Variable Stepsize, Variable Order Methods for Partial Differential Equations

by

V. P. DeCaria

B.S. in Mathematics, Millersville University, 2014

M.A. in Mathematics, University of Pittsburgh, 2017

Submitted to the Graduate Faculty of

the Kenneth P. Dietrich School of Arts and Sciences in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2019

UNIVERSITY OF PITTSBURGH

DIETRICH SCHOOL OF ARTS AND SCIENCES

This dissertation was presented

by

V. P. DeCaria

It was defended on

July 25th, 2019

and approved by

William Layton, Professor of Mathematics, University of Pittsburgh

Michael Neilan, Associate Professor of Mathematics, University of Pittsburgh

Catalin Trenchea, Associate Professor of Mathematics, University of Pittsburgh

Noel Walkington, Professor of Mathematics, Carnegie Mellon University

Dissertation Director: William Layton, Professor of Mathematics, University of Pittsburgh

Variable Stepsize, Variable Order Methods for Partial Differential Equations

V. P. DeCaria, PhD

University of Pittsburgh, 2019

Variable stepsize, variable order (VSVO) methods are the methods of choice to efficiently solve a wide range of ODEs with minimal work and assured accuracy. However, VSVO methods have limited impact in complex applications due to their computational complexity and the difficulty to implement them in legacy code. The goal of this dissertation is to develop, analyze, and test new VSVO methods that have the same computational complexity as their nonadaptive counterparts per step. Adaptivity allows these methods to take fewer steps, which makes them *globally* less complex.

Herein, we show how to use any backward differentiation formula (BDF) method as the basis for a VSVO method. Order adaptivity is achieved using an inexpensive post-processing technique known as time filtering. Time filters do not add to the asymptotic complexity of these methods, and allow for every possible order in the VSVO family to be computed for the same cost as one BDF solve. This approach yields new, nonstandard timestepping methods that are not in the literature, and we analyze their stability and accuracy herein.

Backward Euler (BDF1) and BDF2 are extremely ubiquitous methods, and this research demonstrates how they can be converted to order adaptive codes with only a few additional lines of code. We also develop a solver called Multiple Order One Solve Embedded 2,3,4 (MOOSE234). MOOSE234 is a VSVO method based on BDF3 that computes approximations of order two, three and four each step. All three approximations in MOOSE234 are at least $A(\alpha)$ stable, and the second order approximation is A stable.

While these methods are generally applicable to any system that is first order in time, we focus on issues pertaining to the Navier-Stokes equations. Our methods have been optimized for Navier-Stokes solvers, and we include linearly implicit and implicit-explicit (IMEX) versions.

Table of Contents

List of Tables

List of Figures

List of Algorithms

Preface

Professor William Layton is a fantastic advisor, and I would not have completed the program without his constant availability and encouragement. Thanks for always having your door open, and for giving me interesting problems during my time here!

I'd like to thank Professors Michael Neilan, Catalin Trenchea, and Noel Walkington for their guidance and input while serving on my dissertation committee.

Thank you to all my other collaborators, especially Sigal Gottlieb, Zachary Grant, Traian Iliescu, Yi Li, Michael McLaughlin, Michael Schneier and Haiyun Zhao.

Thanks to my parents, family and my two cats Gnocchi and Lulu for their unflagging support. Finally, a biggest thanks to my fiancée, Alicia Grundhoffer, for her support and for enduring the process alongside me. I am lucky to be engaged to such a wonderful person.

1.0 Introduction

This research addresses the specific need for time marching methods in a high performance computing setting to be adaptive in an efficient and easily implementable way. Due to the perceived computational and implementational difficulties of adaptive algorithms, many practitioners resort to simulations performed on uniform time grids with a static formula. While this is simple to implement, it is inefficient when the same small grid size that is required for sudden transient events is also used in uneventful periods. The situation is even worse when using a low order method that requires small stepsizes for accuracy requirements, or a high order method that requires small stepsizes for stability. Finally, there is the problem of how to pick the optimal global stepsize a'priori.

Attaining time accuracy in numerical simulations of many flows ranges from difficult to not yet possible. Contributing factors (each a topic of current research) include phenomenological models of unresolved processes omitting dynamic effects such as intermittent, Newtonian uncertainty implying a finite predictability horizon (requiring ensemble simulations), lack of computational resources for spacial meshes in the asymptotic regime (a central issue of large eddy simulation) and the low order, non-adaptive time discretizations commonly used in computational fluid dynamics (CFD). These issues are linked. For example, where model steady state represents statistical equilibrium, constant stepsize backward Euler is commonly used to time step to steady state.

This dissertation considers higher order, time adaptive discretizations in CFD. Approaches for time adaptivity include small time step, adaptive, explicit 1-step methods (common in gas dynamics) for short time simulations, and the goal oriented, adaptive space-time Galerkin Finite Element methods, e.g., [12]. In [59], Kay, Gresho, Griffiths and Silvester developed a time adaptive trapezoid rule method suitable for longer time simulations where errors are estimated using an AB2 implementation not requiring additional storage. The aim herein is similar to that of [59]. We give higher order, time adaptive methods for time accurate fluid flow simulations. Due to the richness of scales in higher Reynolds number flows, single order, variable timestep linear multistep methods are limited to A-stable (hence

second order) methods. We present a family of variable order methods up to order four, although the possibility exists to extend this to higher orders. These methods are modular, and are easily implementable on top of several popular existing methods such as the backward differentiation formula (BDF) methods. Switching the order of the method is trivial, and does not require additional linear or nonlinear solves, or function evaluations to do so.

Good, general purpose variable step, variable order (VSVO) methods such as Gear's method, or MATLAB's ode15s already exist. We stress that the methods herein address a specific difficulty of complex applications (like CFD) that are constrained by cognitive, memory, and computational complexity. While uncommon, it is certainly possible that general purpose VSVO methods can be adapted to CFD and that special purpose methods (such as herein) can be made more general. Heuristics are often used in production codes to further improve efficiency. These possibilities are interesting directions for exploration.

Our approach to adaptivity is novel, and yields new timestepping methods not equivalent to any standard methods in the literature. To adapt the order, we use time filters, which are an established tool to non-intrusively modify weather models to suppress spurious, nonphysical modes to improve predictions [8] with recent improvements [83],[84],[85], [66],[5], [50]. They have also been used to improve the physical fidelity of artificial compression methods [28]. They are simple to implement and inexpensive to compute.

This dissertation is organized as follows. In Chapter 2, we develop a VSVO method of orders one and two that is based on the backward Euler method for the Navier-Stokes equations. This is then generalized to a family of VSVO methods based on the BDF methods in Chapter 3. In Chapter 4, we revisit the specific method developed in Chapter 4 by analyzing an implicit-explicit version. In Chapter 5, we prove some new stability results for variable stepsize BDF2.

2.0 A Second Order VSVO Method Based on Implicit Euler

2.1 Introduction

The backward Euler time discretization is often used for complex, viscous flows due to its stability, rapid convergence to steady state solutions and simplicity to implement. However, it has poor time transient flow accuracy, [40], and can fail by overdamping a solution's dynamic behavior. For ODEs, adding a time filter to backward Euler, as in (2.3) below, yields two, embedded, A-stable approximations of first and second order accuracy, [46].

This chapter develops this idea into an adaptive time-step and adaptive order method for time accurate fluid flow simulation and gives an analysis of the resulting methods properties for constant time-steps. For constant time-steps, the resulting Algorithm 1 below involves adding only one extra line to a backward Euler code. The added filter step increases accuracy and adds negligible additional computational complexity, see Figure 1a and Figure 1b. Further, both time adaptivity and order adaptivity, presented in Section 2.2 and tested in Section 2.6, are easily implemented in a constant time step backward Euler code with $\mathcal{O}(20)$ added lines.

Thus, algorithms herein have two main features. First, they can be implemented in a legacy code based on backward Euler without modifying the legacy components. Second, both time step and method order can easily be adapted due to the embedded structure of the method. The variable step, variable order step (VSVO) method is presented in Section 2.2 and tested in Section 2.6.2.

Even for constant time-steps and constant order, the method herein does not reduce to a standard / named method. Algorithm 1 with Option B is (for constant order and time-step) equivalent to a member of the known, 2 parameter family of second order, 2-step, A-stable one leg methods (OLMs), see Algorithm 4, Section 2.3. Stability and velocity convergence of the (constant time step) general second order, two-step, A-stable method for the Navier-Stokes equations was proven already in [38], see equation (3.20) p. 185, and has been elaborated thereafter, e.g., [55].

