Computation of viscoelastic fluid flows at high Weissenberg number using continuation methods

Jason S. Howell

June 14, 2007

Abstract. The numerical simulation of viscoelastic fluid flow becomes more difficult as a physical parameter, the Weissenberg number, increases. Specifically, at a Weissenberg number larger than a critical value, the iterative nonlinear solver fails to converge, a phenomenon known as the High Weissenberg Number Problem. In this work we describe the application and implementation of continuation methods to the nonlinear Johnson-Segalman model for steady-state viscoelastic flows. Simple, natural, and pseudo-arclength continuation approaches in Weissenberg number are investigated for a discontinuous Galerkin finite element discretization of the equations. Computations are performed for a benchmark contraction flow and several aspects of the performance of the continuation methods, including high Weissenberg number limits, are discussed.

Key words. viscoelastic fluid, continuation, finite element, discontinuous Galerkin, Weissenberg number

AMS Mathematics subject classifications. 65N30

1 Introduction

The study of non-Newtonian fluid flow continues to be a topic of interest on many fronts. To accompany theoretical and experimental observations, researchers have developed computational algorithms for simulating viscoelastic fluid flow. Of these, methods which utilize a finite element approach are particularly prominent, and they usually incorporate some stabilization to handle the hyperbolic behavior of the constitutive model of the fluid. However, difficulties in the numerical solution of the equations of viscoelastic fluid flow remain, in particular the high Weissenberg number problem is an issue that garners significant study. This problem usually manifests in the inability for nonlinear solvers to converge for increasing values of the Weissenberg (Deborah) number, a physical parameter that is related to the elasticity of the fluid. At present, the high Weissenberg number limit encountered in numerical studies is generally regarded as numerical artifact and not a property of the modeling equations [2, 39].
For nonlinear problems, continuation methods have been developed as a way to analyze the behavior of solutions as a problem parameter is varied [1, 28, 29, 43, 38], and numerous computational algorithms and software packages have been developed to implement these methods [44, 45]. Predictor-corrector continuation methods trace the solution manifold through the parameter space by first forming a good approximation to the solution at an incremented value of the parameter and then correcting through an iterative scheme. This approach may identify singular and bifurcation points along the manifold which are of particular interest.

Continuation methods have been studied within the context of structural mechanics and elasticity [37, 32, 22] as well as fluid dynamics. For issues in simulating the Navier-Stokes equations at high Reynolds number, a common approach is to perform continuation in Reynolds number [24]. Carey and Krishnan [7] described a continuation method applied to a penalty approximation for Navier-Stokes, gave conditions for convergence for the method, and presented numerical results for the driven cavity problem. The authors also formulated a method for continuation in arc length. Gunzburger and Peterson [25] investigated predictor and steplength selection for continuation in Reynolds number and concluded that for certain values of the Reynolds number, the parameter steplength can be chosen independently of the type of predictor step used. Recently, de Almeida and Derby [13] described natural and pseudo-arclength continuation algorithms adapted for large-scale simulations of driven-cavity flows and successfully computed approximations for large values of the Reynolds number.

Similarly, continuation for viscoelastic fluid flow can be performed for increasing values of the Weissenberg number. Various implementations of continuation in Weissenberg number can be found in [9, 40, 49, 11]. In [34], Mendelson, et al. described an algorithm for continuation in Weissenberg number for Maxwell and second-order constitutive equations using a Galerkin finite element method. The authors observed numerically that, as the critical value of the Weissenberg number is approached, the length of the step that can be taken became infeasibly small. In [48], Yeh, et al. used natural and arclength continuation methods to arrive at a bifurcation in the numerical solution of an upper convected Maxwell fluid, at which the authors conjected is a product of the mathematical model of the fluid. However, later work has shown that the multiple solutions were an artifact of the numerical approximation (see [39]).

The objective of this work is to describe the application and implementation of continuation algorithms for the discontinuous Galerkin finite element approximation of the steady-state Johnson-Segalman model for viscoelastic fluid flow, and to use these algorithms to study the behavior of the resulting nonlinear system of equations as the Weissenberg number is increased. Simple and natural continuation methods are investigated with incremental increases in the Weissenberg number. Pseudo-arclength continuation reparametrizes the equations with respect to an arclength-like parameter and incorporates an additional constraint to be solved with the underlying system. The remainder of this paper is organized as follows: in Section 2, the original continuous problem and its discrete approximation are described. In Section 3 simple and natural continuation approaches are discussed and applied to the discrete problem. In Section 4 the pseudo-arclength continuation method and various choices for the pseudo-arclength constraint are discussed and applied to the discrete problem. Numerical experiments with continuation methods are performed on the benchmark 4:1 contraction problem in Section 5.
2 Problem Description

Let $\Omega$ be a bounded domain in $\mathbb{R}^d$, $d = 2$ or 3, with Lipschitz continuous boundary $\Gamma$. Let $\Gamma_{in}$ denote the subset of $\Gamma$ upon which fluid flows into $\Omega$. Let $u$ denote fluid velocity and $p$ denote the pressure. The flow is assumed to be creeping (or inertialess) and the extra stress, with polymeric contribution $\sigma$, is described by a Johnson-Segalman [27] constitutive law. The steady-state model of this behavior is described by

$$\sigma + \lambda (u \cdot \nabla)\sigma + \lambda g_a(\sigma, \nabla u) - 2 \alpha D(u) = 0 \quad \text{in } \Omega, \quad \text{(2.1)}$$

$$- \nabla \cdot \sigma - 2(1 - \alpha) \nabla \cdot D(u) + \nabla p = f \quad \text{in } \Omega, \quad \text{(2.2)}$$

$$\text{div } u = 0 \quad \text{in } \Omega, \quad \text{(2.3)}$$

$$u = u_{\Gamma} \quad \text{on } \Gamma, \quad \text{(2.4)}$$

$$\sigma = \sigma_{\Gamma_{in}} \quad \text{on } \Gamma_{in}. \quad \text{(2.5)}$$

In (2.1)-(2.5) $\sigma$ denotes the rank 2 (symmetric) tensor

$$\sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{xy} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{xz} & \sigma_{yz} & \sigma_{zz} \end{bmatrix}$$

in three spatial dimensions, and $\lambda$ is the Weissenberg number (defined as the product of the relaxation time of the fluid and a characteristic strain rate). The rank 2 tensor $\nabla u$ denotes the gradient of $u$ with components $(\nabla u)_{i,j} = \partial u_i / \partial x_j$ (consistent with the definition used in [3, 6, 42]).

Assume that $p$ has zero mean value over $\Omega$. In (2.1) and (2.2), $D(u) := (\nabla u + (\nabla u)^T)/2$ is the rate of the strain tensor, $\alpha \in (0, 1)$, and $f$ the body force. In (2.1), $g_a(\sigma, \nabla u)$ is defined by

$$g_a(\sigma, \nabla u) := \frac{1 - a}{2} (\sigma \nabla u + \nabla u^T \sigma) - \frac{1 + a}{2} (\nabla u \sigma + \sigma \nabla u^T)$$

for $a \in [-1, 1]$.

Remark 2.1 For the case $a = 1$ the Johnson-Segalman model reduces to the well-known Oldroyd-B model [3, 42].

2.1 Continuous Problem

For $\mathcal{D} \subset \Omega$, denote the standard Sobolev spaces by $W^{m,p}(\mathcal{D})$ with norms $\| \cdot \|_{m,p,\mathcal{D}}$ if $p < \infty$, $\| \cdot \|_{m,\infty,\mathcal{D}}$ if $p = \infty$. The Sobolev space $W^{m,2}$ is denoted by $H^m$ with the norm $\| \cdot \|_m$. The corresponding space of vector-valued or tensor-valued functions is given by $\mathbf{H}^m$. If $\mathcal{D} = \Omega$, $\mathcal{D}$ is omitted, i.e., $(\cdot, \cdot) = (\cdot, \cdot)_\Omega$ and $\| \cdot \| = \| \cdot \|_\Omega$.

Existence of a solution to the problem (2.1)-(2.5) was shown by Renardy [41] under a small data assumption. (See also [23] and [19].) Specifically, if $\Omega$ has a $C^\infty$-smooth boundary, with $f$ and $u_{\Gamma}$ sufficiently regular and small, the problem (2.1)-(2.5) admits a unique bounded solution $(\sigma, u, p) \in H^2(\Omega) \times H^3(\Omega) \times H^2(\Omega)$. 

