbf{q}, \lambda), \lambda\right) + \mathcal{E}(\mathbf{q}, \lambda), \tag{26}$$

where $\gamma \in \mathbb{R}$, $\gamma > 0$ (see Remark 1). Consider now some $\lambda_k \in \mathbb{R}$ such that

$$\mathbf{q}_s(\lambda_k) - \mathbf{q}'_s(\lambda_k) \in \mathcal{B}_r. \tag{27}$$

Then

$$\left\|\mathbf{q}'_s(\lambda) - \mathbf{q}_s(\lambda)\right\| \leq \left\|\mathbf{q}'_s(\lambda_k) - \mathbf{q}_s(\lambda_k)\right\| e^{-(\gamma\beta-\omega)|\lambda_k-\lambda|} \tag{28}$$

for all $\lambda < \lambda_k$.

Proof.
The proof is similar to that of Getz [14], pp. 25-26. Let $\mathbf{z}(\lambda) = \mathbf{q}'_s(\lambda) - \mathbf{q}_s(\lambda)$, $\mathbf{z} \in \mathbb{R}^N$. Let $V(\mathbf{z}(\lambda)) = \frac{1}{2}\|\mathbf{z}(\lambda)\|^2$, $V : \mathbb{R}^N \to \mathbb{R}$. Then we have:

$$\begin{aligned}
\dot{\mathbf{z}} &= \dot{\mathbf{q}}'_s - \dot{\mathbf{q}}_s \\
&= -\gamma\mathcal{H}^*\left(\mathcal{H}(\mathbf{q}'_s, \lambda)\lambda\right) - \mathcal{E}(\mathbf{q}'_s, \lambda) - \dot{\mathbf{q}}_s \\
&= -\gamma\mathcal{H}^*\left(\mathcal{H}(\mathbf{q}_s + \mathbf{z}, \lambda)\right) - \mathcal{E}(\mathbf{q}_s + \mathbf{z}, \lambda) - \dot{\mathbf{q}}_s.
\end{aligned} \tag{29}$$

Differentiating $V(\mathbf{z}(\lambda))$ with respect to $-\lambda$ gives:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}(-\lambda)} V(\mathbf{z}(\lambda)) &= -\mathbf{z}^T\dot{\mathbf{z}} \\
&= \mathbf{z}^T\left[\gamma\mathcal{H}^*\left(\mathcal{H}(\mathbf{q}_s + \mathbf{z}, \lambda)\right) + \mathcal{E}(\mathbf{q}_s + \mathbf{z}, \lambda) + \dot{\mathbf{q}}_s\right] \\
&\leq -\gamma\beta\|\mathbf{z}\|^2 + \omega\|\mathbf{z}\|^2.
\end{aligned} \tag{30}$$

So, for $\mathbf{z} \in \mathcal{B}_r$,

$$\frac{\mathrm{d}}{\mathrm{d}(-\lambda)} V(\mathbf{z}(\lambda)) \leq -2(\gamma\beta - \omega) V(\mathbf{z}(\lambda)). \tag{31}$$

For an initial condition $-\lambda = -\lambda_k$, then by the Comparison Theorem (see, for example, Hartman [17]), we obtain:

$$V(\mathbf{z}(\lambda)) \leq V(\mathbf{z}(\lambda_k))\, e^{-2(\gamma\beta-\omega)(\lambda_k-\lambda)}, \quad \lambda < \lambda_k. \tag{32}$$

Taking the square root of both sides of equation (32) returns equation (28) as required. □

The ODE (26) converges to the homotopy curve asymptotically as long as $\gamma\beta > \omega$ and with an upper bound on the convergence rate as given by equation (28). To achieve a high convergence rate, it is desired that $\gamma\beta - \omega$ should be as large as possible. An explicit value for the parameter β is generally unavailable but depends on the definiteness of $\mathcal{H}^*\mathcal{H}(\mathbf{q}, \lambda)$. The parameter ω is a measure of the quality of the predictor term $\mathcal{E}$ and is 0 if $\mathcal{E}$ is exact, which is possible in the continuous case if $\mathcal{E}$ is set analytically to $-\dot{\mathbf{q}}_s$, as can be verified by inspection of equation (25). The final parameter γ is an additional degree of freedom introduced in the definition of the dynamic inverse and can be chosen by the user. The relationship between γ and $|\Delta\lambda|$ is covered in Section 8.5.

8.2. Construction of $\mathcal{E}$

The quantity $\dot{\mathbf{q}}_s(\lambda)$ is the tangent vector with λ-parametrization and can easily be derived by directly differentiating $\mathcal{H}(\mathbf{q}_s(\lambda), \lambda) = \mathbf{0}$ and rearranging:

$$\dot{\mathbf{q}}_s(\lambda) = -\left[\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_s(\lambda), \lambda)\right]^{-1} \frac{\partial}{\partial\lambda}\mathcal{H}(\mathbf{q}_s(\lambda), \lambda). \tag{33}$$

A suitable choice for the operator $\mathcal{E}$ can be obtained by setting $\mathcal{E}$ to $-\dot{\mathbf{q}}_s$, as per equation (25):

$$\mathcal{E}(\mathbf{q}, \lambda) = \left[\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}'_s(\lambda), \lambda)\right]^{-1} \frac{\partial}{\partial\lambda}\mathcal{H}(\mathbf{q}'_s(\lambda), \lambda). \tag{34}$$

For the special case of convex homotopy, the specific form of this equation is

$$\mathcal{E}(\mathbf{q}, \lambda) = \left[\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}'_s(\lambda), \lambda)\right]^{-1} \left[\mathcal{G}(\mathbf{q}'_s) - \mathcal{R}(\mathbf{q}'_s)\right]. \tag{35}$$

8.3. Construction of $\mathcal{H}^*$

As mentioned previously, a stable ODE can be constructed for the homotopies studied in this paper by taking the reverse mode dynamic inverse $\mathcal{H}^*$ as the identity operator. However, the ODE that would result from equation (26) would resemble an explicit time-marching method and a high convergence rate would not be expected.

A more efficient alternative might be to use the inverse Jacobian evaluated near the curve, which is a forward dynamic inverse [15]. The most practical location to evaluate the inverse Jacobian, of course, is at the current point generated by the algorithm. It is shown in Section 8.5 that this results in a Newton-like update. While using an inverse Jacobian as the dynamic inverse requires for a linear system of equations to be solved, this contributes no additional cost when constructing $\mathcal{E}$ according to equation (34) because this term also contains an inverse Jacobian and the two linear solves can be combined into one in the MH update expression.

We now show, following the approach of Getz and Marsden [15], that the inverse Jacobian evaluated near the curve is a dynamic inverse of $\mathcal{H}$. Assume that the continuation algorithm

has arrived at some point $\mathbf{q}_s + \Delta\mathbf{q}$, $\Delta\mathbf{q} \in \mathcal{B}_r \subset \mathbb{R}^N$, $r > 0$. The residual $\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda)$ can be represented locally by taking the Taylor expansion at $\mathbf{q}_s$:

$$\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda) = \mathcal{H}(\mathbf{q}_s, \lambda) + \nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_s, \lambda)\Delta\mathbf{q} + O\left(\|\Delta\mathbf{q}\|^2\right). \tag{36}$$

Similarly expanding $\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}, \lambda)$ at $\mathbf{q}_s$ gives

$$\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_s, \lambda) = \nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda) + O(\|\Delta\mathbf{q}\|). \tag{37}$$

Using equation (37) and $\mathcal{H}(\mathbf{q}_s, \lambda) = \mathbf{0}$, equation (36) becomes

$$\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda) = \nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda)\Delta\mathbf{q} + O\left(\|\Delta\mathbf{q}\|^2\right). \tag{38}$$

Let $\mathcal{H}^* = \left[\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda)\right]^{-1}$ be a candidate for a forward dynamic inverse of $\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda)$. If it is a forward dynamic inverse, it must satisfy Definition 1. This is verified by applying $\Delta\mathbf{q}^T\mathcal{H}^*$ to both sides of equation (38):

$$\begin{aligned}\Delta\mathbf{q}^T\mathcal{H}^*\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda) &= \Delta\mathbf{q}^T\mathcal{H}^*\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda)\Delta\mathbf{q} + \Delta\mathbf{q}^T\mathcal{H}^*O\left(\left\|\Delta\mathbf{q}^2\right\|\right) \\ &= \|\Delta\mathbf{q}\|^2 + O\left(\|\Delta\mathbf{q}\|^3\right).\end{aligned} \tag{39}$$

The contribution of the $O\left(\|\Delta\mathbf{q}\|^3\right)$ terms can be positive or negative. In either case, for sufficiently small $r > 0$, there exists fixed $0 < \beta < 1$ such that these combined terms are upper-bounded by $(1-\beta)\|\Delta\mathbf{q}\|^2$. Thus, for sufficiently small $r > 0$, there exists fixed $0 < \beta < 1$ such that

$$\Delta\mathbf{q}^T\mathcal{H}^*\mathcal{H}(\mathbf{q}_s + \Delta\mathbf{q}, \lambda) \geq \|\Delta\mathbf{q}\|^2 - (1-\beta)\|\Delta\mathbf{q}\|^2 = \beta\|\Delta\mathbf{q}\|^2 \tag{40}$$

for all $\Delta\mathbf{q} \in \mathcal{B}_r$. Hence, the inverse Jacobian evaluated at $\mathbf{q}_s + \Delta\mathbf{q}$ is a forward dynamic inverse. Since a reverse-mode dynamic inverse is what is needed in the context of homotopy, then, by Remark 2, a suitable expression for $\mathcal{H}^*$ is

$$\mathcal{H}^* = -\left[\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}, \lambda)\right]^{-1}. \tag{41}$$