Our *velocity* stability and error analysis, while necessary for completeness, parallels this previous work and is thus collected in Appendix A.1. On the other hand, Algorithm 1 with Option A does *not* fit within a general theory even for constant stepsize, and produces more accurate pressure approximations.

We begin by presenting the simplest, constant stepsize case to fix ideas. Consider the time dependent incompressible Navier-Stokes (NS) equations:

$$\begin{aligned} &u_t + u \cdot \nabla u - \nu \Delta u + \nabla p = f, \text{ and } \nabla \cdot u = 0 \text{ in } \Omega, \\ &u = 0 \text{ on } \partial\Omega, \text{ and } \int_\Omega p \, dx = 0, \\ &u(x,0) = u_0(x) \text{ in } \Omega. \end{aligned} \tag{2.1}$$

Here, $\Omega \subset \mathbb{R}^d (d{=}2{,}3)$ is a bounded polyhedral domain; $u : \Omega \times [0,T] \to \mathbb{R}^d$ is the fluid velocity; $p : \Omega \times (0,T] \to \mathbb{R}$ is the fluid pressure. The body force $f(x,t)$ is known, and ν is the kinematic viscosity of the fluid.

Suppressing the spacial discretization, the method calculates an intermediate velocity $\hat{u}^{n+1}$ using the backward Euler / fully implicit method. Time filters (requiring only two additional lines of code and not affecting the BE calculation) are applied to produce u^{n+1} and p^{n+1} follows:

Algorithm 1 (Constant $\triangle t$ BE plus time filter)**.** *With $u^* = \hat{u}^{n+1}$ (Implicit) or $u^* = 2u^n - u^{n-1}$ (Linearly-Implicit), Step 1: (Backward Euler)*

$$\begin{aligned} &\frac{\hat{u}^{n+1} - u^n}{\Delta t} + u^* \cdot \nabla \hat{u}^{n+1} - \nu \Delta \hat{u}^{n+1} + \nabla \hat{p}^{n+1} = f(t^{n+1}), \\ &\nabla \cdot \hat{u}^{n+1} = 0, \end{aligned} \tag{2.2}$$

Step 2: (Time Filter for velocity and pressure)

$$u^{n+1} = \hat{u}^{n+1} - \frac{1}{3}(\hat{u}^{n+1} - 2u^n + u^{n-1}) \tag{2.3}$$

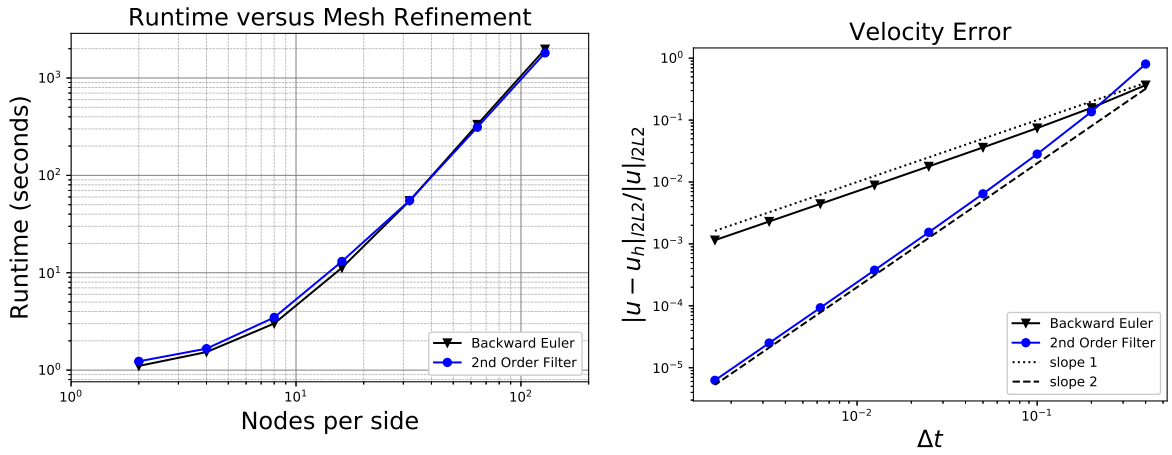
Option A: (No pressure filter)

$$p^{n+1} = \hat{p}^{n+1}.$$

Option B:

$$p^{n+1} = \hat{p}^{n+1} - \frac{1}{3}(\hat{p}^{n+1} - 2p^n + p^{n-1})$$

Algorithm 1A means Option A is used, and Algorithm 1B means Option B is used.



(a) (b)

Figure 1: The time filter does not add to the computational complexity (Fig. 1a), yet increases the method to second order (Fig. 1b).

Its implementation in a backward Euler code does not require additional function evaluations or solves, only a minor increase in floating point operations. Figure 1a presents a runtime comparison with and without the filter step. It is apparent that the added computational complexity of Step 2 is negligible. However, adding the time filter step has a profound impact on solution quality, see Figure 1b.

Herein, we give a velocity stability analysis in Section 2.4 and error analysis for constant timestep in Appendix A.1. Since (eliminating the intermediate step) the constant time-step method is equivalent to an A-stable, second order, two step method, its velocity analysis has only minor deviations from the analysis in [38] and [55]. We also give an analysis of the unfiltered pressure error, which does not have a parallel in [38] or [55].

The predicted (optimal) convergence rates are confirmed in numerical tests in Section 2.6. We prove the pressure approximation is stable and second order accurate provided only the velocity is filtered. The predicted second order pressure convergence, with or without filtering the pressure, is also confirmed in our tests, Figure 2.

The rest of the chapter is organized as follow. In Section 2.2, we give the full, self-adaptive VSVO algorithm for a general initial value problem. Section 2.3 introduces some

important mathematical notations and preliminaries necessary and analyze the method for the Navier-Stokes equations. In Section 2.4, we prove unconditional, nonlinear energy stability in Theorem 2.4.1. We analyze consistency error in Section 2.4.1. In A.1.2, we prove $\mathcal{O}(\Delta t^2)$ convergence for velocity, Theorem 2.4.2. The proof of the stability of the pressure is in Theorem 2.5.1 in Section 2.5.1. We prove second order accuracy for pressure in Section 2.5.2. Numerical tests are given in Section 2.6 to validate the theoretical predictions.

2.1.1 Related work

Time filters are primarily used to stabilize leapfrog time discretizations of weather models; see [74], [7], [83]. In [46] it was shown that the time filter used herein increases accuracy to second order, preserves A-stability, anti-diffuses the backward Euler approximation and yields an error estimator useful for time adaptivity. The analysis in [46] is an application of classical numerical ODE theory and does not extend to the Navier-Stokes equations.

For the constant time step case, our analysis is based on eliminating the intermediate approximation $\hat{u}^{n+1}$ and reducing the method to an equivalent two step, OLM (a twin of a linear multistep method). The velocity stability and convergence of the general A-stable OLM was analyzed for the NSE (semi-implicit, constant time step and without space discretization) in [38]. Thus, the constant time step, discrete *velocity* results herein follow from these results.

There is considerable previous work on analysis of multistep time discretizations of various PDEs, e.g. Crouzeix and Raviart [21]. Baker, Dougalis, and Karakashian [10] gave a long-time error analysis of the BDF methods for the NSE under a small data condition. (We stress that the method herein is *not a BDF method*.) The analysis of the method in Girault and Raviart [38] was extended to include spacial discretizations in [55]. The work in [55] also shows how to choose those parameters to improve accuracy in higher Reynolds number flows - a significant contribution by itself. Other interesting extensions include the work of Gevici [37], Emmrich [30], [31], Jiang [54], Ravindran [73] and [64].

2.2 The Adaptive VSVO Method

Section 2.6.2 tests both the constant time step method and the method with adaptive step and adaptive order. This section will present the algorithmic details of adapting both the order and time step based on estimates of local truncation errors based on established methods [42]. The constant time step Algorithm 1 involves adding one (Option A) or two (Option B) lines to a backward Euler FEM code. The full self adaptive VSVO Algorithm 2 below adds $\mathcal{O}(20)$ lines. We first give the method for the initial value problem

$$y\,'(t) = f(t, y(t)), \text{ for } t > 0 \text{ and } y(0) = y_0.$$

Denote the n^{th} time step size by Δt_n. Let $t^{n+1} = t^n + \Delta t_n$ and y^n an approximation to $y(t_n)$. The choice of filtering weights depend on $\omega_n := \Delta t_n / \Delta t_{n-1}$, Step 2 below. TOL is the user supplied tolerance on the allowable error per step.