3
2.2 Variational Formulation

Next we define the function spaces for the velocity $u$, the pressure $p$ and the stress $\sigma$. Let

$$X_r := H_0^1(\Omega) = \{v \in H^1(\Omega) : v = r \text{ on } \Gamma\},$$

$$S := L_0^2(\Omega) = \{q \in L^2(\Omega) : \int_\Omega q \, d\Omega = 0\},$$

$$\Sigma_s := (L^2(\Omega))^{d \times d} \cap \{\tau = (\tau_{ij}) : \tau_{ij} = \tau_{ji}, \ u \cdot \nabla \tau \in (L^2(\Omega))^{d \times d}, \ \text{and} \ \tau = s \ \text{on } \Gamma_{in}\}.$$

The variational formulation is: Given $f \in H^{-1}(\Omega)$, find $(\sigma, u, p) \in \Sigma_{\sigma_{\Gamma_{in}}} \times X_{ur} \times S$ such that

$$\langle \sigma, \tau \rangle + \lambda ((u \cdot \nabla)\sigma, \tau) + \lambda (g_0(\sigma, \nabla u), \tau) - 2\alpha(D(u), \tau) = 0, \ \forall \tau \in \Sigma_0, \quad (2.6)$$

$$\langle \sigma, D(v) \rangle + 2(1 - \alpha)(D(u), D(v)) - (p, \nabla \cdot v) = (f, v), \ \forall v \in X_0, \quad (2.7)$$

$$\langle q, \nabla \cdot u \rangle = 0, \ \forall q \in S. \quad (2.8)$$

Introduce the bilinear form $A$ defined on $\Sigma \times X$ by

$$A((\sigma, u), (\tau, v)) := \langle \sigma, \tau \rangle - 2\alpha(D(u), \tau) + 2\alpha(\sigma, D(v)) + 4\alpha(1 - \alpha)(D(u), D(v)). \quad (2.9)$$

2.3 Finite Element Approximation

Let $T_h$ denote a triangulation of $\Omega$ such that $\overline{\Omega} = \{\cup K : K \in T_h\}$. Assume that there exist positive constants $c_1, c_2$ such that

$$c_1 h \leq h_K \leq c_2 \rho_K,$$

where $h_K$ is the diameter of $K$, $\rho_K$ is the diameter of the greatest ball included in $K$, and $h = \max_{K \in T_h} h_K$. Let $P_h(K)$ denote the space of polynomials of degree less than or equal to $k$ on $K \in T_h$. Let $P_h(T_h) = \cup_{K \in T_h} P_h(K)$. Let $T_h^u : H^1(\Omega) \to P_h(T_h)$ and $T_h^\sigma : (L^2(\Omega))^{d \times d} \to P_h(T_h)$ denote suitable interpolation operators [10, 6]. Define the following finite element spaces (Taylor-Hood) for the approximation of $u$ and $p$:

$$X_r^h := \{v^h \in H^1(\Omega) \cap (C^0(\overline{\Omega}))^d : v^h|_K \in P_2(K)^d, \ \forall K \in T_h, \ v^h = T_2^u r \text{ on } \Gamma\},$$

$$S^h := \{q^h \in S \cap C^0(\overline{\Omega}) : q^h|_K \in P_1(K), \ \forall K \in T_h\}.$$

For the approximation of the constitutive equation (2.1) the discontinuous Galerkin method is used, where the stress $\sigma$ is approximated in the discontinuous finite element space of piecewise linear functions:

$$\Sigma_s^h := (L^2(\Omega))^{d \times d} \cap \{\tau^h = (\tau_{ij}^h) : \tau_{ij}^h = \tau_{ji}^h, \ \tau^h|_K \in P_1(K)^{d \times d}, \ \forall K \in T_h, \ \tau^h = T_1^\sigma s \text{ on } \Gamma_{in}\}.$$

Below some notation used in [3] is introduced to describe and analyze an approximate solution obtained using the discontinuous Galerkin method. Let

$$\Gamma^h = \{\cup \partial K, K \in T_h\} \setminus \Gamma,$$

$$\partial K^- (u) := \{x \in \partial K, \ u \cdot n < 0\},$$

$$\partial K^+ (u) := \{x \in \partial K, \ u \cdot n > 0\}.$$
where $\partial K$ is the boundary of $K \in T_h$ and $n$ is the outward unit normal vector on $\partial K$, and

$$
\tau^\pm(u) := \lim_{\epsilon \to 0^\pm} \tau(x + \epsilon u(x)).
$$

Also, let

$$(\sigma, \tau)_h := \sum_{K \in T_h} (\sigma, \tau)_K,$$

$$
(\sigma^+, \tau^+)_{h, u} := \sum_{K \in T_h} \int_{\partial K^-(u)} (\sigma^+(u) : \tau^+(u)) |n \cdot u| \, ds,
$$

$$
\langle \langle \sigma \rangle \rangle_{h, u} := (\sigma^+, \sigma^+)_{h, u}^{1/2},
$$

$$
\|\tau\|_{0, \Gamma_h} := \left( \sum_{K \in T_h} \|\tau\|_{0, \partial K}^2 \right)^{1/2},
$$

for $\sigma, \tau \in \prod_{K \in T_h} (L^2(K))^{d \times d}$, and

$$
\|\xi\|_{m, h} := \left( \sum_{K \in T_h} \|\xi\|_{m, K}^2 \right)^{1/2},
$$

for $\xi \in \prod_{K \in T_h} (W^{m,2}(K))^{d \times d}$.

Introduce the operator $B_h$ on $X_h \times \Sigma_h \times \Sigma_h$ defined by

$$
B_h(u^h, \sigma^h, \tau^h) := ((u^h \cdot \nabla)\sigma^h, \tau^h)_h + \frac{1}{2}(\nabla \cdot u^h \sigma^h, \tau^h) + \langle \langle \sigma^+ - \sigma^-, \tau^+ \rangle \rangle_{h, u^h}.
$$

(2.11)

Note that the second term vanishes when $\nabla \cdot u^h = 0$.

On occasion it will be necessary to consider (2.12) without the associated jump term. Therefore, let

$$
\tilde{B}_h(u^h, \sigma^h, \tau^h) := ((u^h \cdot \nabla)\sigma^h, \tau^h)_h + \frac{1}{2}(\nabla \cdot u^h \sigma^h, \tau^h).
$$

(2.13)

The discontinuous Galerkin finite element approximation of (2.6)–(2.8) is then as follows. Given $f \in H^{-1}(\Omega)$, find $(\sigma^h, u^h, p^h) \in \Sigma_h^{\Gamma_{\text{in}}} \times X_h^{\text{u}} \times S^h$ such that

$$
(\sigma^h, \tau^h) + \lambda B_h(u^h, \sigma^h, \tau^h) + \lambda(g_0(\sigma^h, \nabla u^h), \tau^h) - 2\alpha(D(u^h), \tau^h) = 0, \quad \forall \tau^h \in \Sigma_0^h,
$$

(2.14)

$$
(\sigma^h, D(v^h)) + 2(1 - \alpha)(D(u^h), D(v^h)) - (p^h, \nabla \cdot v^h) = (f, v^h), \quad \forall v^h \in X_0^h,
$$

(2.15)

$$
(q^h, \nabla \cdot u^h) = 0, \quad \forall q^h \in S^h.
$$

(2.16)

Existence of a solution to the discrete problem (2.14)-(2.16) has been shown by Baranger and Sandri [3] under the assumption that the continuous problem (2.1)-(2.5) has a bounded solution $(\sigma, u, p) \in H^2(\Omega) \times H^3(\Omega) \times H^2(\Omega)$. The error estimates

$$
\|\sigma - \sigma^h\|_0 + \|\nabla (u - u^h)\|_0 \leq Ch^{3/2}, \quad \|p - p^h\|_0 \leq Ch^{3/2}
$$

for constant $C > 0$, are also proven in [3] under the same assumptions.
### 2.4 Newton Iteration

For ease of notation, let $\Pi^h = \Sigma^h_0 \times X^h_0 \times S^h$ and let $\Pi^h_{bc} = \Sigma^h_{\Gamma_{in}} \times X^h_{u_i} \times S^h$, with $u = (\sigma^h, u^h, p^h) \in \Pi^h_{bc}$ and $v = (\tau^h, v^h, q^h) \in \Pi^h$. Also let $(\Pi^h)^*$ denote the dual space of $\Pi^h$. With $A$ as defined in (2.9), (2.14)-(2.16) can be written as

$$A((\sigma^h, u^h), (\tau^h, v^h)) + \lambda B^h(u^h, \sigma^h, \tau^h) + \lambda (g_a(\sigma^h, \nabla u^h), \tau^h) - (p^h, \nabla \cdot v^h) + (q^h, \nabla \cdot u^h) = 2\alpha(f, v^h) \quad \forall (\tau^h, v^h, q^h) \in \Pi^h. \quad (2.17)$$