8.4. Numerical Implementation

Substituting equations (41) and (35) into (26) gives the following ODE:

$$-\dot{\mathbf{q}}(\lambda) = \left[\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}, \lambda)\right]^{-1}\left[-\gamma\mathcal{H}(\mathbf{q}, \lambda) + \mathcal{G}(\mathbf{q}) - \mathcal{R}(\mathbf{q})\right], \tag{42}$$

which can then be integrated numerically to obtain an update formula for the state. For example, using an Euler integration scheme, we obtain:

$$\mathbf{q}_{k+1} = \mathbf{q}_k + (\lambda_{k+1} - \lambda_k)\left[\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}_k, \lambda_k)\right]^{-1}\left[\gamma_k\mathcal{H}(\mathbf{q}_k, \lambda_k) - \mathcal{G}(\mathbf{q}_k) + \mathcal{R}(\mathbf{q}_k)\right], \tag{43}$$

where selection of $\Delta\lambda_k$ and γ are discussed in Section 8.5. Though not of direct import to the analysis in this paper, it may be worth noting that equation (42) can equivalently be written as

$$\frac{\mathrm{d}}{\mathrm{d}\lambda}\mathcal{H}(\mathbf{q}, \lambda) = \gamma\mathcal{H}(\mathbf{q}, \lambda). \tag{44}$$

Algorithm 3: Monolithic homotopy (MH) continuation algorithm

Set $\lambda = 1$ and solve $\mathcal{G}(\mathbf{q}) = \mathbf{0}$ if necessary
while $\lambda > 0$ **do**
  Get γ, $\mathcal{H}$, $\|\mathcal{H}\|$, and $-\gamma\mathcal{H} + \mathcal{G} - \mathcal{R}$
  Form and factor the preconditioner approximating the matrix $\nabla\mathcal{H}$
  Solve the linear system $\nabla_{\mathbf{q}}\mathcal{H}\Delta\mathbf{q} = [-\gamma\mathcal{H} + \mathcal{G} - \mathcal{R}]$ for $\Delta\mathbf{q}$
  Determine $\Delta\lambda$ from step-length adaptation (Algorithm 4)
  Update λ and $\mathbf{q}$
end

Ideally, the monolithic homotopy algorithm is initiated at $\lambda_0 = 1$ with an exact solution for $\mathcal{G}(\mathbf{q}) = \mathbf{0}$, thus ensuring that condition (27) is met for $\lambda = 1$. Taking small enough $|\Delta\lambda|$ should ensure that $-\left[\nabla_{\mathbf{q}}\mathcal{H}(\mathbf{q}, \lambda)\right]^{-1}$ evaluated at the current value of $\mathbf{q}$ remains a dynamic inverse of $\mathcal{H}(\mathbf{q}, \lambda)$ and so there should always exist small enough $|\Delta\lambda|$ such that the MH algorithm is stable.

Once traversing is complete and λ is reduced to 0, the inexact Newton phase is initiated to solve the system $\mathcal{R}(\mathbf{q}) = \mathbf{0}$. A pseudo-code for the MH algorithm is provided as Algorithm 3.

8.5. Predictor-Corrector Analogue and Selection of γ_k and $\Delta\lambda_k$

For notational convenience, introduce $t_{\mathbf{q}} \in \mathbb{R}^N$, $t_\lambda \in \mathbb{R}$, such the $t = \left[t_{\mathbf{q}};\, t_\lambda\right]$. Since an Euler update for the PC method is given by

$$\begin{pmatrix} \mathbf{q} \\ \lambda \end{pmatrix}_{k+1}^{(0)} = \begin{pmatrix} \mathbf{q} \\ \lambda \end{pmatrix}_{k}^{(p_k)} + h_k \begin{pmatrix} t_{\mathbf{q}} \\ t_\lambda \end{pmatrix}_k, \tag{45}$$

where $h_k \in \mathbb{R}$ is the step-length, then h_k and $\Delta\lambda_k$ are correlated by

$$\Delta\lambda_k = -h_k / \|\tau_k\|. \tag{46}$$

Equation (46) can be used with equation (16) to develop an equation for $\Delta\lambda_k\mathbf{z}_k$:

$$\Delta\lambda_k\mathbf{z}_k = \Delta\lambda_k \|\tau_k\| t_{\mathbf{q}} = -h_k t_{\mathbf{q}}, \tag{47}$$

which can then be used to rewrite equation (43) as a combination of more familiar quantities:

$$\begin{aligned} \Delta\mathbf{q}_k &= \gamma_k\Delta\lambda_k \left[\nabla\mathcal{H}(\mathbf{q}_k, \lambda_k)\right]^{-1} \mathcal{H}(\mathbf{q}_k, \lambda_k) - \Delta\lambda_k\mathbf{z}_k \\ &= \gamma_k\Delta\lambda_k \left[\nabla\mathcal{H}(\mathbf{q}_k, \lambda_k)\right]^{-1} \mathcal{H}(\mathbf{q}_k, \lambda_k) + h_k t_{\mathbf{q}}. \end{aligned} \tag{48}$$

Numerically, the monolithic homotopy update formula is applied in the form of equation (43). However, writing this equation in the form of equation (48) reveals that each update iteration with $\gamma_k = 1/|\Delta\lambda_k|$ is analogous to a combination of a Newton iteration and a predictor step, both evaluated at the same state. When $|\Delta\lambda_k|$ is small, the corrector portion of the update will be emphasized and the algorithm will be more robust, but more steps will be required for traversing.

Since it may be desirable to adjust $\Delta\lambda_k$ after the linear solve, the following formula can be used for γ_k:

$$\gamma_k = \frac{1}{|\Delta\lambda_k|^*}, \tag{49}$$

where $|\Delta\lambda_k|^*$ is an approximation to $|\Delta\lambda_k|$. A suitable choice might be $|\Delta\lambda_k|^* = |\Delta\lambda_{k-1}|$, $k \geq 1$. This is the approach adopted in this paper.

8.6. Step-length Adaptation

Though the MH method allows for more control over traversing, less information is available to do so, especially since it is not possible to distinguish between the predictor and corrector portion of the update. The distance to the curve is not available, and any change in orientation of the corrector portion of the update is not particularly meaningful, so that the angle between consecutive updates is not a useful metric.

The only useful information comes from the update itself. If the norm of the state update $||\Delta\mathbf{q}_k||$ is large, then $|\Delta\lambda_k|$ should be reduced in order to attempt to maintain a consistent Δs. The proposed update formula is based on keeping $||\Delta\mathbf{q}_k||$ constant, calibrated from the first iteration. Thus, the user chooses an initial value of $|\Delta\lambda|$ to calibrate the step-length adaptation rather than choosing a target value of $||\Delta\mathbf{q}||$.

Denote the target value of $||\Delta\mathbf{q}||$ by $||\Delta\mathbf{q}||_{\text{tar}}$. This quantity can be set according to

$$||\Delta\mathbf{q}||_{\text{tar}} = ||\dot{\mathbf{q}}_0||\,|\Delta\lambda_0|\,, \tag{50}$$

which can be used to calculate the numerical integration step

$$\Delta\lambda_k = -\frac{||\Delta\mathbf{q}||_{\text{tar}}}{||\dot{\mathbf{q}}||}. \tag{51}$$

The step size $|\Delta\lambda|$ is also restricted by upper and lower bounds:

$$\min\left\{|\Delta\lambda|_{\min}\,, f_{\min}\,|\Delta\lambda_{k-1}|\right\} \leq |\Delta\lambda_k| \leq \max\left\{|\Delta\lambda|_{\max}\,, f_{\max}\,|\Delta\lambda_{k-1}|\right\}, \tag{52}$$

where $|\Delta\lambda|_{\min}\,, f_{\min}, |\Delta\lambda|_{\max}\,, f_{\max} \in \mathbb{R}$ can be user-defined. For the current study, $f_{\min} = 1/3$ and $f_{\max} = 2$; $|\Delta\lambda|_{\min}$ and $|\Delta\lambda|_{\max}$ are adjusted according to the initial step size $|\Delta\lambda_0|$ but are generally not chosen to be overly restrictive. In addition, a maximum step size can be imposed at the final step to attempt to improve the quality of the initial guess for the inexact Newton phase at $\lambda = 0$. A detailed pseudo-code of the step-length adaptation is provided in Algorithm 4, including many different checks that are applied to $\Delta\lambda$.

Surrogate curves generated from the values of the lift coefficient C_L calculated along the homotopy curve can be used as a semi-quantitative assessment of how accurately the curve is being traced. The 'exact' values of C_L along the curve are estimated using the predictor-corrector method with a small step size and tight convergence tolerance in the corrector phase.

The C_L values calculated along the MH trajectory using several values of $|\Delta\lambda_0|$ are compared to the surrogate curves generated by the PC algorithm in Figure 1. The case is inviscid flow over the ONERA M6 wing at Mach 0.5 and angle of attack 6° on a 1.9207×10^6 node mesh. We can see from the figure the effects of $|\Delta\lambda_0|$ on the ability of the MH algorithm to trace the curve accurately. The expected trend is observed that larger $|\Delta\lambda_0|$ results in fewer iterations in total but reduces the curve-tracing accuracy. We also see that the number of iterations taken for traversing is not linearly proportional to $|\Delta\lambda_0|$. This effect is due to the active step-length adaptation. The algorithm takes 17 iterations at $|\Delta\lambda_0| = 0.1$, 11 iterations at $|\Delta\lambda_0| = 0.15$, 10 iterations at $|\Delta\lambda_0| = 0.2$, and 9 iterations at $|\Delta\lambda_0| = 0.25$. While all flow solves converged, the apparent loss of accuracy at $|\Delta\lambda_0| = 0.25$ is an indication that robustness may be lost for other cases with this parameter setting. Since there appears to be no major CPU time reduction under this setting, we might recommend, based on this figure, that $|\Delta\lambda_0|$ values in the range of 0.15 to 0.2 are suitable for this case.