Algorithm 2 (Variable stepsize, variable order 1 and 2 (VSVO-12))**.**

Step 1 : Backward Euler

$$\frac{y_{(1)}^{n+1} - y^n}{\Delta t_n} = f(t_{n+1}, y_{(1)}^{n+1})$$

Step 2 : Time Filter

$$y_{(2)}^{n+1} = y_{(1)}^{n+1} - \frac{\omega_n}{2\omega_n + 1}\left(y_{(1)}^{n+1} - (1+\omega_n)y^n + \omega_n y^{n-1})\right)$$

Step 3 : Estimate error in $y_{(1)}^{n+1}$ ***and*** $y_{(2)}^{n+1}$**.**

$$EST_1 = y_{(2)}^{n+1} - y_{(1)}^{n+1}$$

$$EST_2 = \frac{\omega_{n-1}\omega_n(1+\omega_n)}{1 + 2\omega_n + \omega_{n-1}\left(1 + 4\omega_n + 3\omega_n^2\right)}\Bigg(y_{(2)}^{n+1} - \frac{(1+\omega_n)(1+\omega_{n-1}(1+\omega_n))}{1+\omega_{n-1}}y^n + \omega_n(1+\omega_{n-1}(1+\omega_n))y^{n-1} - \frac{\omega_{n-1}^2\omega_n(1+\omega_n)}{1+\omega_{n-1}}y^{n-2}\Bigg).$$

Step 4 : Check if tolerance is satisfied.

If $\|EST_1\| < TOL$ or $\|EST_2\| < TOL$, at least one approximation is acceptable. Go to Step 5a. Otherwise, the step is rejected. Go to Step 5b.

Step 5a : At least one approximation is accepted. Pick an order and stepsize to proceed.

If both approximations are acceptable, set

$$\Delta t^{(1)} = 0.9\Delta t_n \left(\frac{TOL}{\|EST_1\|}\right)^{\frac{1}{2}}, \qquad \Delta t^{(2)} = 0.9\Delta t_n \left(\frac{TOL}{\|EST_2\|}\right)^{\frac{1}{3}}.$$

Set

$$i = \arg\max_{i\in\{1,2\}} \Delta t^{(i)}, \qquad \Delta t_{n+1} = \Delta t^{(i)}, \qquad t^{n+2} = t^{n+1} + \Delta t_{n+1}, \qquad y^{n+1} = y^{n+1}_{(i)}.$$

If only $y^{(1)}$ (resp. $y^{(2)}$) satisfies TOL, set $\Delta t_{n+1} = \Delta t^{(1)}$ (resp. $\Delta t^{(2)}$), and $y^{n+1} = y^{n+1}_{(1)}$ (resp. $y^{n+1}_{(2)}$). Proceed to Step 1 to calculate y^{n+2}.

Step 5b : Neither approximations satisfy TOL.

Set

$$\Delta t^{(1)} = 0.7\Delta t_n \left(\frac{TOL}{\|EST_1\|}\right)^{\frac{1}{2}}, \qquad \Delta t^{(2)} = 0.7\Delta t_n \left(\frac{TOL}{\|EST_2\|}\right)^{\frac{1}{3}}.$$

Set

$$i = \arg\max_{i\in\{1,2\}} \Delta t^{(i)}, \qquad \Delta t_n = \Delta t^{(i)}, \qquad t^{n+1} = t^n + \Delta t_n$$

Return to Step 1 to try again.

For clarity, we have not mentioned several standard features such as setting a maximum and minimum timestep, the maximum or minimum stepsize ratio, etc.

The implementation above computes an estimation of the local errors in Step 3. EST_1 provides an estimation for the local error of the first order approximation $y^{(1)}_{n+1}$ since $y^{(2)}_{n+1}$ is a second order approximation. For a justification of EST_2, see Appendix A.2.

Standard formulas, see e.g. [41], are used to pick the next stepsize in Steps 5a and 5b. Based on the previous Δt and the current error estimator, the formula estimates the largest next stepsize that can be taken by the method such that the tolerance will still be satisfied. Out of the approximations that satisfy the tolerance, the approximation which yielded the largest estimated Δt is chosen to advance the solution.

The numbers 0.9 in Step 5a and 0.7 in Step 5b are commonly used safety factors to make the next approximation more likely to be accepted since the exact optimal Δt is unknowable.

One more line is needed for linearly implicit methods. For linearly implicit methods the point of linearization must also have $\mathcal{O}(\Delta t^2)$ accuracy. For example, with $u^* = u^n$

$$\frac{u^{n+1} - u^n}{\Delta t_n} + u^* \cdot \nabla u^{n+1} + \frac{1}{2}(\nabla \cdot u^*)u^{n+1} + \nabla p^{n+1} - \nu \Delta u^{n+1} = f^{n+1} \ \& \ \nabla \cdot u^{n+1} = 0 \quad (2.4)$$

is a common first order linearly implicit method. The required modification in the BE step to ensure second order accuracy after the filter is to shift the point of linearization from $u^* = u^n$ to

$$u^* = \left(1 + \frac{\Delta t_n}{\Delta t_{n-1}}\right) u^n - \frac{\Delta t_n}{\Delta t_{n-1}} u^{n-1} = (1 + \omega_n)\, u^n - \omega_n u^{n-1}.$$

Other simplifications. The algorithm can be simplified if only the time-step is adapted (not order adaptive). It can be further simplified using *extrapolation* where the second order approximation is adapted based on EST_1 (pessimistic for the second order approximation).

2.3 Notations and Preliminaries

We introduce some notations and inequalities which will be used in later sections. $(\cdot, \cdot), \|\cdot\|$ denotes the $L^2(\Omega)$ inner product and norm. C will denote a generic, finite constant depending possibly on T, Ω and f. The velocity space X and pressure space Q are defined

$$X := H_0^1(\Omega)^d = \{v \in H^1(\Omega)^d : v|_{\partial\Omega} = 0\},$$
$$Q := L_0^2(\Omega)^d = \{q \in L^2(\Omega) : \int_\Omega q = 0\}.$$

The divergence free space V is given by

$$V := \{v \in X : (\nabla \cdot v, q) = 0 \quad \forall q \in Q)\}.$$

For measurable $v : [0,T] \to X$, define for, respectively, $1 \le p < \infty$ and $p = \infty$

$$||v||_{L^p(0,T;X)} = \left(\int_0^T ||v(t)||_X^p dt \right)^{1/p} \text{ and } ||v||_{L^\infty(0,T;X)} = ess \sup_{0 \le t \le T} ||v(t)||_X,$$
$$|||v|||_{p,k} = \left(\int_0^T \|v(t)\|_{H^k(\Omega)}^p \right)^{1/p} \text{ and } |||v|||_{\infty,k} = ess \sup_{0 \le t \le T} \|v(t)\|_{H^k(\Omega)}.$$

We define the skew-symmetrized nonlinear form:

$$B(u,v) := u \cdot \nabla v + \frac{1}{2}(\nabla \cdot u)v, \quad \forall\, u, v, w \in X,$$
$$b(u,v,w) := (B(u,v), w).$$

Lemma 2.3.1. *There exists $C > 0$ such that*

$$b(u,v,w) \le C\|\nabla u\|\|\nabla v\|\|\nabla w\|, \qquad \forall\, u, v, w \in X$$
$$b(u,v,w) \le C\|u\|\|v\|_2\|\nabla w\| \quad \forall u, w \in X, v \in X \cap H^2(\Omega).$$

Proof. See Lemma 2.1 on p. 12 of [78]. □

We use the following discrete Gronwall inequality found in [49, Lemma 5.1].