For the remainder of this chapter, the superscript $h$ will be dropped from the expression of $\sigma^h, u^h, p^h, \tau^h, v^h, q^h$, and it is assumed that these quantities are elements of the discrete approximation spaces. Define the function $G : \Pi^h_{bc} \rightarrow (\Pi^h)^*$ by

$$\langle G(u), v \rangle := A((\sigma, u), (\tau, v)) + \lambda B(u, \sigma, \tau) + \lambda (g_a(\sigma, \nabla u), \tau) - (p, \nabla \cdot v) + (q, \nabla \cdot u) - 2\alpha(f, v) \quad \forall v \in \Pi^h. \quad (2.18)$$

One approach to solving $G(u) = 0$ is to solve the system of nonlinear equations (2.18) via Newton iteration. Let $G_u = \partial G/\partial u$ denote the Jacobian of $G$ with respect to the unknown $u$. Given an initial iterate $u^0 \in \Pi^h_{bc}$, for $i = 0, 1, 2, \ldots$, solve for $\delta u^i \in \Pi^h$ satisfying the linear system

$$\langle G_u(u^i)(\delta u^i), v \rangle = -\langle G(u^i), v \rangle \quad \forall v \in \Pi^h, \quad (2.19)$$

setting $u^{i+1} := u^i + \delta u^i$ until $\|u^{i+1} - u^i\| < tol$ for an appropriate choice of norm and tolerance $tol$. The Jacobian is formed using the Fréchet derivative of $G$ at $u^i$ in the direction of $w$, i.e.,

$$\langle G_u(u^i)(w), v \rangle := \lim_{\varepsilon \to 0} \frac{\langle G(u^i + \varepsilon w), v \rangle - \langle G(u^i), v \rangle}{\varepsilon}.$$

Combining (2.18) and (2.19), the Newton iteration consists of the following linear problem: For $i = 0, 1, 2, \ldots$, solve for $u^{i+1} = (\sigma^{i+1}, u^{i+1}, p^{i+1}) \in \Pi^h_{bc}$ satisfying

$$A((\sigma^{i+1} - \sigma^i, u^{i+1} - u^i), (\tau, v)) + \lambda B(u^{i+1}, \sigma^{i+1} - \sigma^i, \tau) + \lambda B(u^{i+1} - u^i, \sigma^i, \tau) + \lambda (g_a(\sigma^{i+1} - \sigma^i, \nabla u^{i+1}), \tau) + \lambda (g_a(\sigma^i, \nabla u^{i+1} - u^i), \tau) - (p^{i+1} - p^i, \nabla \cdot v) + (q, \nabla \cdot u^{i+1} - u^i) - A((\sigma^i, u^i), (\tau, v)) - \lambda B(u^i, \sigma^i, \tau) - \lambda(g_a(\sigma^i, \nabla u^i), \tau) + (p^i, \nabla \cdot v) - (q, \nabla \cdot u^i) + 2\alpha(f, v) \quad \forall (\tau, v, q) = v \in \Pi^h. \quad (2.20)$$

The term

$$B^h(u^{i+1} - u^i, \sigma^i, \tau) = (u^{i+1} - u^i, \nabla)\sigma^i, \tau)_h + \frac{1}{2}(\nabla \cdot (u^{i+1} - u^i)\sigma^i, \tau) + (\sigma^{i+1} - \sigma^i, \tau^+_h u^{i+1} - u^i) \quad (2.21)$$

requires the calculation of the stress jump in the direction of an unknown velocity. However, the contribution of the jump term

$$\langle \sigma^{i+} - \sigma^{i-}, \tau^+ \rangle_{h, u^{i+1} - u^i} = \sum_{K \in \partial \tilde{K} \cap (u^{i+1} - u^i)} \int_{K \cap (u^{i+1} - u^i)} \left( \left( \sigma^{i+}(u^{i+1} - u^i) - \sigma^{i-}(u^{i+1} - u^i) \right) : \tau^+(u^{i+1} - u^i) \right) \cdot |n \cdot (u^{i+1} - u^i)| \ ds,$$
itself is assumed to be negligible, as $u^{i+1} - u^i = \delta u^i$ is usually small. Thus the jump term is dropped from (2.21). With this adjustment we can write (2.20) as

$$A((\sigma^{i+1}, u^{i+1}), (\tau, v)) + \lambda B^h(u^i, \sigma^{i+1}, \tau) + \lambda \tilde{B}^h(u^{i+1}, \sigma^i, \tau)$$

$$+ \lambda (g_a(\sigma^{i+1}, \nabla u^i), \tau^h) + \lambda (g_a(\sigma^i, \nabla u^{i+1}), \tau) - (p^{i+1}, \nabla \cdot v) + (g, \nabla \cdot u^{i+1})$$

$$= \lambda \tilde{B}^h(u^i, \sigma^i, \tau) + \lambda (g_a(\sigma^i, \nabla u^i), \tau) + 2\alpha(f, v) \quad \forall (\tau, v, q) = v \in \Pi^h. \quad (2.22)$$

This iteration is performed for fixed values of the parameters $a, \alpha$, and the Weissenberg number $\lambda$. In order for (2.22) to converge, the initial iterate $u^0$ must be within the radius of convergence of the solution of the discrete problem $G(u) = 0$.

At some point in the process of increasing $\lambda$, a value $\lambda^*$ will be encountered for which the nonlinear iteration fails for values of $\lambda$ just beyond $\lambda^*$. As described by Owens and Phillips [39], there are a number of possibilities for the behavior of solution curves of the continuous problem (2.1)-(2.3) for increasing Weissenberg number. Although the interest here is the behavior of the discrete system (2.14)-(2.16), the same possibilities apply. Some of these are shown in Figure 2.1. To visualize this, consider a simplified planar representation of the solution manifold of $G$ in the $(u, \lambda)$ space (for sake of illustration consider $\lambda$ on the horizontal axis). Figure 1(a) represents the situation in which no solutions exist for (2.1)-(2.3) beyond $\lambda^*$. In Figure 1(b), there is a gap in the solution curve, i.e., a range of $\lambda$ for which solutions do not exist. The presence of a turning point in Figure 1(c) may indicate multiple solutions for certain values of $\lambda$, and a bifurcation point (shown in Figure 1(d)) is present when multiple branches of the solution originate from $\lambda^*$. In addition, the possibility of a unique solution for all nonnegative $\lambda$ exists, however the previously described Newton iteration still may encounter a critical value for various choices of the parameters $a$ and $\alpha$. 

Figure 2.1: Some possibilities for the behavior of solution curves for high Weissenberg number.
3 Continuation in Weissenberg Number

As discussed in Section 1, convergence of the nonlinear iteration becomes more problematic as the Weissenberg number is increased. The defect-correction method has been applied to steady-state viscoelastic flows [18, 17, 33] for high Weissenberg number. In their approach, the defect step consisted of a nonlinear iteration in which the Weissenberg number was replaced with an artificially reduced value, and the correction step sought to improve on the approximation found in the defect step.

Continuation methods [29, 1, 43] provide a means for stepping along solution manifolds for varying values of a problem parameter or group of parameters. In the context of viscoelastic flows, the behavior of computed solution manifolds in the Weissenberg parameter space is of great interest. This leads to considering the Weissenberg number \( \lambda \) as a prime candidate for continuation.

3.1 Simple Continuation

In some manner, continuation methods try to ensure that the initial iterate \( u^0 \) is within the radius of convergence of the solution to the discrete problem for the current parameter value. The basic approach to the nonlinear iteration (2.22) is to use \( u^0 = 0 \) as an initial iterate, regardless of the value of \( \lambda \) that is being solved for. A slightly more advanced approach is to first compute a solution for \( \lambda = 0 \), i.e., solve the corresponding linear Stokes problem to find \( u^0 \). This method should provide a better initial approximation than \( u^0 = 0 \) for \( \lambda > 0 \). However, as \( \lambda \) increases, nonlinear iterations will still fail to converge with a Stokes initial iterate.

Consider \( G(u) = G(u, \lambda) \) to be the nonlinear system of equations arising from the discrete problem (2.18) for a particular value of \( \lambda \). Let \( u_j \) be the solution computed by the nonlinear iteration (2.22) for \( \lambda = \lambda_0 \). Then \( u_0 \) serves as a “good” choice of initial iterate for the problem \( G(u, \lambda_1) \), where \( \lambda_1 = \lambda_0 + \Delta \lambda_0 \) for some \( \Delta \lambda_0 > 0 \). This process can be thought of as a simple continuation in \( \lambda \). Computation proceeds along the solution manifold in the \((\lambda, u)\) space by incrementing \( \lambda \) after each convergent nonlinear iteration, using the \( u \) computed at a point on the curve as the initial iterate for the next larger value of \( \lambda \). This process is described in Algorithm 3.1.