Algorithm 4: Detailed step-length adaptation pseudo-code for the MH algorithm

Data: $|\Delta\lambda|_{\min}$, $f_{\min}$, $|\Delta\lambda|_{\max}$, $f_{\max}$, $\Delta\lambda_{\text{prev}}$, $|\Delta\lambda|_{\max,\text{final}}$, k
Result: $\Delta\lambda$
if $k = 0$ **then**
  Initialize $\Delta\lambda$ from the input value
  $||\Delta\mathbf{q}||_{\text{tar}} \leftarrow ||\dot{\mathbf{q}}||\,|\Delta\lambda|$
else
  $\Delta\lambda \leftarrow -\,||\dot{\mathbf{q}}|| \,/\, ||\Delta\mathbf{q}||_{\text{tar}}$
  $f \leftarrow |\Delta\lambda| \,/\, \left|\Delta\lambda_{\text{prev}}\right|$
  $f \leftarrow \max\left\{\min\left\{f, f_{\max}\right\}, f_{\min}\right\}$
  $\Delta\lambda \leftarrow f\Delta\lambda_{\text{prev}}$
end
/* Check $|\Delta\lambda|$ against $|\Delta\lambda|_{\min}$ and $|\Delta\lambda|_{\max}$ */
if $|\Delta\lambda| < |\Delta\lambda|_{\min}$ *and* $\lambda > 0$ **then**
  $\Delta\lambda \leftarrow \Delta\lambda_{\min}$
else if $|\Delta\lambda| > |\Delta\lambda|_{\max}$ **then**
  $\Delta\lambda \leftarrow \Delta\lambda_{\max}$
end
/* Additional checks near $\lambda = 0$ */
$\lambda^* \leftarrow \lambda + \Delta\lambda$
$\lambda_{\min} \leftarrow 0.25\lambda$
if $\lambda^* < 0$ **then**
  /* Adjust $\Delta\lambda$ so that the Euler update will give exactly $\lambda = 0$ */
  $\Delta\lambda \leftarrow -\lambda$
  if $\lambda^* > |\Delta\lambda|_{\max,\text{final}}$ **then**
    /* Adjust $\Delta\lambda$ if it is the final step and exceeds $|\Delta\lambda|_{\max,\text{final}}$ */
    $\lambda_{\min} \leftarrow \min\left\{\lambda_{\min}, |\Delta\lambda|_{\max,\text{final}}\right\}$
    $\Delta\lambda \leftarrow \lambda_{\min} - \lambda$
  end
else if $\lambda^* < \lambda_{\min}$ **then**
  /* This logic exists to even out the final steps */
  $\Delta\lambda \leftarrow \lambda_{\min} - \lambda$
end

9. Convergence Investigation

Convergence of the algorithm is proved on condition that equation (35) is satisfied with strict equality. However, when using the Jacobian-free Newton-Krylov approach, the linear systems are solved inexactly, so the equation is not strictly satisfied. The consequences of solving equation (35) inexactly are studied in this section.

9.1. Inviscid Flow Study

The expected relationship between $|\Delta\lambda|$ and the homotopy residual $||\mathcal{H}(\mathbf{q}, \lambda)||$ is that as $|\Delta\lambda|$ is reduced, the curve should be traced more accurately, so $||\mathcal{H}(\mathbf{q}, \lambda)||$ should become smaller.

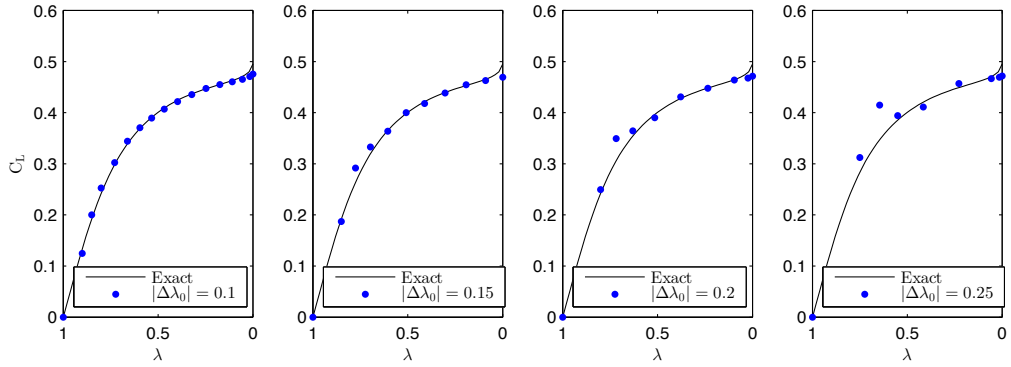


Figure 1: Tracking error history for the MH method, including step-length adaptation, when applied to the inviscid ONERA M6 wing at Mach number 0.5 and angle of attack 6°

However, it was consistently observed that $\|\mathcal{H}(\mathbf{q}, \lambda)\|$ can grow to larger values during traversing for cases where smaller step sizes were taken.

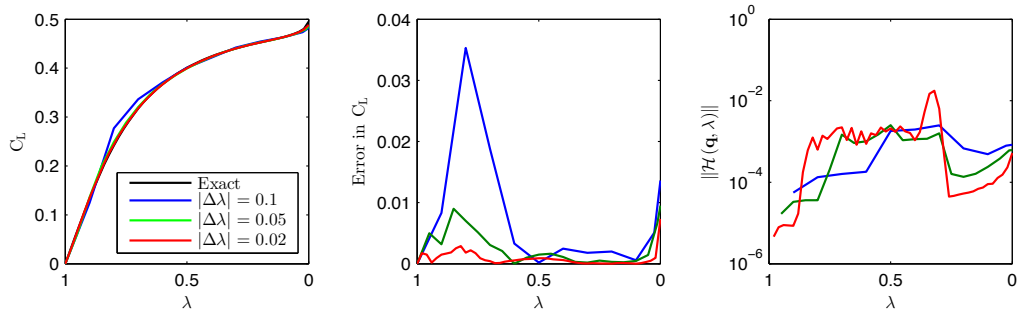
This behaviour is investigated numerically for inviscid flow over the ONERA M6 wing at Mach 0.5 and angle of attack of 3° by applying the MH algorithm several times with several values of $|\Delta\lambda|$ and with the step-length adaptation inactive. The mesh has an H-C topology with about 1.88 million nodes divided evenly into 32 blocks. It is observed from Figure 2 that for smaller $|\Delta\lambda|$, the tracking error is consistently lower, but the homotopy residual norm $\|\mathcal{H}(\mathbf{q}, \lambda)\|$ is generally higher. This behaviour was found to be linked to the linear solver tolerance. Figure 2b shows the same study performed using a linear solver tolerance of $\tau_l = 0.001$, in which $\|\mathcal{H}\|$ and the error in C_L are both consistently lower for smaller $|\Delta\lambda|$.

The observed inverse relationship between $\|\mathcal{H}(\mathbf{q}, \lambda)\|$ and $|\Delta\lambda|$ is an effect of evaluating discrete residuals on data which contains high-frequency error and is indicative of the onset of instability in the MH iterations caused by inexactly forming $\mathcal{E}$. However, no performance degradation is apparent for these inviscid cases.

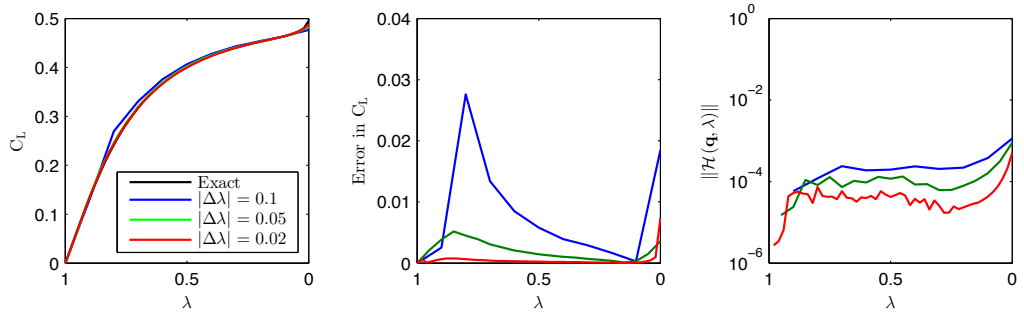
9.2. Turbulent Flow Study

The second stability study is investigated for the case of three-dimensional transonic flow over the ONERA M6 wing at Reynolds number 1×10^7, Mach number 0.8, and angle of attack 3°. The grid consists of 3.68×10^7 nodes divided into 1024 blocks with a minimum off-wall distance of 8.00×10^{-7} mean chord units. The step size $|\Delta\lambda|$ was kept constant at a very small value of 0.005 in order to attempt to isolate the effects of certain solver parameters in the continuation phase. The parameters studied were the use of approximate matrix-vector products versus finite-difference matrix-vector products, the tolerance used for the linear solver, and different scalings for the linear system. The scalings employed are the geometric scaling, which is consistent with Osusky and Zingg [29], and the row/column normalization scaling, which is performed by normalizing the rows and then subsequently normalizing the columns. The parameter combinations investigated are presented in Table 1.