Lemma 2.3.2 (Discrete Gronwall Inequality)**.** *Let Δt, H, a_n, b_n, c_n, d_n (for integers $n \ge 0$) be non-negative numbers such that*

$$a_N + \Delta t \sum_{n=0}^{N} b_n \le \Delta t \sum_{n=0}^{N} d_n a_n + \Delta t \sum_{n=0}^{N} c_n + H, \quad \forall\, N \ge 0 \tag{2.5}$$

Suppose $\Delta t d_n < 1$ $\forall n$, then,

$$a_N + \Delta t \sum_{n=0}^{N} b_n \le \exp\Big(\Delta t \sum_{n=0}^{N} \frac{1}{1 - \Delta t d_n} \Big) \Big(\Delta t \sum_{n=0}^{N} c_n + H \Big), \quad \forall\, N \ge 0 \tag{2.6}$$

Multiplying (2.1) by test functions $(v, q) \in (X, Q)$ and integrating by parts gives

$$(u_t, v) + b(u, u, v) + \nu(\nabla u, \nabla v) - (p, \nabla \cdot v) + (\nabla \cdot u, q) = (f, v), \quad (\nabla \cdot u, q) = 0. \tag{2.7}$$

To discretize the above system in space, we choose conforming finite element spaces for velocity $X_h \subset X$ and pressure $Q_h \subset Q$ satisfying the discrete inf-sup condition and the following approximation properties:

$$\inf_{q_h \in Q_h} \sup_{v_h \in X_h} \frac{(q_h, \nabla \cdot v_h)}{\|q_h\| \|\nabla v_h\|} \geq \beta > 0.$$

We further assume that for each $u \in X \cap H^{k+1}(\Omega)^d$, and $p \in Q \cap H^{s+1}(\Omega)$ there exists $v_h \in X_h$ and $q_h \in Q_h$ such that

$$\begin{aligned} \|u - v_h\| &\leq C h^{k+1} \|u\|_{k+1}, \\ \|u - v_h\|_1 &\leq C h^{k} \|u\|_{k+1}, \\ \|p - q_h\| &\leq C h^{s+1} \|p\|_{s+1}. \end{aligned} \tag{2.8}$$

h denotes the maximum triangle diameter. Examples of finite element spaces satisfying these conditions are the MINI [6] and Taylor-Hood [80] elements. The discretely divergence free subspace $V_h \in X_h$ is defined

$$V_h := \{v_h \in X_h : (\nabla \cdot v_h, q_h) = 0 \quad \forall q_h \in Q_h\}.$$

The dual norms of X_h and V_h are

$$\|w\|_{X_h^*} := \sup_{v_h \in X_h} \frac{(w, v_h)}{\|\nabla v_h\|}, \|w\|_{V_h^*} := \sup_{v_h \in V_h} \frac{(w, v_h)}{\|\nabla v_h\|}.$$

The following Lemma from Galvin [36, p. 243] establishes the equivalence of these norms on V_h.

Lemma 2.3.3. *Suppose the discrete inf-sup condition holds, let $w \in V_h$, then there exists $C > 0$, independent of h, such that*

$$C\|w\|_{X_h^*} \leq \|w\|_{V_h^*} \leq \|w\|_{X_h^*}.$$

Lemma 2.3.3 is used to derive pressure error estimates with a technique shown in Fiordilino [34]. We will use the following, easily proven, algebraic identity.

Lemma 2.3.4. *The following identity holds.*

$$\left(\frac{3}{2}a - 2b + \frac{1}{2}c\right)\left(\frac{3}{2}a - b + \frac{1}{2}c\right) = \tag{2.9}$$
$$\left(\frac{a^2}{4} + \frac{(2a-b)^2}{4} + \frac{(a-b)^2}{4}\right) - \left(\frac{b^2}{4} + \frac{(2b-c)^2}{4} + \frac{(b-c)^2}{4}\right) + \frac{3}{4}(a-2b+c)^2$$

With the notation in place, we state the fully discrete method.

Algorithm 3 (Fully discrete method)**.** *Given $u_h^{n-1}, u_h^n \in X_h$ (and if necessary, given $p_h^{n-1}, p_h^n \in Q_h$), find $(\hat{u}_h^{n+1}, \hat{p}^{n+1}) \in (X_h, Q_h)$ satisfying*

$$\left(\frac{\hat{u}_h^{n+1} - u_h^n}{\Delta t_n}, v_h\right) + b(\hat{u}_h^{n+1}, \hat{u}_h^{n+1}, v_h) + \nu(\nabla \hat{u}_h^{n+1}, \nabla v_h) - (\hat{p}_h^{n+1}, \nabla \cdot v_h) = (f(t^{n+1}), v_h),$$
$$(\nabla \cdot \hat{u}^{n+1}, q_h) = 0. \tag{2.10}$$

for all $(v_h, q_h) \in (X_h, Q_h)$. Then compute

$$u_h^{n+1} = \hat{u}_h^{n+1} - \frac{\omega_n}{2\omega_n + 1}\left(\hat{u}_h^{n+1} - (1+\omega_n)u_h^n + \omega_n u_h^{n-1})\right).$$

Option A: (No pressure filter)

$$p_h^{n+1} = \hat{p}_h^{n+1}.$$

Option B:

$$p_h^{n+1} = \hat{p}_h^{n+1} - \frac{\omega_n}{2\omega_n + 1}\left(\hat{p}_h^{n+1} - (1+\omega_n)p_h^n + \omega_n p_h^{n-1})\right).$$

The constant time-step stability and error analysis works with the following equivalent formulation of the method. We stress that what follows is not the preferred implementation since it only yields one approximation, while Algorithm 3 gives the embedded approximations $\hat{u}_h^{n+1}$ and u_h^{n+1} and an error estimator.

Algorithm 4 (Constant time-step, equivalent method)**.** *Assume the time-step is constant. Given* (u_h^n, p_h^n) *and* (u_h^{n-1}, p_h^{n-1})*, find* (u_h^{n+1}, p_h^{n+1}) *such that for all* $(v_h, q_h) \in (X_h, Q_h)$*,*

Option A

$$\left(\frac{\frac{3}{2}u_h^{n+1} - 2u_h^n + \frac{1}{2}u_h^{n-1}}{\Delta t}, v_h\right) + b\left(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}, \frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}, v_h\right) \quad (2.11)$$
$$+\nu\left(\nabla\left(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}\right), \nabla v_h\right) - \left(\mathbf{p_h^{n+1}}, \nabla \cdot \mathbf{v_h}\right) = \left(f^{n+1}, v_h\right),$$
$$\left(\nabla \cdot \left(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}\right), q_h\right) = 0,$$

or Option B

$$\left(\frac{\frac{3}{2}u_h^{n+1} - 2u_h^n + \frac{1}{2}u_h^{n-1}}{\Delta t}, v_h\right) + b\left(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}, \frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}, v_h\right) \quad (2.12)$$
$$+\nu\left(\nabla\left(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}\right), \nabla v_h\right) - \left(\frac{\mathbf{3}}{\mathbf{2}}\mathbf{p_h^{n+1}} - \mathbf{p_h^n} + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{p_h^{n-1}}, \nabla \cdot \mathbf{v_h}\right) = \left(f^{n+1}, v_h\right),$$
$$\left(\nabla \cdot \left(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}\right), q_h\right) = 0.$$

The pressure is highlighted in bold, and is the only difference between the two above equations. The time difference term of the above equivalent method is that of BDF2 but the remainder is different. This is *not* the standard BDF2 method.

Proposition 2.3.1. *Algorithm 3A (respectively B) is equivalent Algorithm 4A (respectively B).*

Proof. We will just prove the case for Option A since the other case is similar. Let (u_h^{n+1}, p_h^{n+1}) be the solution to Algorithm 3. By linearity of the time filter, $(u_h^{n+1}, p_h^{n+1}) \in (X_h, Q_h)$. We can write $\hat{u}_h^{n+1}$ in terms of u_h^{n+1},u_h^n, and u_h^{n-1} as $\hat{u}^{n+1} = \frac{3}{2}u^{n+1} - u^n + \frac{1}{2}u^{n-1}$. Substitute this into (2.10). Then (u_h^{n+1}, p_h^{n+1}) satisfies equation (2.11). These steps can be reversed to show the converse. □

We next define the discrete kinetic energy, viscous and numerical dissipation terms that arise naturally from a G-stability analysis of Algorithm 4, *regardless* of whether Option A or B is used. The (constant time-step) discrete kinetic energy, discrete viscous energy dissipation rate and the numerical energy dissipation rate of Algorithm 4 are

$$\begin{aligned}
\text{discrete energy:} \quad & \mathcal{E}^n = \tfrac{1}{4}\left[\|u^n\|^2 + \|2u^n - u^{n-1}\|^2 + \|u^n - u^{n-1}\|^2\right], \\
\text{viscous dissipation:} \quad & \mathcal{D}^{n+1} = \Delta t\nu||\nabla\left(\tfrac{3}{2}u^{n+1} - u^n + \tfrac{1}{2}u^{n-1}\right)||^2, \\
\text{numerical dissipation:} \quad & \mathcal{Z}^{n+1} = \tfrac{3}{4}\|u^{n+1} - 2u^n + u^{n-1}\|^2.
\end{aligned}$$

Remark 2.3.1. *As $\Delta t \to 0$, $\mathcal{E}^n$ is consistent with the kinetic energy $\frac{1}{2}\|u\|^2$ and $\mathcal{D}^n$ is consistent with the instantaneous viscous dissipation $\nu\|\nabla u\|^2$. The numerical dissipation $\mathcal{Z}^{n+1} \approx \frac{3}{4}\Delta t^4\|u_{tt}(t^{n+1})\|^2$, is asymptotically smaller than the numerical dissipation of backward Euler, $\frac{1}{2}\Delta t^2\|u_t(t^{n+1})\|^2$.*

The method's kinetic energy differs from that of BDF2, which is (e.g. [62])

$$\mathcal{E}^n_{BDF2} = \frac{1}{4}\left[\|u^n\|^2 + \|2u^n - u^{n-1}\|^2\right]$$

due to the term $\|u^n - u^{n-1}\|^2$ in $\mathcal{E}^n$ which is a dispersive penalization of a discrete acceleration.