Algorithm 3.1 (Simple Continuation in \( \lambda \)) Let \((u_0, \lambda_0) \in \Pi^{h}_{bc} \times \mathbb{R} \) solve \( G(u, \lambda) = 0 \). For \( j = 0, 1, \ldots \), do

1. Determine the step length \( \Delta \lambda_j \).
2. Set \( u^{0}_{j+1} = u_j \) and \( \lambda_{j+1} = \lambda_j + \Delta \lambda_j \).
3. Solve \( G(u_{j+1}, \lambda_{j+1}) = 0 \) by the iteration: For \( i = 0, 1, \ldots \), solve for \((u^{i+1}_{j+1}, \lambda_{j+1}) \in \Pi^{h}_{bc} \times \mathbb{R} \) satisfying

\[
A((\sigma^{i+1}_{j+1}, u^{i+1}_{j+1}, (\tau, v)) + \lambda_{j+1} B^{h}(\sigma^{i+1}_{j+1}, u^{i+1}_{j+1}, \sigma^{i+1}_{j+1}, \tau) + \lambda_{j+1} \tilde{B}^{h}(u^{i+1}_{j+1}, \sigma^{i+1}_{j+1}, \tau) + \lambda_{j+1}(\sigma^{i+1}_{j+1}, \nabla u^{i+1}_{j+1}, \tau) + \lambda_{j+1}(g_0, \sigma^{i+1}_{j+1}, \nabla u^{i+1}_{j+1}, \tau) - (p^{i+1}_{j+1}, \nabla \cdot v) + (q, \nabla \cdot u^{i+1}_{j+1})
\]

where \( (\tau, v) \in \Pi^{h}. \)
The simple continuation approach can be effective in providing the Newton iteration for $G(u_j, \lambda_j) = 0$ with a good initial approximation. Figure 3.1 gives a plot of the solution manifold and a solution $(u_j, \lambda_j)$ along the curve. Note that in order for $u_j$ to be a satisfactory initial iterate for $\lambda_{j+1}$, the increment $\Delta \lambda_j$ must be chosen in such a way to ensure that $u_j$ is within the radius of convergence of the nonlinear operator (2.14)-(2.16).

A basic method for choosing the steplengths is to start with a moderate value for $\Delta \lambda_j$ and attempt the nonlinear iteration. If the iteration succeeds, then either set $\Delta \lambda_{j+1} := \Delta \lambda_j$ or $\Delta \lambda_{j+1} := \gamma \Delta \lambda_j$ for some $\gamma > 1$. If the iteration fails, then set $\Delta \lambda_{j+1} := \gamma \Delta \lambda_j$ for some $\gamma < 1$ and reattempt the nonlinear iteration. Once the steplength falls below a specified tolerance, the continuation procedure will terminate.

### 3.2 Natural Continuation

The simple continuation procedure may encounter difficulty if the point $(u_j, \lambda_j)$ is not sufficiently close to the point $(u_{j+1}, \lambda_{j+1})$. This can occur when attempting a step of too large in magnitude, or when the solution curve experiences large changes in $u$ for moderate changes in $\lambda$ (high slope). In addition, the simple continuation process can be inefficient if there is only moderate change in $u$ for a significant range of $\lambda$ (small slope).

Forming a predicted value based upon the slope of the solution curve at the point $(u_j, \lambda_j)$ can provide a better initial iterate for the subsequent nonlinear iteration. This slope can be found by
computing the quantity $\partial u/\partial \lambda$ at $\lambda_j$. Assume $G$ is continuously differentiable in $u$ and $\lambda$, and $u$ is continuously differentiable in $\lambda$. Then, for $G(u, \lambda) = 0$ at $(u_j, \lambda_j)$, we have from the chain rule

$$
\left( \frac{\partial G}{\partial u} \right)_{(u_j, \lambda_j)} \left( \frac{\partial u}{\partial \lambda} \right)_{\lambda_j} + \frac{\partial G}{\partial \lambda} \bigg|_{(u_j, \lambda_j)} = 0 \quad \text{in } \Omega,
$$

or

$$
\left\langle G_u(u_j, \lambda_j) \left( \frac{\partial u}{\partial \lambda} \right)_{\lambda_j}, v \right\rangle = -\left\langle G_u(u_j, \lambda_j), v \right\rangle, \quad \forall v \in \Pi^h. \quad (3.1)
$$

where $G_\lambda$ denotes the Fréchet derivative of $G$ with respect to the parameter $\lambda$. It is easy to see that

$$
\left\langle G_\lambda(u_j, \lambda_j), v \right\rangle = B^h(u_j, \sigma_j, \tau) + (g_a(\sigma_j, \nabla u_j), \tau) \quad \forall (\tau, v, q) = v \in \Pi^h.
$$

Once the tangent slope $\partial u/\partial \lambda$ at $\lambda_j$ has been found (denote this by $\hat{u}_j = (\sigma_j, \hat{u}_j, \hat{\tau}_j)$), then for some steplength $\Delta \lambda_j$, set

$$
u^0_{j+1} := u_j + \Delta \lambda_j \left( \frac{\partial u}{\partial \lambda} \right)_{\lambda_j} \quad = u_j + \Delta \lambda_j \hat{u}_j,
$$

as the initial iterate for $\lambda_{j+1} := \lambda_j + \Delta \lambda_j$. The standard nonlinear iteration (2.22) can then be performed to approximate $(u_{j+1}, \lambda_{j+1})$. This procedure is described in Algorithm 3.2, and one step of the method is illustrated in Figure 3.2.

**Algorithm 3.2 (Natural Continuation in $\lambda$)** Let $(u_0, \lambda_0) \in \Pi^{bc}_0 \times \mathbb{R}$ solve $G(u, \lambda) = 0$. For $j = 0, 1, \ldots$, do

1. Solve the linear problem (3.1) for $\hat{u}_j \in \Pi^h$ by:

$$
A((\hat{\sigma}, \hat{u}_j), (\tau, v)) + \lambda_j \bar{B}^h(\hat{u}_j, \sigma_j, \tau) + \lambda_j B^h(u_j, \sigma_j, \tau)
+ \lambda_j (g_a(\sigma_j, \nabla \hat{u}_j), \tau) + \lambda_j (g_a(\hat{\sigma}_j, \nabla u_j), \tau) - (\hat{\tau}_j, \nabla \cdot v) + (q, \nabla \cdot \hat{u}_j)
\quad = -B^h(u_j, \sigma_j, \tau) + (g_a(\sigma_j, \nabla u_j), \tau) \quad \forall (\tau, v, q) \in \Pi^h.
$$

2. Determine the step length $\Delta \lambda_j$.

3. Set $u^0_{j+1} = u_j + \Delta \lambda_j \hat{u}_j$ and $\lambda_{j+1} = \lambda_j + \Delta \lambda_j$.

4. Solve $G(u_{j+1}, \lambda_{j+1}) = 0$ by the iteration: For $i = 0, 1, \ldots$, solve for $(u^{i+1}_{j+1}, \lambda^{i+1}_{j+1}) \in \Pi^{bc}_0 \times \mathbb{R}$ satisfying

$$
A((\sigma^{i+1}_{j+1}, u^{i+1}_{j+1}), (\tau, v)) + \lambda_{j+1} B^h(u^{i+1}_{j+1}, \sigma^{i+1}_{j+1}, \tau) + \lambda_{j+1} \bar{B}^h(u^{i+1}_{j+1}, \sigma^{i+1}_{j+1}, \tau)
+ \lambda_{j+1} (g_a(\sigma^{i+1}_{j+1}, \nabla u^{i+1}_{j+1}), \tau) + \lambda_{j+1} (g_a(\sigma^{i+1}_{j+1}, \nabla u^{i+1}_{j+1}), \tau) - (\tau_{j+1}, \nabla \cdot v) + (q, \nabla \cdot u^{i+1}_{j+1})
\quad = \lambda_{j+1} \bar{B}^h(u^{i}_{j+1}, \sigma^{i}_{j+1}, \tau) + \lambda_{j+1} (g_a(\sigma^{i}_{j+1}, \nabla u^{i}_{j+1}), \tau) + 2\alpha(f, v) \quad \forall (\tau, v, q) \in \Pi^h.
$$

5. Go to Step 1.
This natural continuation procedure is sometimes described as Euler-Newton continuation [7], as the process of determining the predictor \( u^0_{j+1} \) is similar in nature to the forward Euler method for ODEs, and a Newton iteration is utilized to solve the nonlinear system of equations. Algorithm 3.2 is also sometimes referred to as first-order continuation. In the same manner, Algorithm 3.1 may be referred to as zeroth-order continuation.