The effects of the solver parameters on the tracking error are shown in Figure 3. As with the



(a) $\tau_1 = 0.01$



(b) $\tau_1 = 0.001$

Figure 2: Trajectory of the MH algorithm at Mach number 0.5 and angle of attack 6° with $\tau_1 = 0.001$

Case	Linear solver tolerance	Matrix-vector product	Scaling
F2	10^{-2}	finite-difference	Geometric
F3	10^{-3}	finite-difference	Geometric
A2	10^{-2}	approximate	Geometric
A3	10^{-3}	approximate	Geometric
RC-F2	10^{-2}	finite-difference	Row/column normalization

Table 1: List of studies performed with the MH method with constant $|\Delta\lambda|$ when solving the RANS-SA equations on the ONERA M6 wing at Mach 0.8, angle of attack 3°, and Reynolds number 1×10^7

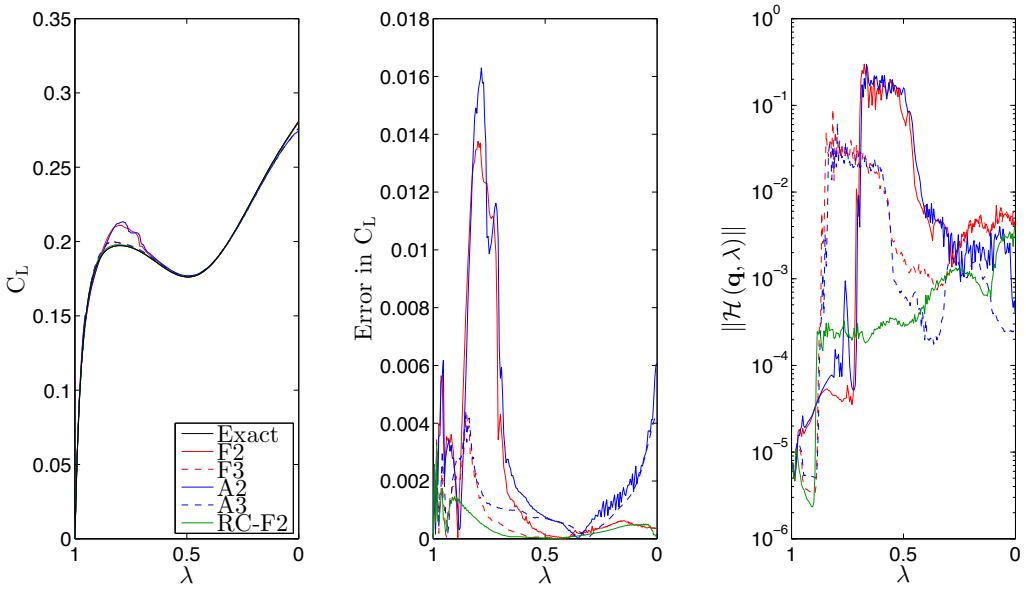


Figure 3: Tracking error history for the MH algorithm with constant $|\Delta\lambda|$ when applied to the 1024 block turbulent case at Mach 0.8, angle of attack 3°, and Reynolds number 1×10^7

inviscid case, there is a clear correlation between the linear solver tolerance and $\|\mathcal{H}\|$. However, an additional effect is observed here that the tracking error becomes very large in the (approximate) region $0.6 < \lambda < 0.9$ regardless of whether the finite-difference or approximate matrix-vector products were used. This effect was greatly reduced when the linear solver tolerance was reduced to $\tau_l = 0.001$. This is an important observation, since such large deviation from the curve can lead to non-convergence of the algorithm.

Some important performance differences were observed when forming the matrix-vector products with the finite-difference or approximate matrix-vector products. Though a similar error trend was observed throughout most of the traversing process between Cases F2 and A2, and similarly between Cases F3 and A3, there is significant discrepancy near $\lambda = 0$ where C_L values are predicted much better using the finite-difference matrix-vector products. This can be attributed to the growth in the approximation error resulting from the use of the approximate Jacobian as the contribution from the flow residual becomes increasingly dominant. This observation is important because it affects the quality of the starting point for Newton's method and can significantly affect the success rate in the inexact Newton phase. By comparison, the accuracy of the matrix-vector products is less important for PTC.

10. Performance Studies

Performance studies are conducted to evaluate the performance of the monolithic homotopy continuation algorithm compared to the established homotopy predictor-corrector continuation algorithm and the pseudo-transient continuation algorithm. Test suites are constructed over a

wide range of subsonic and transonic conditions. Since it is not always known which cases will be transonic and which will be subsonic, the second-difference dissipation with the pressure sensor is active for all test cases. Parameters for each test suite are chosen to attempt to match robustness for all algorithms (e.g. a similar number of cases should converge for each algorithm) and then comparing the time to convergence for each algorithm, averaged over all cases that converged for all three algorithms.

The range of test cases included in each test suite often includes several configurations which are expected to be unsteady, and thus the steady state solution, if it exists, is expected to be dynamically unstable. While some researchers have demonstrated interest in obtaining dynamically unstable steady solutions [34], these solutions are non-physical and rarely of engineering value. There are many cases at high Mach number and angle of attack considered in the test suites presented in this section which we have been unable to converge with any of the three algorithms, but, since these cases are presumed to be unstable, we do not consider this to be an issue that needs to be resolved or addressed.

The parameters chosen for the pseudo-transient method are based on our experience and testing and are consistent with those used by previous authors [2, 11, 19, 29]. The parameters chosen for the homotopy methods are similarly based on our experience and testing. The algorithm parameters used for each test suite are listed in Table 2 for all three algorithms. For all algorithms and for all test suites, the entire suite has been executed for several combinations of parameters to ensure that no algorithm is significantly under-represented due to poor parameter selection, though fine-tuning of parameters has been avoided. Fine-tuning has also been avoided by virtue of the fact that a large number of test cases has been included in each test suite.

All timing statistics are reported in TauBench work units[2]. The TauBench code roughly simulates the CPU cost of running the DLR-developed flow solver Tau, which is a three-dimensional hybrid multigrid solver for the RANS equations. A benchmark timing factor is generated by the TauBench codes by specifying a grid size, number of processors, and number of iterative steps. For this study, the TauBench benchmark that was used is a 2.50×10^5 node mesh run in serial with 10 iterative steps, which takes about 9.571s on the SciNet general purpose cluster where the performance studies were carried out.

10.1. Explanation of the Figures Accompanying Each Test Suite

Timing data is shown in the bar-graph figures accompanying each test suite, an example of which is Figure 4. If data for a particular case is missing from the figure then this indicates that that particular case failed to converge. The white portion of the bar-graph labeled INP refers to the inexact Newton phase.

In addition to timing statistics, a sample convergence history is displayed for one case from each test suite. The convergence history is characterized with four plots, an example of which is Figure 5. Each marker on the plots represents a nonlinear iteration. The amount of time taken for each nonlinear iteration depends mainly on the number of Krylov iterations as well as the cost of the matrix-vector products. The dashed vertical lines indicate the switch to the inexact Newton phase, and the solid vertical lines indicate that the flow solve has completed.

The top left plot shows the evolution of the % error in C_L with time for each continuation algorithm based on the final C_L value. Relative error values below 10^{-6}% are not displayed.

[2]DLR Germany, TauBench Version 1.1, IPACS. `http://www.ipacs-benchmark.org`. Accessed: 2014-09-20.

The top right plot shows the evolution of the residual for PTC and the evolution of λ for CHC-PC and CHC-MH. The residual for PTC is shown on the left vertical axis and λ for the CHC algorithms is shown on the right vertical axis. The plot on the bottom left shows the residual history, either $\mathcal{R}(\mathbf{q})$ for PTC or $\mathcal{H}(\mathbf{q}, \lambda)$ for CHC. In the inexact Newton phase, both residuals are $\mathcal{R}(\mathbf{q})$. However, these residuals cannot be directly compared for RANS cases because we have scaled the residual differently for the homotopy and PTC algorithms. The final plot on the bottom right shows a comparison of the residuals of the two homotopy algorithms during traversing.

10.2. Inviscid Flow Cases

The inviscid test cases are three-dimensional flows over the ONERA M6 wing. The performance studies are carried out on two meshes. The first mesh consists of 1.9207×10^6 nodes and has an H-C topology. The nodes are divided evenly into 32 blocks for parallelization on 32 processors. The second mesh is generated from the first by doubling the number of grid points in each direction and also splitting the blocks once in each direction, resulting in eight times more grid nodes and blocks. The parameters used for the pseudo-transient flow solves on the coarser mesh are similar to those obtained by Dias [11] through several parameter studies. For the predictor-corrector method on the original coarser mesh, we have found that it is sufficient to solve the subproblems to a relative tolerance of only 0.5 and that a large initial step size of $|\Delta\lambda_0| = 0.2$ could be used. The same initial step size is used for the monolithic method. A listing of the algorithm parameters used for all three algorithms is provided in Table 2.

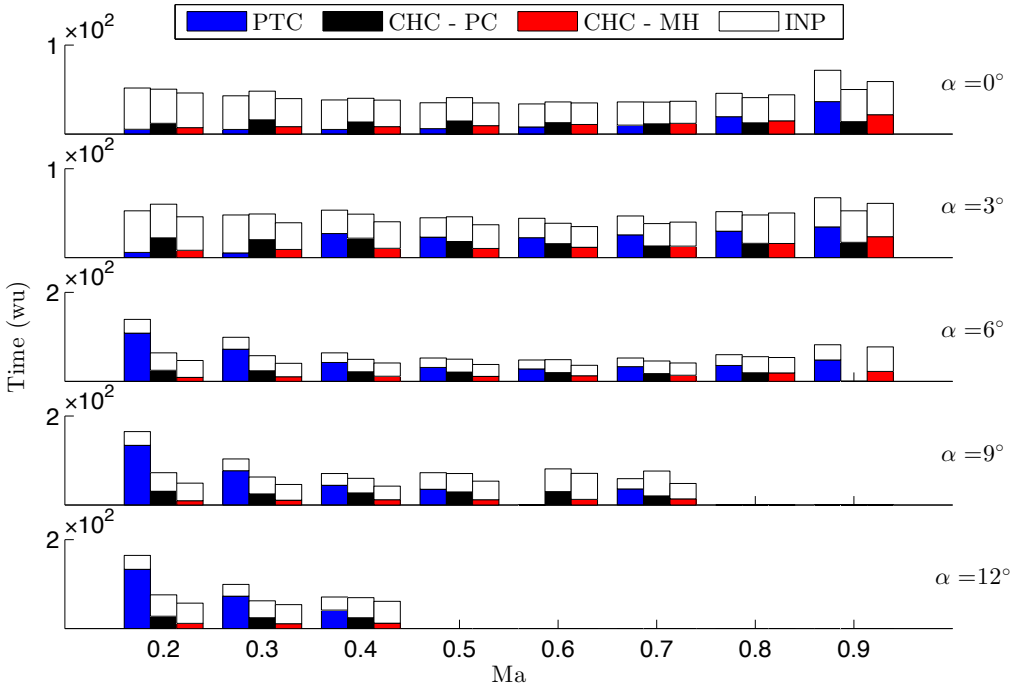
Timing data for the complete flow solve for all three algorithms averaged over all test cases is shown in Figure 4 and Table 2 for the two test suites. On the coarser grid, the convex homotopy with the predictor-corrector method performs only slightly better than the pseudo-transient method with a similar success rate and a 12% reduction in wall time. The monolithic homotopy algorithm has in turn performed slightly better than the predictor-corrector algorithm. On average, on the coarser grid, convergence of the algorithm is achieved in about 23% less wall time with the monolithic homotopy continuation algorithm than with pseudo-transient continuation.