Define the interpolation and difference operators as follows

Definition 2.3.1. *The interpolation operator I and difference operator D are*

$$I[w^{n+1}] = \frac{3}{2}w^{n+1} - w^n + \frac{1}{2}w^{n-1} \quad \textit{and} \quad D[w^{n+1}] = \frac{3}{2}w^{n+1} - 2w^n + \frac{1}{2}w^{n-1}.$$

Formally, $I[w(t^{n+1})] = w(t^{n+1}) + \mathcal{O}(\Delta t^2)$, and $\frac{D[w(t^{n+1})]}{\Delta t} = w_t(t^{n+1}) + \mathcal{O}(\Delta t^2)$. This will be made more precise in the consistency error analysis in Section 2.4.1.

2.4 Stability and Error Analysis

We prove stability and error analysis of the *constant time-step* method. The velocity proofs parallel ones in [38] and [55] and are collected in Appendix A.1. The pressure analysis is presented in Section 2.5.1.

Theorem 2.4.1. *Assume the stepsize is constant. The following equality holds.*

$$\mathcal{E}^N + \sum_{n=1}^{N-1} \mathcal{D}^{n+1} + \sum_{n=1}^{N-1} \mathcal{Z}^{n+1} = \Delta t \sum_{n=1}^{N-1} (f, I[u_h^{n+1}]) + \mathcal{E}^1.$$

Proof. In Algorithm 4, set $v_h = \Delta t I[u_h^{n+1}]$ and $q_h = p_h^{n+1}$ for Option A, or $q_h = I[p_h^{n+1}]$ for Option B, and add.

$$(D[u_h^{n+1}], I[u_h^{n+1}]) + \mathcal{D}^{n+1} = \Delta t (f, I[u_h^{n+1}]). \tag{2.13}$$

By Lemma 2.3.4 and Definition 2.3.1,

$$(D[u_h^{n+1}], I[u_h^{n+1}]) = \mathcal{E}^{n+1} - \mathcal{E}^n + \mathcal{Z}^{n+1}.$$

Thus, (2.13) can be written

$$\mathcal{E}^{n+1} - \mathcal{E}^n + \mathcal{D}^{n+1} + \mathcal{Z}^{n+1} = \Delta t (f(t^{n+1}), I[u_h^{n+1}]).$$

Summing over n from 1 to $N-1$ yields the result. □

This result is for the time stepping method applied to the Navier-Stokes equations. More generally, the constant time-step method of Algorithm 1 is G-Stable, a fact that follows from the equivalence of A and G-Stability [23]. We calculate the G matrix explicitly below.

Corollary 2.4.1. *Assume the time-step is constant. Backward Euler followed by the time filter is G-Stable with G matrix*

$$G = \begin{bmatrix} \frac{3}{2} & -\frac{3}{4} \\ -\frac{3}{4} & \frac{1}{2} \end{bmatrix}.$$

Proof. Simply check that

$$[u^n, u^{n-1}] G \begin{bmatrix} u^n \\ u^{n-1} \end{bmatrix} = \frac{1}{4} \Big[|u^n|^2 + |2u^n - u^{n-1}|^2 + |u^n - u^{n-1}|^2 \Big].$$

□

2.4.1 Consistency error

By manipulating (2.7), we derive the consistency error. The true solution to (2.7) satisfies

$$\begin{aligned}&\left(\frac{D[u(t^{n+1})]}{\Delta t}, v_h\right) + b\left(I[u(t^{n+1})], I[u(t^{n+1})], v_h\right)\\&+ \nu\left(\nabla I[u(t^{n+1})], \nabla v_h\right) - \left(p(t^{n+1}), \nabla\cdot v_h\right)\\&= \left(\mathbf{f}^{n+1}, v_h\right) + \tau^{n+1}(u,p;v_h) \quad \forall v_h \in X_h.\end{aligned} \tag{2.14}$$

If Option A is used (pressure is unfiltered),

$$\tau^{n+1}(u,p;v_h) = \tau_A^{n+1}(u,p;v_h) := \left(\frac{D[u(t^{n+1})]}{\Delta t} - u_t(t^{n+1}), v_h\right) \tag{2.15}$$
$$+b\left(I[u(t^{n+1})], I[u(t^{n+1})], v_h\right) - b(u(t^{n+1}), u(t^{n+1}), v_h) + \nu\left(\nabla(I[u(t^{n+1})] - u(t^{n+1})), \nabla v_h\right)$$

If Option B is used (pressure is filtered),

$$\tau^{n+1}(u,p;v_h) = \tau_A^{n+1}(u,p;v_h) - \left(I[p(t^{n+1})] - p(t^{n+1}), \nabla\cdot v_h\right) \tag{2.16}$$

Thus, filtering the pressure introduces a term that, while still second order, adds to the consistency error. We believe this is why Option A performs better in the numerical tests, Figure 2. Furthermore, Option B requires assuming additional regularity for convergence, see Theorem 2.4.2.

The terms in the consistency error are bounded in the following lemma.

Lemma 2.4.1 (Consistency)**.** *For u, p sufficiently smooth, we have*

$$\left\|\frac{D[u(t^{n+1})]}{\Delta t} - u_t(t^{n+1})\right\|^2 \le \frac{6}{5}\Delta t^3 \int_{t^{n-1}}^{t^{n+1}} \|u_{ttt}\|^2 dt,$$

$$\left\|I[u(t^{n+1})] - u(t^{n+1})\right\|^2 \le \frac{4}{3}\Delta t^3 \int_{t^{n-1}}^{t^{n+1}} \|u_{tt}\|^2 dt. \tag{2.17}$$

$$\left\|I[p(t^{n+1})] - p(t^{n+1})\right\|^2 \le \frac{4}{3}\Delta t^3 \int_{t^{n-1}}^{t^{n+1}} \|p_{tt}\|^2 dt. \tag{2.18}$$

Proof. See Appendix A.1. □

2.4.2 Error estimates for the velocity

Next, we analyze the convergence of Algorithm 4 and give an error estimate for the velocity. Let $t^n = n\Delta t$. Denote the errors $\mathbf{e}_u^n = u(t^n) - u_h^n$ and $e_p^n = p(t^n) - p_h^n$.

Theorem 2.4.2. *Assume that the true solution (u, p) satisfies the following regularity*

$$\begin{aligned} &u \in L^\infty(0,T;(H^{k+1}\Omega))^d), \quad u_t \in L^2(0,T;(H^{k+1}\Omega))^d), \quad u_{tt} \in L^2(0,T;(H^1\Omega))^d), \\ &u_{ttt} \in L^2(0,T;(L^2\Omega))^d), \qquad p \in L^2(0,T;(H^{s+1}(\Omega))^d). \end{aligned} \tag{2.19}$$

Additionally for Option B, assume $p_{tt} \in L^2(0,T;(L^2(\Omega))^d$. For (u_h^{n+1}, p_h^{n+1}) satisfying (2.11), and for Δt sufficiently small, we have the following estimate

$$\begin{aligned} &\|e_u^N\|^2 + \|2e_u^N - e_u^{N-1}\|^2 + \|e_u^N - e_u^{N-1}\|^2 + \sum_{n=1}^{N-1} 3\|e_u^{n+1} - 2e_u^n + e_u^{n-1}\|^2 \\ &+ \nu\Delta t \sum_{n=1}^{N-1} \|\nabla I[e_u^{n+1}]\|^2 \le C\Big(h^{2k} + h^{2s+2} + \Delta t^4\Big) \end{aligned} \tag{2.20}$$

Proof. See Appendix A.1. □

2.5 Pressure Stability and Convergence

2.5.1 Stability of pressure

We introduce the following discrete norms

$$|||\omega|||_{\infty,k} := \max_{0\le n\le T/\Delta t} \|\omega^n\|_k, \quad |||\omega|||_{2,k} := \left(\sum_{n=0}^{T/\Delta t-1} \Delta t\|\omega^n\|_k^2\right)^{1/2}. \tag{2.21}$$

In this section, we prove that the pressure approximation is stable in $l^1(0,T;L^2(\Omega))$. We first give a corollary of Theorem 2.4.1 asserting the stability of the velocity approximation.