Algorithm 3.2 differs from Algorithm 3.1 with the addition of solving a linear problem for \( \hat{u}_j \). It is expected that the \( u^0_{j+1} \) found by natural continuation is a better initial iterate than merely the solution \( u_j \) as used in the simple continuation algorithm. As a result, natural continuation is expected to be able to compute solutions for a larger range of Weissenberg number than simple continuation. In addition, the natural continuation approach may allow for a more aggressive steplength strategy than simple continuation. Den Heijer and Rheinboldt [15] derive some sophisticated algorithms for steplength selection for generalized natural continuation methods. Alternate predictor strategies may be used as well. For example, one may use a second-order Taylor series approximation to form \( u^0 \). Gunzburger and Peterson [25] showed that in the Navier-Stokes equations, for some cases the stepsize in Reynolds number may be chosen independently of the type of predictor used. A detailed discussion of steplength algorithms for viscoelastic flows is a topic to be analyzed in future work.

4 Turning Points and Pseudo-arclength Continuation

The natural continuation process can suffer difficulty at points along the solution manifold where the slope \( \partial u / \partial \lambda \) is undefined, i.e., at turning points or singular points along the solution curve. In this case, the Jacobian \( G_u \) will be singular and the linear problem (3.1) will not have a unique
solution.

4.1 Parametrization with respect to arc length

As the natural continuation algorithm will fail near turning points, a different continuation approach is needed. Keller [28, 29] proposed the reparametrization of the solution curve with an arclength (or arclength-like) parameter. The description of the method presented here is similar to those found in [5, 8, 13, 21, 36] and others. In a manner similar to natural continuation, first a tangent to the solution curve is found and a predicted value that lies on the tangent is computed. Then an iterative procedure attempts to reconcile the predicted value back to the solution curve. The arclength parametrization requires additional information in the form of an equation describing the arclength, but allows for a more robust iteration that can proceed beyond turning points.

Let \( \vartheta \) be a parameter describing the arc length of the solution manifold in the \((u, \lambda)\)-space. Then we have
\[
\left\| \frac{\partial u}{\partial \vartheta} \right\|^2 + \left( \frac{\partial \lambda}{\partial \vartheta} \right)^2 = 1,
\]
(4.1)
where the norm in (4.1) is appropriate for \( u \). For \( u \in \Pi^h \), this norm will be
\[
\| u \|^2 = \| \sigma \|^2_0 + \| u \|^2_1 + \| p \|^2_0.
\]

Let \( s \) represent an arclength-like parameter and consider \( u(s) \) and \( \lambda(s) \) to be functions of \( s \). Let \( \dot{u} = \partial u / \partial s \) and \( \dot{\lambda} = \partial \lambda / \partial s \), and assume \((u_j, \lambda_j) = (u(s_j), \lambda(s_j))\) is a solution of \( G(u, \lambda) = 0 \). A unit tangent vector \([\dot{u}_j, \dot{\lambda}_j]^T\) to the curve is computed by solving
\[
G_u \dot{u}_j + G_\lambda \dot{\lambda}_j = 0,
\]
(4.2)

for \( \dot{u}_j \). Then \( \dot{u}_j \) and \( \dot{\lambda}_j \) can be determined by
\[
\dot{\lambda}_j = \frac{\pm 1}{\sqrt{1 + |\dot{u}_j|^2}},
\]
(4.4)
\[
\dot{u}_j = \dot{\lambda}_j \dot{u}_j.
\]
(4.5)

Note that (4.4) and (4.5) imply \( |\dot{u}_j|^2 + |\dot{\lambda}_j|^2 = 1 \). The sign in (4.4) is chosen such that the angle between successive tangents on the curve is no less than 0 and no more than \( \pi/2 \). This results in a construction of the solution manifold that moves “forward” with respect to the parameter \( s \) [5, 13]. To determine the correct sign, given two solutions \((u_{j-1}, \lambda_{j-1})\) and \((u_j, \lambda_j)\), compute the quantity
\[
\langle \dot{u}_j, u_j - u_{j-1} \rangle + \dot{\lambda}_j (\lambda_j - \lambda_{j-1}) \equiv 0,
\]
(4.6)

where \( \langle \cdot, \cdot \rangle \) in (4.6) is the inner product that induces the norm in (4.1). If (4.6) is positive, then the choice of \((\dot{u}_j, \dot{\lambda}_j)\) means that the computation will proceed in the same direction as it did from \((u_{j-1}, \lambda_{j-1})\) to \((u_j, \lambda_j)\). If (4.6) is negative, the opposite sign should be chosen in (4.4).
Once the appropriate tangent direction has been chosen and the arclength parameter increment \( \Delta s_j = s_{j+1} - s_j \) is set, Euler predictors of

\[
\begin{align*}
\mathbf{u}^0_{j+1} &= \mathbf{u}_j + \Delta s_j \dot{\mathbf{u}}_j, \\
\lambda^0_{j+1} &= \lambda_j + \Delta s_j \dot{\lambda}_j
\end{align*}
\]

are chosen as the initial iterates for the nonlinear iteration. Note that, as \( \lambda_{j+1} = \lambda(s^j_{j+1}) \) is a function of the arclength parameter, its value may vary during the nonlinear iteration. This is precisely the flexibility that will allow computation beyond singular points.

To complete the nonlinear system of equations, \( \mathbf{G}(\mathbf{u}(s), \lambda(s)) = \mathbf{0} \) is augmented with a suitable arclength condition or constraint given by some \( N(\mathbf{u}(s), \lambda(s), s) = 0 \). This constraint is based upon the arclength equation (4.1). In practice, an approximation to (4.1), such as a linearization, is used. Thus \( s \) is referred to as a pseudo-arclength parameter instead of the actual arc length. The discussion of the details of the pseudo-arclength constraints has been postponed until Section 4.2, however it should be noted that \( N \) must be continuously differentiable in both \( \mathbf{u} \) and \( \lambda \). Thus, the nonlinear system of equations to be solved is

\[
\begin{bmatrix}
\mathbf{G}(\mathbf{u}(s), \lambda(s)) \\
N(\mathbf{u}(s), \lambda(s), s)
\end{bmatrix} = \mathbf{0},
\]

and this can be accomplished by a Newton iteration of the form

\[
\begin{bmatrix}
\mathbf{G}_u(\mathbf{u}^i_{j+1}, \lambda^i_{j+1}) \\
N_u(\mathbf{u}^i_{j+1}, \lambda^i_{j+1})
\end{bmatrix} \begin{bmatrix}
\delta \mathbf{u}^i \\
\delta \lambda^i
\end{bmatrix} = - \begin{bmatrix}
\mathbf{G}(\mathbf{u}^{i+1}_{j+1}, \lambda^{i+1}_{j+1}) \\
N(\mathbf{u}^{i+1}_{j+1}, \lambda^{i+1}_{j+1})
\end{bmatrix}.
\]

The linear system in (4.10) has been shown to be nonsingular at turning points of the solution manifold [28]. The above procedure is summarized in Algorithm 4.1.

**Algorithm 4.1 (Pseudo-arclength Continuation)** Let \( (\mathbf{u}_{-1}, \lambda_{-1}) \in \Pi_{bc}^h \times \mathbb{R} \) and \( (\mathbf{u}_0, \lambda_0) \in \Pi_{bc}^h \times \mathbb{R} \) solve \( \mathbf{G}(\mathbf{u}, \lambda) = \mathbf{0} \). For \( j = 0, 1, \ldots, \) do

1. Solve the linear problem (4.3) for \( \dot{\mathbf{u}}_j \in \Pi^h \) by:

\[
A((\hat{\sigma}_j, \dot{\mathbf{u}}_j), (\tau, \mathbf{v})) + \lambda_j \hat{B}^h(\dot{\mathbf{u}}_j, \sigma_j, \tau) + \lambda_j B^h(\mathbf{u}_j, \hat{\sigma}_j, \tau) \\
+ \lambda_j (g_a(\sigma_j, \nabla \dot{\mathbf{u}}_j), \tau) + \lambda_j (g_a(\hat{\sigma}_j, \nabla \mathbf{u}_j), \tau) - (\mathbf{p}_j, \nabla \cdot \mathbf{v}) + (\mathbf{q}, \nabla \cdot \dot{\mathbf{u}}_j)
\]

\[
= -B^h(\mathbf{u}_j, \sigma_j, \tau) - (g_a(\sigma_j, \nabla \mathbf{u}_j), \tau) \\
\forall (\tau, \mathbf{v}, \mathbf{q}) \in \Pi^h.
\]

2. Set \( \dot{\lambda}_j = (1 + \|\dot{\mathbf{u}}_j\|^2)^{-1/2} \) and \( \dot{\mathbf{u}}_j = \dot{\lambda} \mathbf{u}_j \).