To investigate and compare how the algorithms scale with mesh size, the parameters that have been determined to be suitable for the coarser mesh are used for the same test suite on the finer mesh. For the test suite on the finer mesh, the relative timing data of the homotopy algorithms improved slightly compared to the pseudo-transient algorithm, with the PC algorithm converging in 15% less time and the MH algorithm converging in 34% less time. However, the change in robustness is more significant, with the pseudo transient method converging in only 21/40 cases, down from 32/40 on the coarser mesh, while the PC algorithm converged in 26/40 and the MH algorithm converged in 27/40. Of course these statistics could be improved by parameter tuning, but that is not the purpose of the study.

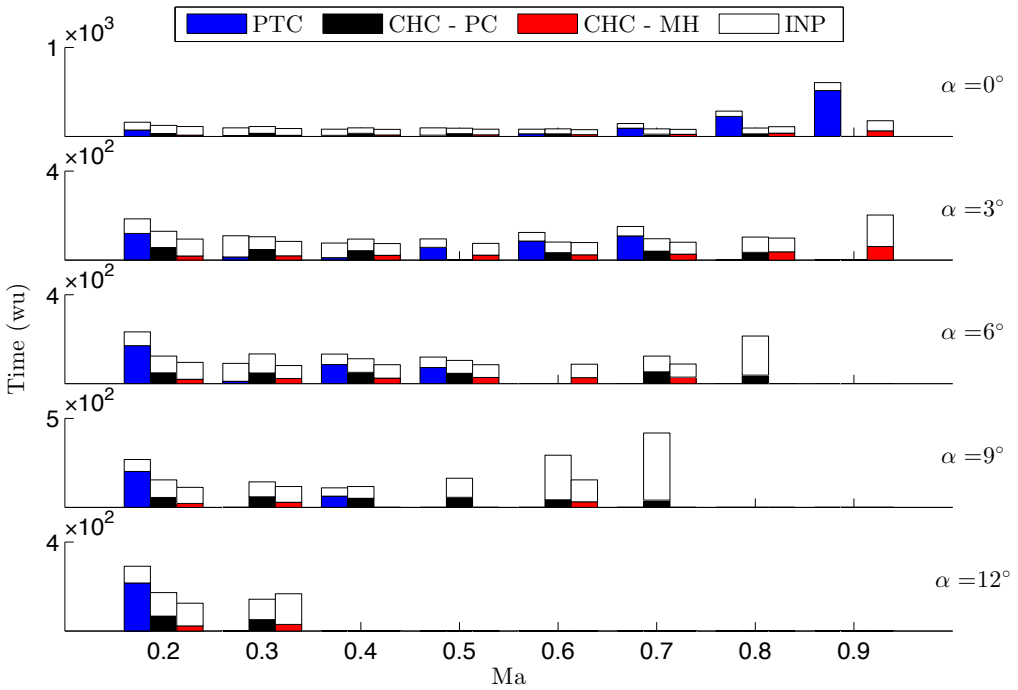
The convergence histories for all three algorithms for a representative case from both test suites are shown in Figure 5. While it seems likely from this figure that the PTC algorithm would converge more efficiently if the switching tolerance for the inexact Newton phase were selected less conservatively, determining the most efficient switching tolerance a priori is impossible and so this over-solving is typical and expected for PTC.

Flow	Geometry	ONERA M6	ONERA M6	NACA 0012	ONERA M6	ONERA M6
	Equations	3D inviscid	3D inviscid	2D RANS-SA	3D RANS-SA	3D RANS-SA
	Ma	0.2, 0.3, . . ., 0.9	0.2, 0.3, . . ., 0.9	0.2, 0.3, . . ., 0.9	0.4, 0.6, 0.75, 0.85	0.4, 0.6, 0.8, 0.9
	α (degrees)	0, 3, 6, 9, 12	0, 3, 6, 9, 12	0, 4, 8, 12	0, 1.5, 3, 4.5	0, 1.5, 3, 4.5
	Re	-		4.00×10^7	1.172×10^7	1.00×10^7
Grid	Topology	H-C	H-C	H	H-H	H-C
	Blocks	32	256	8	192	1024
	Nodes	1.88×10^6	1.53664×10^8	1.92×10^4	2.336064×10^6	3.6799488×10^7
	Off-wall	2.00×10^{-3}	1.09×10^{-3}	7.32×10^{-7}	1.17×10^{-6}	8.00×10^{-7}
Max. wall time (wu)		533	752	31	564	2821
PTC	a	0.01	0.01	0.0001	0.001	0.0001
	b	1.4	1.4	1.35	1.1	1.1
	m	3	3	1	1	1
	ILU(p) fill	0	0	2	2	3
	mat.-vec.	approximate	approximate	approximate	approximate	finite-diff.
	τ_l	0.05	0.05	0.01	0.01	0.01
	τ_{rel}	0.05	0.05	0.0001	0.0001	10^{-5}
	Converged	32/40	21/40	31/32	16/16	13/16
	Rel. Time	1.00	1.00	1.00	1.00	1.00
CHC-PC	$\lvert\Delta\lambda_0\rvert$	0.20	0.20	0.04	0.03	0.03
	ILU(p) fill	1	1	2	2	3
	mat.-vec.	approximate	approximate	approximate	approximate	finite-diff.
	τ_l	0.01	0.01	0.01	0.01	0.01
	τ_s	0.5	0.5	0.5	0.5	0.1
	Converged	32/40	27/40	32/32	16/16	13/16
	Rel. Time	0.88	0.85	0.73	1.14	3.33
CHC-MH	$\lvert\Delta\lambda_0\rvert$	0.20	0.20	0.04	0.02	0.02
	ILU(p) fill	1	1	2	2	3
	mat.-vec.	approximate	approximate	approximate	finite-diff.	finite-diff.
	τ_l	0.01	0.01	0.01	0.01	0.03
	Converged	33/40	27/40	32/32	16/16	14/16
	Rel. Time	0.77	0.66	0.55	0.64	1.00
INP	τ_l	0.01	0.01	0.01	0.01	0.01
	ILU(p) fill	2	2	2	3	3

Table 2: Summary of test case suites and performance statistics for all performance comparisons;
τ_l - relative linear solver tolerance;
τ_{rel} - relative tolerance required for globalization of PTC;
τ_s - relative tolerance to which the subproblems are converged for CHC-PC;
INP - inexact Newton phase;
Rel. Time (relative time) is calculated as follows: the average wall time to complete a flow solve is calculated for each algorithm, considering only the flow solves at the operating conditions that converged for *all three algorithms*; this average wall time is then divided by the average wall time taken by PTC