Corollary 2.5.1. *Suppose $f \in L^2(0,T;H^{-1}(\Omega)^d)$, then the velocity approximation satisfies*

$$\mathcal{E}^N + \frac{1}{2}\sum_{n=1}^{N-1} \mathcal{D}^{n+1} + \sum_{n=1}^{N-1} \mathcal{Z}^{n+1} \le \frac{1}{2\nu}|||f|||_{2,-1}^2 + \mathcal{E}^1.$$

Proof. Consider Theorem 2.4.1. Applying the Cauchy-Schwarz yields the inequality. □

We now prove the stability of the filtered pressure.

Theorem 2.5.1. *Suppose Corollary* 2.5.1 *holds, then the pressure approximation satisfies*

$$\begin{aligned} \beta\Delta t \sum_{n=1}^{N-1} \|p_h^{n+1}\| &\leq C \qquad \textit{for Option A,} \\ \beta\Delta t \sum_{n=1}^{N-1} \|I[p_h^{n+1}]\| &\leq C \quad \textit{for Option B.} \end{aligned} \tag{2.22}$$

Proof. We prove it for Option A, as the other case is similar. Isolating the discrete time derivative in (2.11), and restricting v_h to V_h yields

$$\begin{aligned} \left(\frac{D[u_h^{n+1}]}{\Delta t}, v_h\right) &= -b\left(I[u_h^{n+1}], I[u_h^{n+1}], v_h\right) \\ &- \nu\left(\nabla I[u_h^{n+1}], \nabla v_h\right) + \left(f^{n+1}, v_h\right) \quad \forall v_h \in V_h. \end{aligned} \tag{2.23}$$

The terms on the right hand side of (2.23) can be bounded as follows,

$$\begin{aligned} b\left(I[u_h^{n+1}], I[u_h^{n+1}], v_h\right) &\leq C\|\nabla I[u_h^{n+1}]\|\|\nabla I[u_h^{n+1}]\|\|\nabla v_h\|, \\ -\nu\left(\nabla I[u_h^{n+1}], \nabla v_h\right) &\leq \nu\|\nabla I[u_h^{n+1}]\|\|\nabla v_h\|, \\ \left(f^{n+1}, v_h\right) &\leq \|f^{n+1}\|_{-1}\|\nabla v_h\|. \end{aligned} \tag{2.24}$$

In equation (2.23), we can use the above estimates in (2.24), divide both sides by $\|\nabla v_h\|$, and take the supremum over $v_h \in V_h$. This gives

$$\left\|\frac{D[u_h^{n+1}]}{\Delta t}\right\|_{V_h^*} \leq (C\|\nabla I[u_h^{n+1}]\| + \nu)\|\nabla I[u_h^{n+1}]\| + \|f^{n+1}\|_{-1}. \tag{2.25}$$

Lemma 2.3.3 implies

$$\left\|\frac{D[u_h^{n+1}]}{\Delta t}\right\|_{X_h^*} \leq C\Big[(\|\nabla I[u_h^{n+1}]\| + 1)\|\nabla I[u_h^{n+1}]\| + \|f^{n+1}\|_{-1}\Big]. \tag{2.26}$$

Now consider Algorithm 4 again with $v_h \in X_h$. Isolating the pressure term in (2.11) and using the estimates from (2.24) yields

$$\begin{aligned} \left(p_h^{n+1}, \nabla \cdot v_h\right) &\leq \left(\frac{D[u_h^{n+1}]}{\Delta t}, v_h\right) \\ &+ C(\|\nabla I[u_h^{n+1}]\| + 1)\|\nabla I[u_h^{n+1}]\|\|\nabla v_h\| + \|f^{n+1}\|_{-1}\|\nabla v_h\|. \end{aligned} \tag{2.27}$$

Divide both sides by $\|\nabla v_h\|$, take supremum over $v_h \in X_h$ and use the discrete inf-sup condition and the results in (2.27). Then,

$$\beta\|p_h^{n+1}\| \leq C\Big[(\|\nabla I[u_h^{n+1}]\| + 1)\|\nabla I[u_h^{n+1}]\| + \|f^{n+1}\|_{-1}\Big]. \tag{2.28}$$

We then multiply by Δt, sum from $n = 1$ to $n = N - 1$, and apply Cauchy-Schwarz on the right hand side,

$$\beta\Delta t \sum_{n=1}^{N-1} \|p_h^{n+1}\| \leq C\Delta t\Big[(|||\nabla I[u_h]|||_{2,0} + 1)|||\nabla I[u_h]|||_{2,0} + \|f\|_{2,-1}\Big]. \tag{2.29}$$

Then using the result from velocity approximation, we get,

$$\beta\Delta t \sum_{n=1}^{N-1} \|p_h^{n+1}\| \leq C\Big[(|||f|||_{2,-1} + 1)|||f|||_{2,-1} + (\mathcal{E}^1 + 1)\mathcal{E}^1\Big]. \tag{2.30}$$

□

2.5.2 Error estimates for the pressure

We now prove convergence of the pressure approximation in $l^1(0, T; L^2(\Omega))$. Denote the pressure error as $e_p^n = p(t^n) - p_h^n$.

Theorem 2.5.2. *Let u, p satisfy the equation* (2.20)*. Let the assumption of regularity in Theorem* 2.4.2 *be satisfied. Then there exists a constant $C > 0$ such that*

$$\begin{aligned} \Delta t\beta \sum_{n=1}^{N-1} \|e_p^{n+1}\| &\leq C\Big(h^k + h^{s+1} + \Delta t^2\Big) \qquad \textit{for Option A,} \\ \Delta t\beta \sum_{n=1}^{N-1} \|I[e_p^{n+1}]\| &\leq C\Big(h^k + h^{s+1} + \Delta t^2\Big) \quad \textit{for Option B.} \end{aligned} \tag{2.31}$$

Proof. Again, we only prove this for Option A since the other case requires only slight modification. Using the equations (A.3) and (A.4) yields

$$\begin{aligned}\left(\frac{D[\phi_h^{n+1}]}{\Delta t}, v_h\right) &= -\left(\frac{D[\eta^{n+1}]}{\Delta t}, v_h\right) - b\left(I[e_u^{n+1}], I[u(t^{n+1})], v_h\right)\\ &- b\left(I[u_h^{n+1}], I[e_u^{n+1}], v_h\right) - \nu\left(\nabla I[e_u^{n+1}], \nabla v_h\right)\\ &+ \left(e_p^{n+1}, \nabla\cdot v_h\right) + \tau^{n+1}(u,p;v_h) \quad \forall v_h \in V_h.\end{aligned} \tag{2.32}$$

We bound the six individual terms on the right hand side of (2.32), term by term as follows:

$$\left(\frac{D[\eta^{n+1}]}{\Delta t}, v_h\right) \le C\Delta t^{-\frac{1}{2}}\|\eta_t\|_{L^2(t^{n-1},t^{n+1};L^2(\Omega))}\|\nabla v_h\|, \tag{2.33}$$

$$-b\left(I[e_u^{n+1}], I[u(t^{n+1})], v_h\right) \le C\|\nabla I[e_u^{n+1}]\|\|\nabla I[u(t^{n+1})]\|\|\nabla v_h\|, \tag{2.34}$$

$$-b\left(I[u_h^{n+1}], I[e_u^{n+1}], v_h\right) \le C\|\nabla(I[u_h^{n+1}])\|\|\nabla I[e_u^{n+1}]\|\|\nabla v_h\|, \tag{2.35}$$

$$-\nu\left(\nabla I[e_u^{n+1}], \nabla v_h\right) \le \nu\|\nabla I[e_u^{n+1}]\|\|\nabla v_h\|, \tag{2.36}$$

$$\left(p(t^{n+1}) - \lambda_h^{n+1}, \nabla\cdot v_h\right) \le C\|p(t^{n+1}) - \lambda_h^{n+1}\|\|\nabla v_h\|, \tag{2.37}$$

$$\begin{aligned}\tau^{n+1}(u,p;v_h) \le C\Delta t^{\frac{3}{2}}\Big(&\|u_{ttt}\|_{L^2(t^{n-1},t^{n+1};L^2(\Omega))} + \|\nabla u_{tt}\|_{L^2(t^{n-1},t^{n+1};L^2(\Omega))}\\ &+ \left(\|\nabla u(t^{n+1})\|^2 + \|\nabla I[u(t^{n+1})]\|\right)\|\nabla u_{tt}\|^2_{L^2(t^{n-1},t^{n+1};L^2(\Omega))}\Big)\|\nabla v_h\|.\end{aligned} \tag{2.38}$$