3. Compute \( \omega := \langle \dot{\mathbf{u}}_j, \mathbf{u}_j - \mathbf{u}_{j-1} \rangle + \dot{\lambda}_j (\lambda_j - \lambda_{j-1}) \). If \( \omega < 0 \), set \( \dot{\lambda}_j = -(1 + \|\dot{\mathbf{u}}_j\|^2)^{-1/2} \) and \( \dot{\mathbf{u}}_j = \dot{\lambda} \mathbf{u}_j \).

4. Determine the pseudo-arclength stepsize \( \Delta s_j \).

5. Set \( \mathbf{u}^0_{j+1} = \mathbf{u}_j + \Delta s_j \dot{\mathbf{u}}_j \) and \( \lambda_{j+1} = \lambda_j + \Delta s_j \dot{\lambda}_j \).
6. Solve \( [G(u_{j+1}(s), \lambda_j(s)), N(u_{j+1}(s), \lambda_j(s), s)]^T = 0 \) by the iteration: For \( i = 0, 1, \ldots \), solve the linear system (4.10) for \((\delta u^i, \delta \lambda^i) \in \Pi^h \times \mathbb{R}\) satisfying

\[
\begin{bmatrix}
G_u(u_{j+1}^i, \lambda_j^i) \\
N_u(u_{j+1}^i, \lambda_j^i)
\end{bmatrix}
\begin{bmatrix}
\delta u^i \\
\delta \lambda^i
\end{bmatrix}
= - \begin{bmatrix}
G(u_{j+1}^i, \lambda_j^i) \\
N(u_{j+1}^i, \lambda_j^i)
\end{bmatrix},
\]

with \( u_{j+1}^{i+1} := u_{j+1}^i + \delta u^i \) and \( \lambda_{j+1}^{i+1} := \lambda_{j+1}^i + \delta \lambda^i \).

7. Go to Step 1.

The block Jacobians \( G_u \) and \( G_\lambda \) are as described in the natural continuation algorithm, while the vectors \( N_u \) and \( N_\lambda \) will be described in Section 4.2. Note that each pass of Algorithm 4.1 does not terminate with a predetermined value for \( \lambda_{j+1} \).

### 4.2 Pseudo-arclength constraints

As stated in Section 4.1, the pseudo-arclength constraint \( N(u(s), \lambda(s), s) \) must be continuously differentiable with respect to \( u \) and \( \lambda \). There are several choices for \( N \) that serve as good defining functions for \( s \). However, \( N \) must contain some characteristic of the solution that is to be measured along the solution curve.

#### 4.2.1 Orthogonal Constraint

Presented by Keller [28], the most commonly used pseudo-arclength constraint is derived from a linear approximation to (4.1). This condition is given by

\[
N_1(u(s), \lambda(s), s) := \langle \dot{u}_j, (u(s) - u(s_j)) \rangle + \dot{\lambda}_j (\lambda(s) - \lambda(s_j)) - (s - s_j) = 0,
\]

and it requires that successive solution iterates lie on the hyperplane orthogonal to the tangent vector \([\dot{u}_j, \dot{\lambda}_j]^T\) and at a distance of \( \Delta s_j \) from the solution \((u(s_j), \lambda(s_j)) = (u_j, \lambda_j)\). Figure 4.1 gives a graphical representation of the tangent \([\dot{u}_j, \dot{\lambda}_j]^T\) and the iterates satisfying \( N_1 \). Because of the nature of the geometry of the iterates, \( N_1 \) will be referred to as an “orthogonal” constraint. The orthogonality can be seen as follows. Note that from the structure of the predicted values (4.7) and (4.8), we have

\[
\dot{u}_j = \frac{1}{\Delta s_j} (u_{j+1}^0 - u_j) \quad \text{and} \quad \dot{\lambda}_j = \frac{1}{\Delta s_j} (\lambda_{j+1}^0 - \lambda_j).
\]

Now \( N_1(u_{j+1}^i, \lambda_{j+1}^i, s_{j+1}) \) and (4.12) yield the relation

\[
\langle (u_{j+1}^0 - u_j), (u_{j+1}^i - u_j) \rangle + (\lambda_{j+1}^i - \lambda_j) (\lambda_{j+1}^i - \lambda_j) = (\Delta s_j)^2,
\]

or,

\[
\langle (u_{j+1}^0 - u_j), (u_{j+1}^i - u_{j+1}^0) \rangle + \langle (u_{j+1}^0 - u_j), (u_{j+1}^0 - u_j) \rangle + (\lambda_{j+1}^i - \lambda_j) (\lambda_{j+1}^i - \lambda_{j+1}^0) + (\lambda_{j+1}^i - \lambda_j) (\lambda_{j+1}^0 - \lambda_j) = (\Delta s_j)^2.
\]

(4.13)
Figure 4.1: Illustration of the pseudo-arclength condition $N_1$. From Figure 4.1 it is easy to see that

$$(\Delta s_j)^2 = \|u_{j+1}^0 - u_j^0\|^2 + |\lambda_{j+1}^0 - \lambda_j^0|^2,$$  

and thus, subtracting (4.14) from (4.13) we have

$$\langle (u_{j+1}^0 - u_j^0), (\lambda_{j+1}^0 - u_{j+1}^0) \rangle + (\lambda_{j+1}^0 - \lambda_j^0) (\lambda_{j+1}^0 - \lambda_{j+1}^0) = 0,$$  

which implies that the vectors

$$\begin{bmatrix} u_{j+1}^0 - u_j^0 \\ \lambda_{j+1}^0 - \lambda_j^0 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} u_{j+1}^0 - u_{j+1}^0 \\ \lambda_{j+1}^0 - \lambda_{j+1}^0 \end{bmatrix}$$

are orthogonal. For implementation in Algorithm 4.1, the derivatives $N_{1,u}$ and $N_{1,\lambda}$ evaluated at $(u_{j+1}^i, \lambda_{j+1}^i, s)$ are necessary. These are given as

$$N_{1,u}(u_{j+1}^i, \lambda_{j+1}^i, s) = \dot{u}_j \quad \text{and} \quad N_{1,\lambda}(u_{j+1}^i, \lambda_{j+1}^i, s) = \dot{\lambda}_j.$$

### 4.2.2 Spherical Constraint

Another constraint presented by Keller [28] enforces successive iterates to be a particular distance from the previous solution. This constraint is of the form

$$N_2(u(s), \lambda(s), s) := \|u(s) - u(s_j)\|^2 + |\lambda(s) - \lambda(s_j)|^2 - (s - s_j)^2 = 0,$$  

and it requires that successive iterates lie on the sphere centered at $(u_j, \lambda_j)$ of radius $s - s_j$ and will thus be referred to as a “spherical” constraint. Figure 4.2 gives a graphical representation of the tangent $[\dot{u}_j, \dot{\lambda}_j]^T$ and the iterates satisfying $N_2$. The derivatives of $N_2$ are found to be

$$N_{2,u}(u_{j+1}^i, \lambda_{j+1}^i, s) = 2 (u_{j+1}^i - u_j^i),$$

$$N_{2,\lambda}(u_{j+1}^i, \lambda_{j+1}^i, s) = 2 |\lambda_{j+1}^i - \lambda_j^i|.$$
This constraint is less frequently used than $N_1$ in pseudo-arclength continuation descriptions and implementations [5, 7, 8, 16, 45, 36], perhaps due to the required recomputation of (4.17) and (4.18) for each linear iteration $i$. The orthogonal constraint requires no such recomputation as the derivatives $N_1, u$ and $N_1, \lambda$ do not vary with respect to $i$.