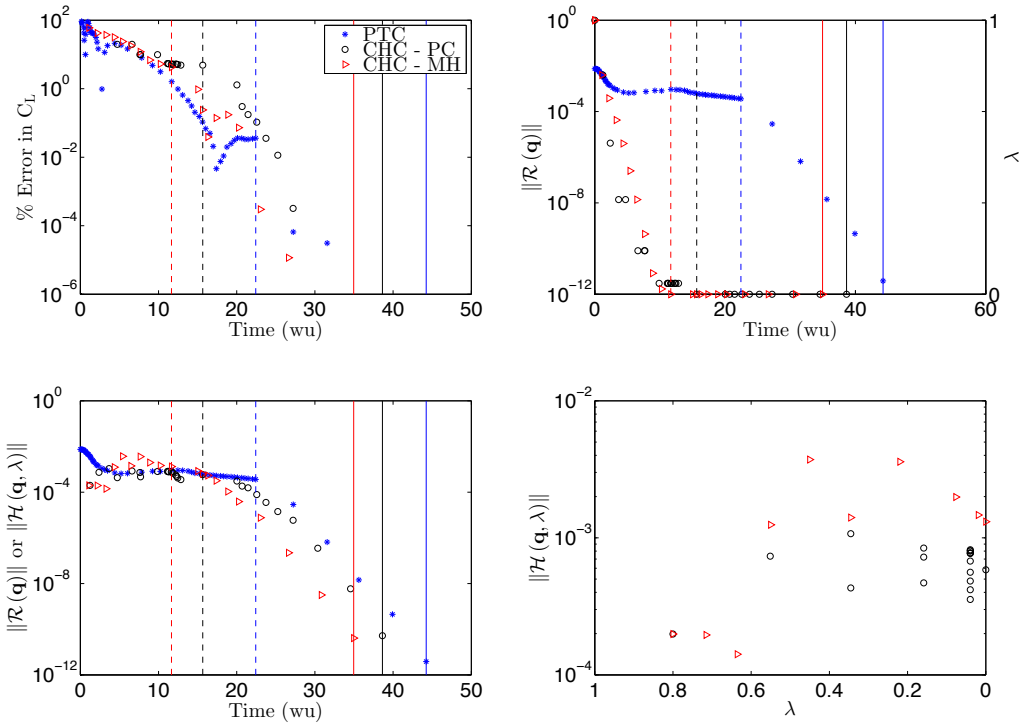


(a) Coarser grid

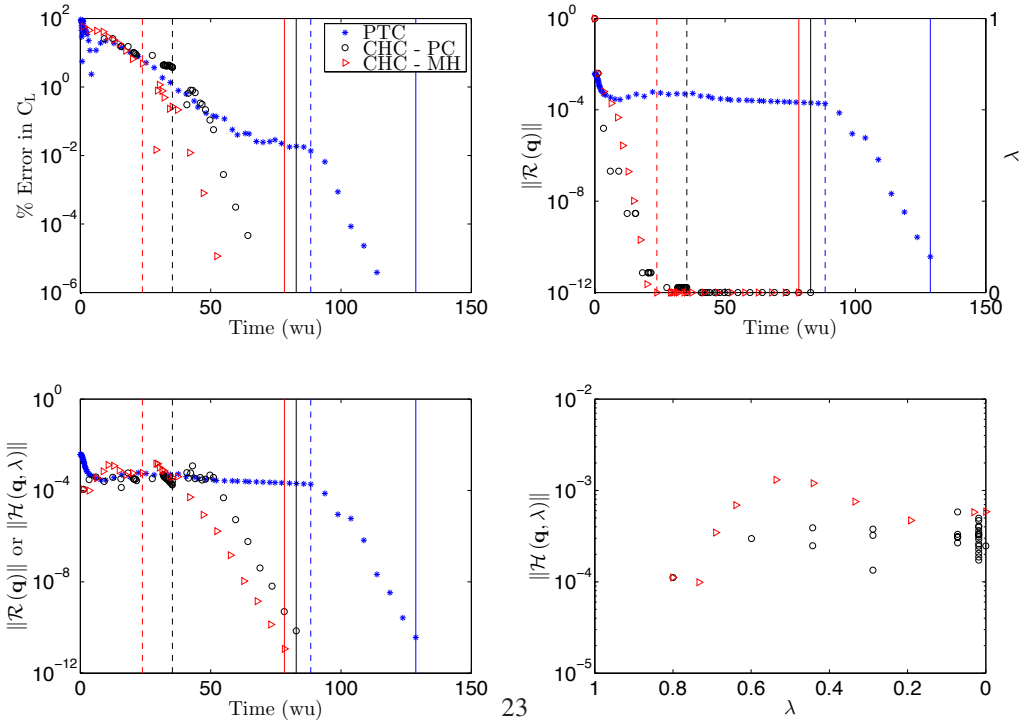


(b) Finer grid

Figure 4: Performance comparison of several continuation algorithms for inviscid flows over the ONERA M6 wing



(a) Coarser grid



(b) Finer grid

Figure 5: Convergence histories for different continuation algorithms for inviscid flow over the ONERA M6 wing at Mach 0.6 and angle of attack 3°

10.3. Turbulent Flow Cases

Three turbulent test suites are investigated. One is a two-dimensional flow over the NACA 0012 airfoil at Reynolds number 4×10^7. The grid has an H topology and consists of 1.92×10^4 nodes divided evenly into 8 blocks. Flow separation is expected in many cases in this suite as they are at high Mach number and angle of attack. The other two suites are on the ONERA M6 wing. One is at Reynolds number 1.172×10^7. The grid has an H-H topology and consists of 2.336064×10^6 nodes divided evenly into 192 blocks. The other is at Reynolds number 1×10^7 on a finer grid consisting of 3.6799488×10^8 nodes divided evenly into 1024 blocks. As with the inviscid cases, the parameters used for all three test suites are listed in Table 2.

The performance data for the NACA 0012 test suite is shown in Figure 6 and Table 2. Convergence histories for all three algorithms for a representative case are shown in Figure 7. Flow solves were completed successfully across the entire range of input parameters investigated. On average, the CHC-PC algorithm converged in 27% less wall time than the PTC method and the CHC-MH algorithm converged in 45% less wall time than PTC, all with 100% success rate. It was found that the approximate matrix-vector products could be used for the CHC-MH algorithm without incurring a robustness penalty.

The performance data for the ONERA M6 test suite on the coarser grid is shown in Figure 8 and Table 2. Convergence histories for all three algorithms for a representative case are shown in Figure 9. The success rate of all three algorithms was also high in this case. The finite-differencing method was used to compute the matrix-vector products for the MH method. While it has been found that using the approximate matrix-vector products for this suite resulted in similar performance statistics, the homotopy residual increased throughout traversing to the point where stability was a concern. From Figure 9, it appears that stability may still be a concern for these cases, even when the finite-differencing method is used for the matrix-vector products. While the need to use finite-difference matrix-vector products has implications on cost, the flow solves still completed in 36% less wall time with the CHC-MH method than with PTC, whereas the CHC-PC method took 14% longer than PTC.

The turbulent test suite on the ONERA M6 with the finer mesh was the least successful for the homotopy methods. The timing data for this test suite is shown in Figure 10 and Table 2. Convergence histories for all three algorithms for a representative test case are shown in Figure 11. The CHC-PC method was found to be unreliable for this test suite unless a traceback method is employed, where the predictor phase is repeated if a subproblem fails to converge to the specified tolerance. Since this method is needed for convergence in many cases, it would appear that the step-length adaptation is not adequate to determine an appropriate step size for this case. Using this method is expensive and inefficient but at least the success rate of PTC could be matched. For the MH cases, the row/column normalization scaling was used instead of the geometric scaling based on the earlier observations that this improves the stability of the algorithm. Since, in our experience, using this scaling results in significantly more error reduction in each linear solve, but also increases the cost of each linear solve significantly, the linear solver tolerance was relaxed to 0.03 for the MH algorithm. On average, the flow solves using the CHC-MH algorithm took about the same amount of time to converge as the PTC flow solves and were successful in one additional case.

10.4. Performance Summary

It was found for the first inviscid test suite that the CHC-MH algorithm converged in an average of 23% less time than the PTC algorithm whereas the CHC-PC algorithm converged in

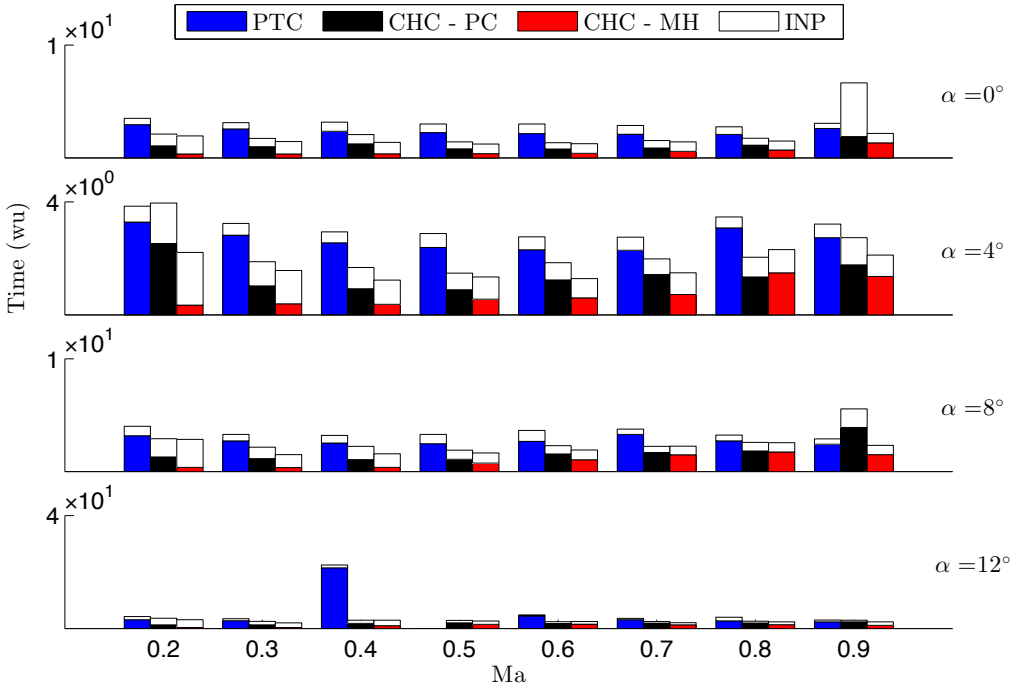


Figure 6: Performance comparison of several continuation algorithms for turbulent flows at Reynolds number 4×10^7 over the NACA 0012 airfoil