Considering equation (2.32) and Lemma 2.3.3 , using equations (2.33)-(2.38), dividing both sides by $\|\nabla v_h\|$ and taking a supremum over V_h gives

$$\begin{aligned}\left\|\frac{D[\phi_h^{n+1}]}{\Delta t}\right\|_{X_h^*} \le C\Big[&\Delta t^{-\frac{1}{2}}\|\eta_t\|_{L^2(t^n,t^{n+1};L^2(\Omega))}\|\\ &+ \|\nabla I[e_u^{n+1}]\|(\|\nabla I[u(t^{n+1})]\| + \|\nabla(I[u_h^{n+1}])\| + 1)\\ &+ \|p(t^{n+1}) - \lambda_h^{n+1}\| + \Delta t^{\frac{3}{2}}\Big(\|u_{ttt}\|_{L^2(t^{n-1},t^{n+1};L^2(\Omega))} + \|\nabla u_{tt}\|_{L^2(t^{n-1},t^{n+1};L^2(\Omega))}\\ &+ \|\nabla u\|^2_{L^4(t^{n-1},t^{n+1};L^2(\Omega))} + \|\nabla u_{tt}\|^2_{L^4(t^{n-1},t^{n+1};L^2(\Omega))}\Big)\Big].\end{aligned} \tag{2.39}$$

Reconsidering (2.32), we separate the pressure error term $e_p^{n+1} = (p(t^{n+1}) - \lambda_h^{n+1}) - (p_h^{n+1} - \lambda_h^{n+1})$ and rearrange, which yields for all $v_h \in X_h$

$$\begin{aligned}\Big(p_h^{n+1} - \lambda_h^{n+1}, \nabla \cdot v_h\Big) &= -\left(\frac{D[\eta^{n+1}]}{\Delta t}, v_h\right) - \left(\frac{D[\phi^{n+1}]}{\Delta t}, v_h\right) - b(I[e_u^{n+1}], I[u(t^{n+1})], v_h) \\ -b(I[u_h^{n+1}], I[e_u^{n+1}], v_h) &- \nu\left(\nabla I[e_u^{n+1}], \nabla v_h\right) + \left(p(t^{n+1}) - \lambda_h^{n+1}, \nabla \cdot v_h\right) + \tau^{n+1}(u, p; v_h).\end{aligned}$$

Consider the estimates in (2.33)-(2.39). Divide by $\|\nabla v_h\|$, take the supremum over $v_h \in X_h$ and use the discrete inf-sup condition to obtain,

$$\begin{aligned}\beta\|p_h^{n+1} - \lambda_h^{n+1}\| &\leq C\Big[\Delta t^{-\frac{1}{2}}\|\eta_t\|_{L^2(t^n, t^{n+1}; L^2(\Omega))} \\ &+ \|\nabla I[e_u^{n+1}]\|\Big(\|\nabla I[u(t^{n+1})]\| + \|\nabla(I[u_h^{n+1}])\| + 1\Big) \\ &+ \|p(t^{n+1}) - \lambda_h^{n+1}\| + \Delta t^{\frac{3}{2}}\Big(\|u_{ttt}\|_{L^2(t^{n-1}, t^{n+1}; L^2(\Omega))} + \|\nabla u_{tt}\|_{L^2(t^{n-1}, t^{n+1}; L^2(\Omega))} \\ &+ \|\nabla u\|^2_{L^4(t^{n-1}, t^{n+1}; L^2(\Omega))} + \|\nabla u_{tt}\|^2_{L^4(t^{n-1}, t^{n+1}; L^2(\Omega))}\Big)\Big].\end{aligned} \tag{2.40}$$

We multiply by Δt, sum from $n = 1$ to $n = N - 1$ and apply triangle inequality. This yields

$$\begin{aligned}\beta\Delta t\sum_{n=1}^{N-1}\|e_p^{n+1}\| &\leq C\Big[\Delta t^{-\frac{1}{2}}\|\eta_t\|_{L^2(0,T;L^2(\Omega))} \\ &+ \|\|p - \lambda_h\|\|_{2,0} + \|\|\nabla I[e_u^{n+1}]\|\|_{2,0} \\ &+ \Delta t^{\frac{5}{2}}\Big(\|u_{ttt}\|_{2,0} + \|\nabla u_{tt}\|_{2,0} + \|\|\nabla u\|\|^2_{4,0} + \|\nabla u_{tt}\|^2_{4,0}\Big)\Big].\end{aligned} \tag{2.41}$$

Results from the equations $(A.20)$ and $(A.23)$ give the bounds for the first two terms. Using error estimates of the velocity on the third term and taking infimum over X_h and Q_h yield the result. □

2.6 Numerical Tests

We verify second order convergence for the new method through an exact solution in Section 2.6.1. Visualizations of the flow and benchmark quantities gives additional support to the increased accuracy of the new method in Section 2.6.3. The tests used P_2/P_1 and P_3/P_2 elements. All computations were performed with FEniCS [4].

2.6.1 Taylor-Green vortex

We apply the backward Euler and the backward Euler plus filter for the 2D Taylor-Green vortex. This test problem is historically used to assess accuracy and convergence rates in CFD [15]. The exact solution is given by

$$u = e^{-2\nu t}(\cos x \sin y, -\sin x \cos y) \text{ and } p = -\frac{1}{4}e^{-4\nu t}(\cos 2x + \cos 2y).$$

To test time accuracy, we solve using P_3/P_2 elements on a uniform mesh of 250×250 squares divided into 2 triangle per square. We take a series of time steps for which the total error is expected to be dominated by the temporal error. Since the true solution decays exponentially, we tabulate and display relative errors. Figure 2 displays the relative errors for backward Euler, backward Euler plus filtering only the velocity (Algorithm 1A), and backward Euler plus filtering both the velocity and pressure (Algorithm 1B). Filtering the pressure does not affect the velocity solution, so the velocity error plot only shows two lines. The velocity error is $\mathcal{O}(\Delta t^2)$, as predicted, and significantly smaller than the backward Euler error. Thus, adding the filter step (1.3) reduces the velocity error substantially, Figure 2, at negligible cost, Figure 1b. The pressure error is $\mathcal{O}(\Delta t^2)$ when either both u and p are filtered, or only u is filtered, which is consistent with our theoretical analysis. Filtering only u has smaller pressure error since the pressure filter introduces an extra consistency error term, see (2.16).

2.6.2 Adaptive test

We test the time/order adaptive algorithm on a problem that showcases the superiority of the VSVO method over the constant stepsize, constant order method.

The Taylor-Green problem can be modified by replacing F with any differentiable function of t. With velocity and pressure defined as before, the required body force is

$$f(x, y, t) = (2\nu F(t) + F'(t))\langle \cos x \sin y, -\cos y \sin x \rangle.$$

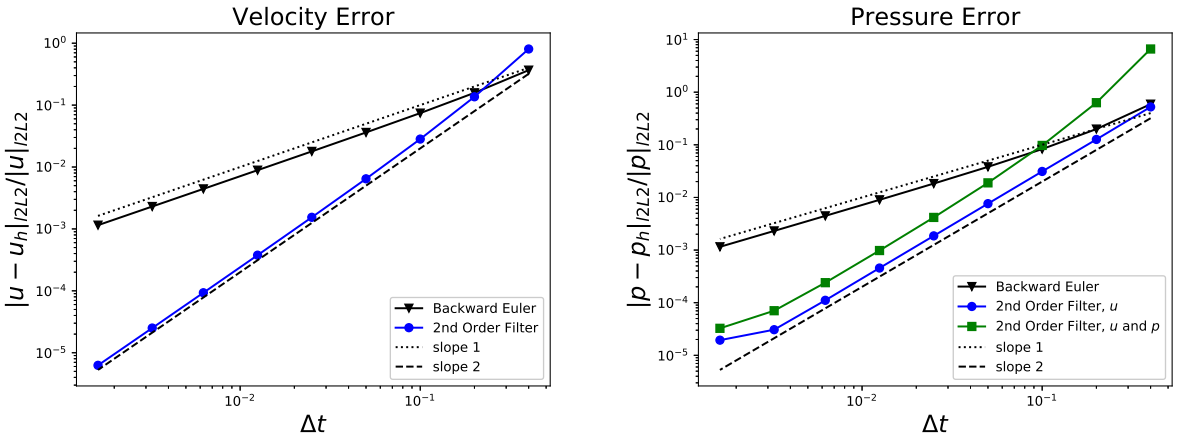


Figure 2: Convergence rates for the filtered quantities are second order as predicted. Filtering only the velocity produces the best pressure.