4.2.3 Constraint Weighting

Both (4.11) and (4.16) can be modified by employing a weighting parameter for the $u$ and $\lambda$ terms. This weighting modifies the geometry of successive iterates by either changing their angle with respect to the tangent or reshaping the ellipsoid on which the iterates lie. The weighting allows for a greater contribution from either the $u$ term or the $\lambda$ term, and enhances the flexibility of the pseudo-arclength continuation method. Specifically, for $0 \leq \theta \leq 2$, weighted constraints can be written as

$$
\tilde{N}_1(u(s), \lambda(s), s) = \theta \langle \dot{u}_j, (u(s) - u(s_j)) \rangle + (2 - \theta) \dot{\lambda}_j \lambda(s) - \lambda(s_j)) - (s - s_j),
$$

(4.19)

$$
\tilde{N}_2(u(s), \lambda(s), s) = \theta \|u(s) - u(s_j)\|^2 + (2 - \theta) |\lambda(s) - \lambda(s_j)|^2 - (s - s_j)^2.
$$

(4.20)

4.3 Combined Methods

The natural and pseudo-arclength continuation methods described above can be combined to create a powerful and flexible solution approach for large-scale problems, as was done by de Almeida and Derby [13] for the Navier-Stokes equations. The less-expensive natural continuation can be used by default and the procedure can switch to pseudo-arclength continuation when a region of high curvature of the solution manifold is encountered. To avoid the high cost of computing the curvature, the implementation described in [13] uses the angle between successive tangents on the curve to determine if the curvature threshold has been met.
5 Numerical experiments

To investigate the performance of the various continuation algorithms discussed in Sections 3 and 4, the algorithms have been implemented using the finite element software package FreeFem++ [26] in 2-d. Linear systems are solved using the UMFPACK solver [12]. As described in Section 2, continuous piecewise quadratic elements are used for velocity, continuous piecewise linears are used for pressure, and discontinuous piecewise linears are used for stress.

5.1 Four-to-one Contraction Flow

Numerical simulations of viscoelastic flow through a planar or axisymmetric contraction have been widely studied (see [2] or Chapter 8 of [39]). Here the case of planar flow through a contraction geometry with a ratio of 4:1 with respect to upstream and downstream channel widths is considered. The contraction angle is a fixed $3\pi/2$ and the channel lengths are sufficiently long to impose a fully developed Poiseuille flow in the inflow and outflow channels. The geometry of the computational domain is illustrated in Figure 5.1. The lower left corner of the domain corresponds to $x = y = 0$.

![Figure 5.1: Geometry of 4:1 contraction domain](image)

The factor $L$ is set to $1/4$ for these computations. On this domain the velocity boundary conditions are

$$
\mathbf{u} = \begin{bmatrix} \frac{1}{32}(1 - y^2) \\ 0 \end{bmatrix} \quad \text{on } \Gamma_{\text{in}}, \quad \mathbf{u} = \begin{bmatrix} 2 \left( \frac{1}{16} - y^2 \right) \\ 0 \end{bmatrix} \quad \text{on } \Gamma_{\text{out}}. \tag{5.1}
$$

Boundary conditions for $\sigma$ must be specified on the inflow boundary. From the constitutive equation (2.1) and the velocity conditions (5.1), for $u_{1,y} = \partial u_1/\partial y$, we have

$$
\sigma_{xx} = -\alpha \lambda (a + 1) u_{1,y}^2 \frac{1}{(a^2 - 1) \lambda^2 u_{1,y}^2 - 1}, \quad \sigma_{xy} = -\alpha u_{1,y} \frac{1}{(a^2 - 1) \lambda^2 u_{1,y}^2 - 1}, \quad \sigma_{yy} = -\alpha \lambda (a - 1) u_{1,y}^2 \frac{1}{(a^2 - 1) \lambda^2 u_{1,y}^2 - 1}, \quad \text{on } \Gamma_{\text{in}}.
$$

Symmetry conditions are imposed on the bottom of the computational domain. The parameter $\alpha$ is set to $8/9$. Computations were performed on two different meshes, M1 and M2. Table 5.1 lists the characteristics of the meshes.
\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|}
\hline
Mesh & \(\Delta x_{\text{min}}\) & \(\Delta y_{\text{min}}\) & No. of degrees of freedom \\
\hline
M1 & 0.25 & 0.0625 & 1880 \\
M2 & 0.125 & 0.03125 & 7321 \\
\hline
\end{tabular}
\caption{Mesh characteristics}
\end{table}

5.2 High Weissenberg Number Results

Of particular interest is the behavior of the solution manifold for large values of \(\lambda\). To investigate this, Algorithms 3.1 and 3.2 are run from a starting value of \(\lambda = 0\) and \(u^0 = 0\) on each computational mesh for selected values of the \(a\) parameter. Algorithm 4.1 (with the \(N_2\) constraint) requires two initial solutions, the solution above is used as well as the solution computed at \(\lambda = 1\). The initial step length in \(\lambda\) or \(s\) is set to 1, and upon failure of the nonlinear iteration, the steplength is reduced by half and the iteration reattempted. The continuation process terminates when the steplength falls below \(10^{-6}\). As a comparison, results are given from a “no continuation” approach of using \(u^0 = 0\) as the initial iterate for all values of \(\lambda\). A stopping criterion of

\[\|u^i - u^{i-1}\|_\infty \leq 10^{-8}\]

is used for the nonlinear iteration, and the nonlinear iteration was terminated if the stopping criterion had not been satisfied after 200 iterations.

Table 5.2 gives the maximum \(\lambda\) values for each solution approach. All of the continuation methods vastly increase the range of \(\lambda\) for which solutions can be computed over using \(u^0 = 0\) for all values of \(\lambda\). It is also observed that while simple and natural continuation produced similar high Weissenberg number limits, pseudo-arclength continuation was able to exceed those limits in all cases. However, the amounts by which pseudo-arclength continuation surpassed the other methods could be considered insignificant. Plots of the solution norm \(\|u_j\|\), where

\[\|u_j\|^2 = \|\sigma\|^2_0 + \|u\|^2_1 + \|p\|^2_0\]

computed from the pseudo-arclength continuation method for both meshes are shown in Figure 5.2.

In terms of the computational effort required by each algorithm, the simple and natural continuation

\[18\]
methods require less work than the pseudo-arclength method. However, while using the same steplength selection and reduction strategy, the pseudo-arclength continuation method requires far fewer steps and steplength reductions to reach its maximum value of $\lambda$. The number of steps and steplength reductions required for each method for computations with $a = 1$ on mesh M1 are reported in Table 5.3.

<table>
<thead>
<tr>
<th>Method</th>
<th>max $\lambda$</th>
<th>Total # of steps</th>
<th># of steplength reductions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
<td>11.1279</td>
<td>28</td>
<td>25</td>
</tr>
<tr>
<td>Natural</td>
<td>11.1279</td>
<td>28</td>
<td>25</td>
</tr>
<tr>
<td>Pseudo-arclength</td>
<td>11.1291</td>
<td>12</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 5.3: Number of steps and steplength reductions required for $a = 1$ on mesh M1.

The behavior of computed approximations for increasing Weissenberg number is of great interest, in particular for the Oldroyd-B constitutive model ($a = 1$). In Figure 5.3, the horizontal velocity of the fluid computed on mesh M2 along the line of symmetry (the center of the computational
domain) is shown for several values of $\lambda$, including the limiting value.

It is observed that, as the value of $\lambda$ increases, the magnitude of the velocity tends to increase just beyond the contraction of the domain, and fluctuates downstream. This may suggest that for highly elastic fluids, as the fluid enters the contraction channel, the elastic features of the fluid cause a speed-up/slow-down behavior. These observations have been seen previously in numerical experiments by Keunings and Crochet [31] for the Phan Thien-Tanner model. This behavior could lead to a loss of stability of solutions at higher values of $\lambda$; one hypothesis is that no steady-state solutions (only temporally unsteady solutions) exist beyond some critical value of $\lambda$. The same observation was also made during numerical experiments with a defect-correction method for viscoelasticity [17]. Trebotich, et al. [46] also observed this situation and suggest the wave-like behavior is related to the elastic Mach number.

As discussed in [42], there is a stress singularity at the reentrant corner of contraction flows. The inability of the nonlinear solver to converge for values of $\lambda$ greater than 9.3466 ($a = 1$, mesh M2) may be due to the steep stress gradients that develop near the reentrant corner. Figure 5.4 is a plot of the $\sigma_{11}$ component of the stress along the horizontal line $y = 0.24$ (the contraction occurs at $y = 0.25$). The plot shows that for increasing $\lambda$, the maximum value of $\sigma_{11}$ near the corner increases considerably. Figure 5.5 is a plot of the velocity field for $\lambda = 9.3466$ ($a = 1$, mesh M2).

### 5.3 Pseudo-Arclength Constraints

Also of interest is how the pseudo-arclength constraints $N_1$ and $N_2$, defined in (4.11) and (4.16) respectively, performed relative to each other. As mentioned in Section 4.2.2, $N_2$ is less often used in reported implementations, perhaps due to its need for recomputation at each iteration. One objective in studying $N_2$ is to learn if the geometry specified by the constraint has an impact on its convergence behavior and efficiency. As $N_2$ requires successive iterates to lie on a sphere of
Figure 5.4: Stress component $\sigma_{11}$ along a line near the reentrant corner for increasing $\lambda$.