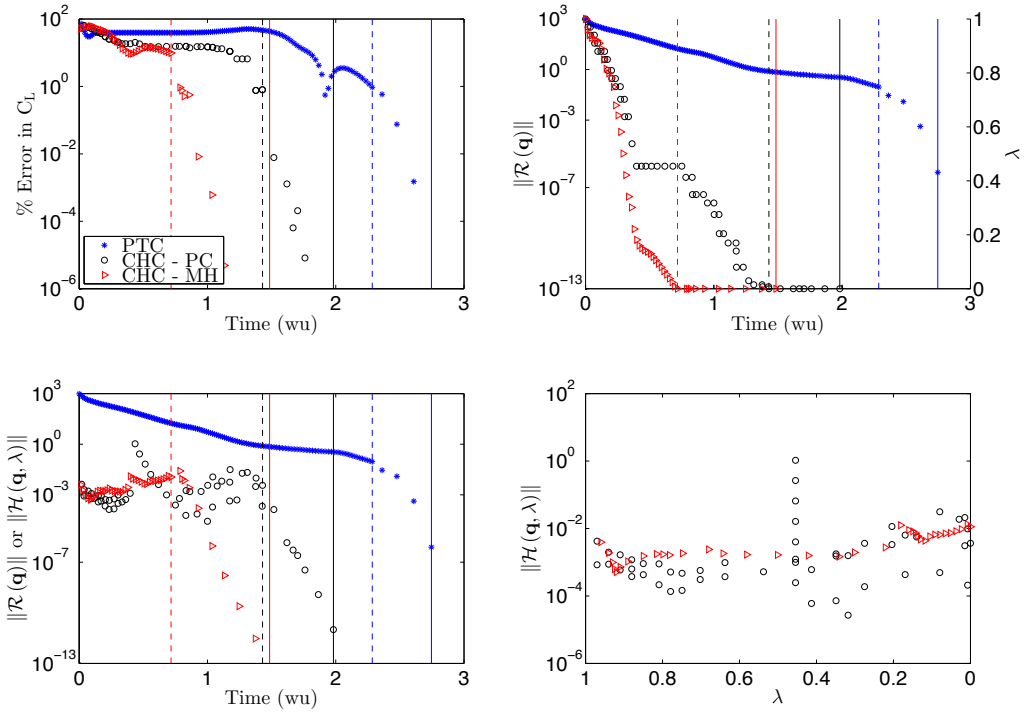


Figure 7: Convergence histories for different continuation algorithms for turbulent flow over the NACA 0012 airfoil at Mach 0.7 and angle of attack 4°

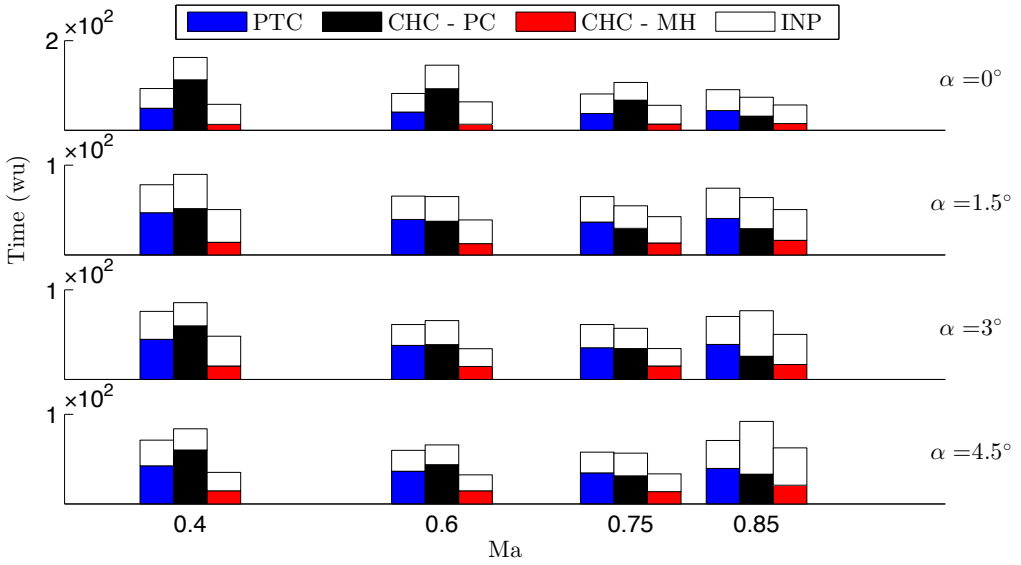


Figure 8: Performance comparison of several continuation algorithms for turbulent flows at Reynolds number 1.172×10^7 over the ONERA M6 wing on the coarser grid

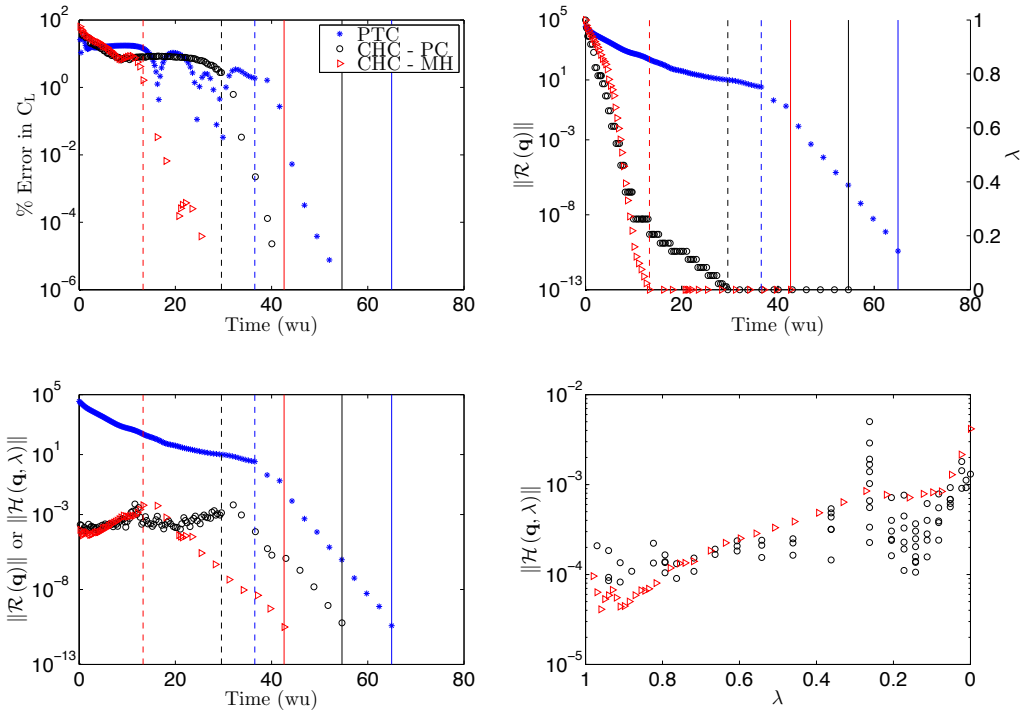


Figure 9: Convergence histories for different continuation algorithms for turbulent flow over the ONERA M6 wing on the coarser grid at Mach 0.75 and angle of attack 1.5°

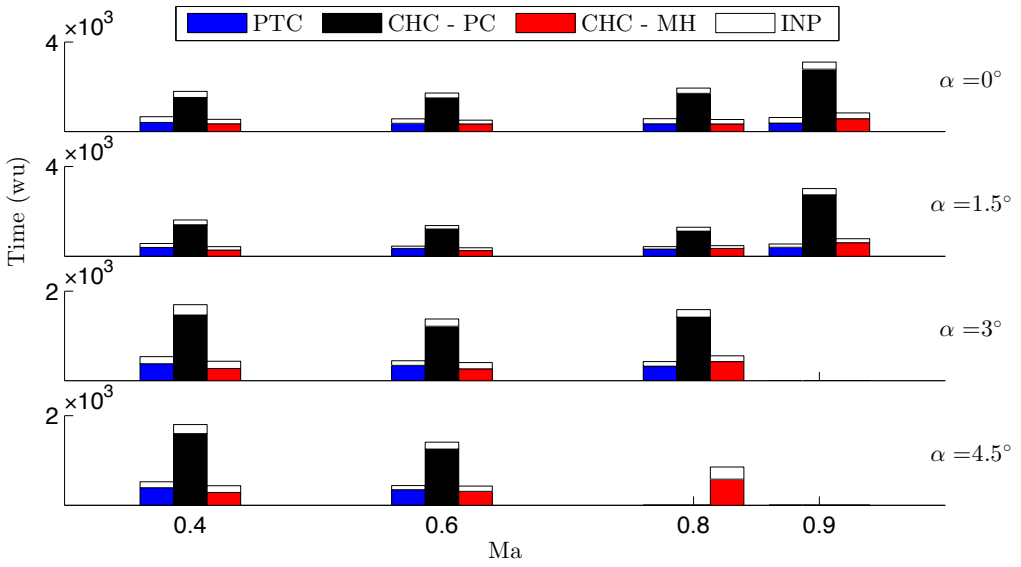


Figure 10: Performance comparison of several continuation algorithms for turbulent flows at Reynolds number 1×10^7 over the ONERA M6 wing on the finer grid

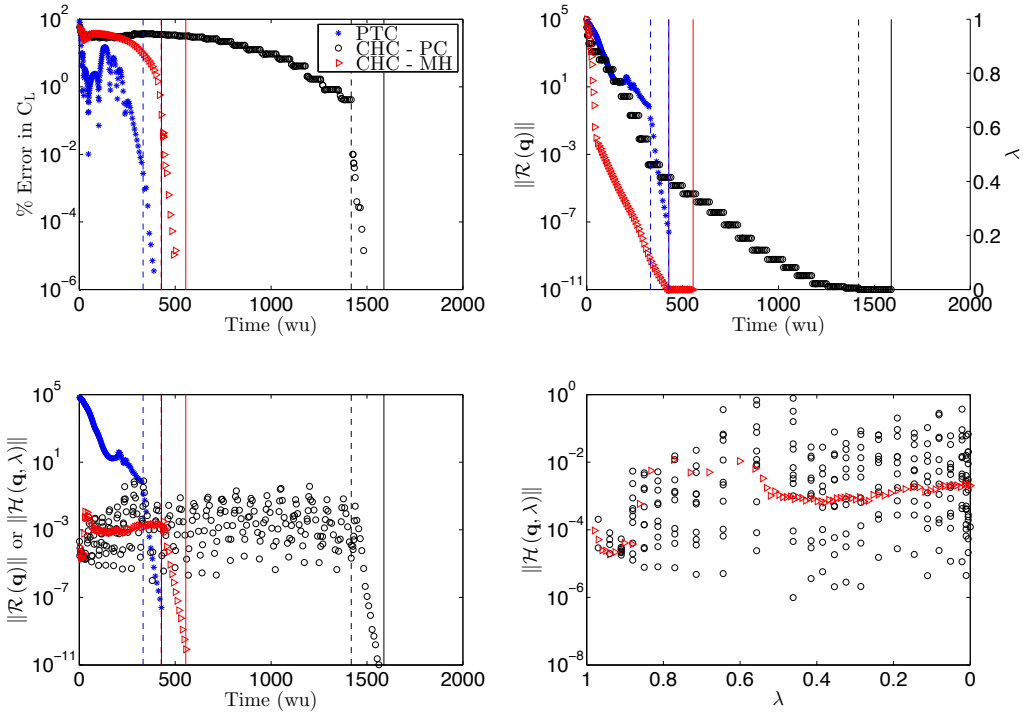


Figure 11: Convergence histories for different continuation algorithms for turbulent flow over the ONERA M6 wing on the finer grid at Mach 0.8 and angle of attack 3°

only 12% less time. A similar success rate was observed for each algorithm. When the mesh was refined, the CPU time reduction in the convergence of the CHC-MH algorithm increased to 34%. Moreover, only 6 additional cases failed, whereas 11 additional cases failed for PTC.