Figure 5.5: Velocity field and speed contour, $\lambda = 9.3466$, $a = 1$, mesh M2.

radius $\Delta s_j$, centered at $(u_j, \lambda_j)$, successive iterates will sweep out an arc in the $(u, \lambda)$-space. This structure may be more efficient than the orthogonality required by $N_1$ in regions of high curvature of the solution manifold, in particular by attaining convergence with a larger stepsize than is possible for $N_1$.

To compare the performance of each constraint in a region of low or moderate curvature, computations were performed using both approaches for mesh M1 and $a = 1$, starting with solutions computed at $\lambda_{-1} = 0$ and $\lambda_0 = 1.0$. The initial steplength was set to $\Delta s_0 = 1.0$ and computations were stopped once the continuation process had reached $\lambda = 11.0$. Table 5.4 lists performance
statistics for both approaches during these computations. The performance of the two methods is

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Final λ</th>
<th># of Steps</th>
<th># of Steplength Reductions</th>
<th>Average # of Iterations/Step</th>
<th>Average CPU Seconds/Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_1$</td>
<td>11.04878019</td>
<td>14</td>
<td>3</td>
<td>26.7</td>
<td>113.7</td>
</tr>
<tr>
<td>$N_2$</td>
<td>11.07273844</td>
<td>14</td>
<td>2</td>
<td>26.7</td>
<td>114.5</td>
</tr>
</tbody>
</table>

Table 5.4: Performance statistics for pseudo-arclength constraints to pass $\lambda = 11.0$, $\Delta s_0 = 1.0$.

very comparable from $\lambda = 1.0$ to $\lambda = 11.0$ - both methods required, the same number of steps and, on average the same number of iterations per step. A slightly larger average CPU time for $N_2$ may reflect the recomputation of $N_{2u}$ and $N_{2,\lambda}$ required at each step.

To examine the behavior of the constraints in a region of high curvature, computations were performed for mesh $M1$ and $a = 1$, starting with solutions computed at $\lambda_{-1} = 11.1$ and $\lambda_0 = 11.124$. The initial steplength was set to $\Delta s_0 = 0.01$. Table 5.5 lists performance statistics for both approaches. From Table 5.5, it is evident that the spherical constraint $N_2$ is much more efficient than

<table>
<thead>
<tr>
<th>Constraint</th>
<th>Final λ</th>
<th># of Steps</th>
<th># of Steplength Reductions</th>
<th>Average # of Iterations/Step</th>
<th>Average CPU Seconds/Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_1$</td>
<td>11.12838171</td>
<td>25</td>
<td>8</td>
<td>189.3</td>
<td>785.5</td>
</tr>
<tr>
<td>$N_2$</td>
<td>11.12838919</td>
<td>1</td>
<td>0</td>
<td>100</td>
<td>419.4</td>
</tr>
</tbody>
</table>

Table 5.5: Performance statistics for pseudo-arclength constraints to pass $\lambda = 11.12838$, $\Delta s_0 = 0.01$.

the orthogonal constraint $N_1$, as $N_2$ required only one step of length 0.01 to exceed the target value, while $N_1$ required 25, with 8 additional convergence failures.

The same experiment was repeated with an initial steplength of $\Delta s_0 = 0.0001$. Table 5.6 lists performance statistics for each constraint. Again we see that far fewer steps, steplength reductions, and average iterations are required for $N_2$ to reach the same value of $\lambda$ (in this case, $\lambda = 11.12884$).

<table>
<thead>
<tr>
<th>Constraint</th>
<th># of Steps</th>
<th># of Steplength Reductions</th>
<th>Average # of Iterations/Step</th>
<th>Average CPU Seconds/Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_1$</td>
<td>270</td>
<td>7</td>
<td>177.3</td>
<td>1346.3</td>
</tr>
<tr>
<td>$N_2$</td>
<td>122</td>
<td>0</td>
<td>56.7</td>
<td>265.9</td>
</tr>
</tbody>
</table>

Table 5.6: Performance statistics for pseudo-arclength constraints, $\lambda = 11.12884$, $\Delta s_0 = 0.0001$.

One disadvantage encountered using the $N_2$ constraint was that a tendency to “sweep around” and skip over the “forward” solution curve and find a solution lying “behind” the current location. This tendency was most pronounced for values of $\lambda$ near the maximum that was obtained by $N_2$. To
correct this behavior, a check was implemented: upon determination of \((u_{j+1}, \lambda_{j+1})\), the two vectors

\[
\begin{bmatrix}
    u_{j+1} - u_j \\
    \lambda_{j+1} - \lambda_j
\end{bmatrix}
\quad \text{and} \quad
\begin{bmatrix}
    u_j - u_{j-1} \\
    \lambda_j - \lambda_{j-1}
\end{bmatrix}
\]

were compared to see if they pointed in opposite direction. If they did, then the iteration had swung back around on the curve. In that case, the point \((u_{j+1}, \lambda_{j+1})\) was discarded, \(\Delta s_j\) was reduced, and the iteration reattempted. The net result is that instances where the iteration wrapped back onto the curve were treated as convergence failures. With this correction in place, \(N_2\) was still a more efficient approach in regions of high curvature of the solution manifold.

Another difference in the approximations obtained using \(N_1\) and \(N_2\) was that, for mesh \(M_1\) and \(a = 1\), a turning point was computed with \(N_2\), while \(N_1\) terminated due to slow convergence before reaching the turning point. In Figure 5.6, the solution curves traced by both methods are shown for \(\lambda\) close to the maximum. The plot on the right is a closer look at the turn than the plot on the left. The turning point produces values of \(\lambda\) for which two approximations have been computed.

![Figure 5.6: Turning point computed by \(N_2\) for mesh \(M_1\), \(a = 1\), \(\Delta s_0 = 0.0001\).](image)

In particular, for \(\lambda = 11.1289745\), a solution was computed on both the lower branch and upper branch of the curve. The norm of the difference of the two solutions was computed to be 8.6638e-4. Figure 5.7 shows a plot of the residual velocity field when the lower branch solution is subtracted from the upper branch solution. Although the magnitude of the difference in the solutions is small relative to the full velocity field on either branch, the vortex observed in the difference between the solutions may indicate a temporal instability, as solutions on the upper branch beyond the turning point may be unsteady.

There are previous reports of the existence of a turning point in viscoelastic contraction flows [14, 20, 30, 48] and other problem geometries [4, 35, 47]. Some observations indicate that the solution behavior is similar to that of a loss of temporal stability. However, turning points observations were later dismissed as numerical artifact as they were not preserved under mesh refinement [39]. In the computations here, no other combination of mesh size and \(a\) parameter produced a turning point. Nevertheless, the existence of the turning point provides an important indicator of the instability of the discretized system of equations describing the flow.
Figure 5.7: Velocity field of upper solution branch minus lower solution branch, $\lambda = 11.1289745$, mesh M1, $a = 1$.

### 5.4 Summary

The continuation methods studied in this work provide some insight into the behavior of numerical approximations of steady Johnson-Segalman fluids at high Weissenberg number. For the choices of parameters studied here, a limiting Weissenberg number is found in each case, with or without the presence of large changes in the solutions for small changes in the Weissenberg number.

In particular, for the Oldroyd-B constitutive model, the solution manifold of the discretized system of equations is seen to exhibit large changes in solutions for small changes in Weissenberg number near the limiting value. Nonsmooth flow behavior and sharp stress gradients have been observed for increasing values of the Weissenberg number. A turning point was observed for the Oldroyd-B model on the coarsest mesh, however no turning point was observed otherwise.

It is also observed that the limiting Weissenberg values are similar across the range of methods examined. For pseudo-arclength continuation methods, a spherical constraint was seen to be more efficient than an orthogonal constraint in a region of high curvature of the solution manifold, while both constraints performed similarly in regions of low or moderate curvature. However, care must be taken to ensure that the spherical constraint does not skip over solutions and merely recompute previous points on the curve.

Further studies related to continuation methods for viscoelastic flows may include the analysis and development of steplength selection algorithms, and the examination of further pseudo-arclength constraint variations. Applications of multiparameter continuation, in which the inflow velocity, $a$, and $\alpha$ are included, as well as studies involving time-dependent formulations of the modeling equations, may provide further understanding of the behavior of the discretized system of equations.
Acknowledgement

The author wishes to thank Dr. John Burkardt, Dr. Vince Ervin, and Dr. Hyesuk Lee for their helpful comments and suggestions in relation to this work.

References