It was found for the two-dimensional RANS test case that a 45% CPU time reduction could be obtained over PTC with the CHC-MH algorithm whereas only 27% CPU time reduction was observed with the CHC-PC algorithm. For both three-dimensional cases, the CHC-PC algorithm performed worse than the PTC algorithm. For the first three-dimensional RANS test case, the CHC-MH algorithm converged in 34% less time than the PTC algorithm. One reason that the MH algorithm out-performed the PC algorithm is that the step sizes are generally smaller with the MH algorithm, giving the algorithm more flexibility in responding to regions of high curvature which are more problematic for the RANS cases than the inviscid cases. The final three-dimensional RANS test case proved the most problematic. We were able to demonstrate that the CHC-MH algorithm can converge in the same amount of time as PTC but converged in one additional case. However, the convergence problems associated with the use of the Krylov solver made parameter selection difficult for this case. At any rate, we were unable to choose parameters for the PC algorithm that made it even remotely competitive with PTC for this case and so the performance benefit of the MH algorithm over the PC algorithm is clearly established for all test suites.

11. Conclusions

A new monolithic homotopy continuation algorithm was developed and conditional local convergence to the homotopy curve was proved. An effective step-length adaptation methodology was also presented for the new algorithm. The new algorithm was demonstrated through numerical testing to be more efficient than the classical predictor-corrector homotopy continuation algorithm as well as an implementation of the pseudo-transient continuation algorithm when applied to a computational aerodynamic flow solver for some three-dimensional inviscid and turbulent external aerodynamic flows using a Newton-Krylov solution methodology.

Acknowledgments

The authors gratefully acknowledge financial assistance from the Natural Science and Engineering Research Council (NSERC), the Canada Research Chairs program, and the University of Toronto. Computations were performed on the GPC supercomputer at the SciNet HPC Consortium. SciNet is funded by: the Canada Foundation for Innovation under the auspices of Compute Canada; the Government of Ontario; Ontario Research Fund - Research Excellence; and the University of Toronto.

References

[1] Allgower, E. L., Georg, K., 1990. Introduction to Numerical Continuation Methods. Society for Industrial and Applied Mathematics.

[2] Brown, D. A., Buckley, H. P., Osusky, M., Zingg, D. W., January 2015. Performance of a Newton-Krylov-Schur algorithm for the numerical solution of the steady Reynolds-Averaged Navier-Stokes equations. In: 53rd AIAA Aerospace Sciences Meeting. Kissimee, Florida, United States, aIAA 2015-1744.

[3] Brown, D. A., Zingg, D. W., June 2013. Advances in homotopy-based globalization strategies in computational fluid dynamics. AIAA-2013-2944.

[4] Buckner, H. M., Pollul, B., Rasch, A., 2009. On CFL evolution strategies for implicit upwind methods in linearized Euler equations. Int. J. Numer. Meth. Fl. 59, 1–18.
[5] Carey, G. F., Krishnan, R., 1985. Continuation techniques for a penalty approximation of the Navier-Stokes equations. Comput. Method. Appl. M. 48, 265–282.
[6] Carpenter, M. H., Gottlieb, D., Abarbanel, S., 1994. Time-stable boundary conditions for finite-difference schemes solving hyperbolic systems: methodology and application to high-order compact schemes. J. Comput. Phys. 111 (2), 220–236.
[7] Ceze, M. A., Fidkowski, K. J., 2015. Constrained pseudo-transient continuation. Int. J. Numer. Meth. Eng. 102 (11), 1683–1703.
[8] Chisholm, T. T., Zingg, D. W., 2009. A Jacobian-free Newton-Krylov algorithm for compressible turbulent fluid flows. J. Comput. Phys. 228, 3490–3507.
[9] Del Rey Fernández, D. C., Hicken, J. E., Zingg, D. W., 2014. Review of summation-by-parts operators with simultaneous approximation terms for the numerical solution of partial differential equations. Comput. Fluids 95, 171–196.
[10] Dembo, R. S., Eisenstat, S. C., Steihaug, T., 1982. Inexact Newton methods. SIAM J. Numer. Anal. 19 (2), 400–408.
[11] Dias, S., Zingg, D. W., June 2009. A high-order parallel Newton-Krylov flow solver for the Euler equations. AIAA-2009-3657.
[12] Funaro, D., Gottlieb, D., 1988. A new method of imposing boundary conditions in pseudospectral approximations of hyperbolic equations. Math. Comput. 51, 599–613.
[13] Georg, K., 1983. A note on stepsize control for numerical curve following. In: Eaves, B. C., Gould, F. J., Peitgen, H., Todd, M. J. (Eds.), Homotopy Methods and Global Convergence. Plenum Press, New York, pp. 145–154.
[14] Getz, N. H., 1995. Dynamic inversion of nonlinear maps with applications to nonlinear control and robotics. Ph.D. thesis, University of California at Berkeley, Berkeley, California, USA.
[15] Getz, N. H., Marsden, J. E., December 1995. A dynamic inverse for nonlinear maps. In: Proceedings of the 34th IEEE Conference on Decision and Control. Vol. 4. pp. 4218–4223.
[16] Getz, N. H., Marsden, J. E., June 1995. Tracking implicit trajectories. In: IFAC Symposium on Nonlinear Control Systems Design. Tahoe City.
[17] Hartman, P., 1982. Ordinary Differential Equations, 2nd Edition. Birkhauser, Berlin.
[18] Hicken, J. E., Buckley, H., Osusky, M., Zingg, D. W., June 2011. Dissipation-based continuation: a globalization for inexact-Newton solvers. AIAA 2011-3237.
[19] Hicken, J. E., Zingg, D. W., 2008. A parallel Newton-Krylov solver for the Euler equations discretized using simultaneous approximation terms. AIAA J. 46 (11), 2773–2786.
[20] Hicken, J. E., Zingg, D. W., June 2009. Globalization strategies for inexact-Newton solvers. AIAA-2009-4139.
[21] Jameson, A., Schmidt, W., Turkel, E., June 1981. Numerical solution of the Euler equations by finite-volume methods using Runge-Kutta time-stepping schemes. AIAA-1981-1259.
[22] Jameson, A., Vassberg, J. C., Ou, K., 2012. Further studies of airfoils supporting non-unique solutions in transonic flow. AIAA J. 50 (12), 2865–2881.
[23] Kelley, C. T., Keyes, D. E., 1998. Convergence analysis of pseudo-transient continuation. SIAM J. Numer. Anal. 35 (2), 508–523.
[24] Knoll, D. A., Keyes, D. E., 2004. Jacobian-free Newton-Krylov methods: a survey of approaches and applications. J. Comput. Phys. 193, 357–397.
[25] Kreiss, H., Scherer, G., 1974. Finite element and finite difference methods for hyperbolic partial differential equations. In: de Boor, C. (Ed.), Mathematical Aspects of Finite Elements in Partial Differential Equations: proceedings of a symposium conducted by the Mathematics Research Center, the University of Wisconsin. Mathematics Research Centre, the University of Wisconsin, Academic Press, pp. 195–212.
[26] Lomax, H., Pulliam, T. H., Zingg, D. W., 2001. Fundamentals of Computational Fluid Dynamics. Springer-Verlag.
[27] Mulder, W. A., van Leer, B., 1985. Experiments with explicit upwind methods for the Euler equations. J. Comput. Phys. 59, 232–246.
[28] Nielsen, E. J., Anderson, W. K., Walters, R. W., Keyes, D. E., June 1995. Application of Newton-Krylov methodology to a three-dimensional unstructured Euler code. AIAA-95-1733.
[29] Osusky, M., Zingg, D. W., 2013. A parallel Newton-Krylov-Schur flow solver for the Navier-Stokes equations discretized using summation-by-parts operators. AIAA J. 51 (12), 2833–2851.
[30] Pulliam, T. H., 1986. Artificial dissipation models for the Euler equations. AIAA J. 24 (12), 1931–1940.
[31] Riley, D. S., Winters, K. H., 1990. A numerical bifurcation of natural convection in a tilted two-dimensional porous cavity. J. Fluid Mech. 215, 309–329.
[32] Saad, Y., 2003. Iterative Methods for Sparse Linear Systems, 2nd Edition. SIAM, Philadelphia, PA.
[33] Spalart, P. R., Allmaras, S. R., January 1992. A one-equation turbulence model for aerodynamic flows. AIAA 92-0439.

[34] Wales, C., Gaitonde, A. L., Jones, D. P., Avitabile, D., Champneys, A. R., 2012. Numerical continuation of high Reynolds number external flows. Int. J. Numer. Meth. Fl. 68 (2), 135–159.
[35] Winters, K. H., 1987. A bifurcation study of laminar flow in a curved rectangular cross-section. J. Fluid Mech. 180, 343–369.