For $F(t)$, we construct a sharp transition function between 0 and 1. First, let

$$g(t) = \begin{cases} 0 & \text{if } t \leq 0 \\ \exp\left(-\frac{1}{(10t)^{10}}\right) & \text{if } t > 0 \end{cases}$$

This is a differentiable function, and $g(5) \equiv 1$ in double precision. Therefore, a differentiable (up to machine precision) function can be constructed with shifts and reflections of this function. This creates sections of flatness, and sections that rapidly change which require adaptivity to resolve efficiently. See Figure 3 for the evolution of $\|u\|$ with time. All tests were initialized at rest spaced at a constant interval of $\Delta t = 0.1$, 100 nodes per side of the square using P_2/P_1 elements, and with final time of 45.

Figure 3 compares two numerical solutions. One is from Algorithm 1 (second order - nonadaptive), and the other is from Algorithm 2 (VSVO-12). With $TOL = 10^{-3}$, the VSVO-12 method takes 342 steps, which comprises 254 accepted steps, and 88 rejected steps. The constant stepsize method which took 535 steps does not accurately capture the energetic jumps.

Figure 4 shows the relative l^2L^2 velocity errors versus steps taken of VSVO-12 for seven different TOLs, starting at 10^{-1}, and dividing by ten down to 10^{-7}. This is compared

with nonadaptive method (which has no rejected steps) sampled at several stepsizes. Both methods show second order convergence, but for smaller tolerances, VSVO-12 performs about 10^3 better than the nonadaptive method for the same amount of work.

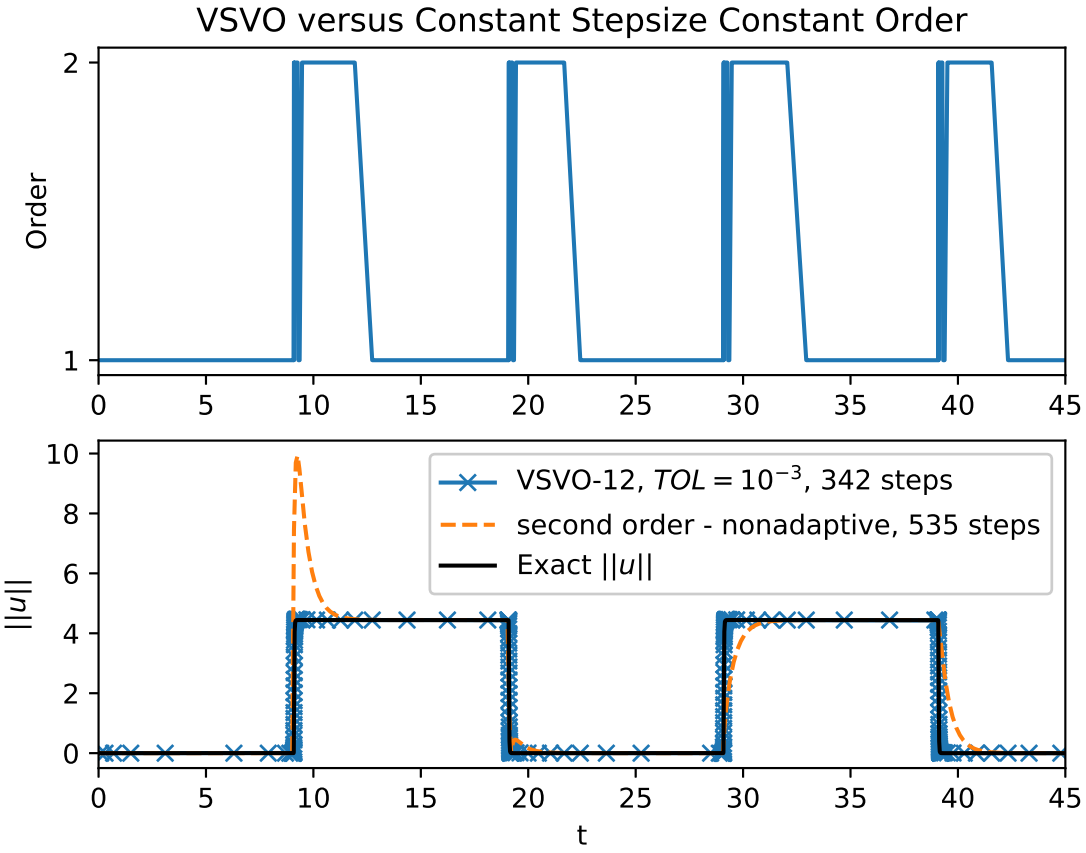


Figure 3: The nonadaptive second order method results in large overshoots and undershoots while requiring more work than the adaptive method.

2.6.3 Flow around a cylinder

We now use the benchmark problem of flow around a cylinder, originally proposed in [75], to test the improvement obtained using filters on flow quantities (drag, lift, and pressure drop) using values obtained via a DNS in [56] as a reference. This problem has also been

used as a benchmark in [71],[67],[12],[14] and others. Let $\nu = 10^{-3}$, $f \equiv 0$, $T_{final} = 8$, and

$$\Omega = \{(x, y) \,|\, 0 < x < 2.2,\ \ 0 < y < 0.41 \text{ and } (x - 0.2)^2 + (y - 0.2)^2 > 0.05^2\},$$

i.e., a channel with a cylindrical cutout. A parabolic velocity of $u = 0.41^{-2} \sin(\pi t/8)(6y(0.41 - y), 0)$ is prescribed at the left and right boundaries. We used a spatial discretization with 479026 degrees of freedom with 1000 vertices on the boundary of the cylinder. The mesh used $P2/P1$ elements, and was obtained by adaptive refinement from solving the steady solution with $u = 0.41^{-2}(6y(0.41 - y), 0)$ as inflow and outflow boundary conditions.

The correct behavior for this problem is that vortices shed off the cylinder as the inlet and outlet velocities increase. Figure 5 shows snapshots of the flow at $t = 6$ for five successively halved Δt's. The Backward Euler approximation shows no vortex shedding for $\Delta t = 0.04, 0.02$, and 0.01. The filtered method of Algorithm 1 shows the qualitatively correct behavior from $\Delta t = 0.02$ on. Clearly, higher order and less dissipative methods are necessary to see dynamics for modestly large Δt.

It was demonstrated in [56] that the backward Euler time discretization greatly under predicts lift except for very small step sizes. Figure 6 demonstrates that the time filter in Algorithm 1 corrects both the amplitude and phase error in the backward Euler approximation. Other quantities that were compared to reference values were the maximum drag $c_{d,\max}$, the time of max drag $t(c_{d,\max})$, time of maximum lift $t(c_{l,\max})$, and pressure drop across the cylinder at $t = 8$ are shown in Table 1.

The choice of whether or not to filter the pressure does not affect the velocity solution, the snapshots shown Figure 5 are the same for both choices. Table 1 shows that filtering u greatly improves the calculated flow quantities whether or not p is filtered.

2.7 Conclusion

Accurate and stable time discretization is important for obtaining correct flow predictions. The backward Euler time discretization is a stable but inaccurate method. We have

shown that for minimum extra programming effort, computational complexity, and storage, second order accuracy and unconditional stability can be obtained by adding a time filter. Due to the embedded and modular structure of the algorithm, both adaptive time-step and adaptive order are easily implemented in a code based on a backward Euler time discretization. Extension of the method and analysis to yet higher order time discretization is important as is exploring the effect of time filters on other methods possible for Step 1 of Algorithm 1. Analysis of the effect of time filters with moving and time dependent boundary conditions would also be a significant extension.

Acknowledgment

The research herein was partially supported by NSF grants DMS 1522267, 1817542 and CBET 160910.

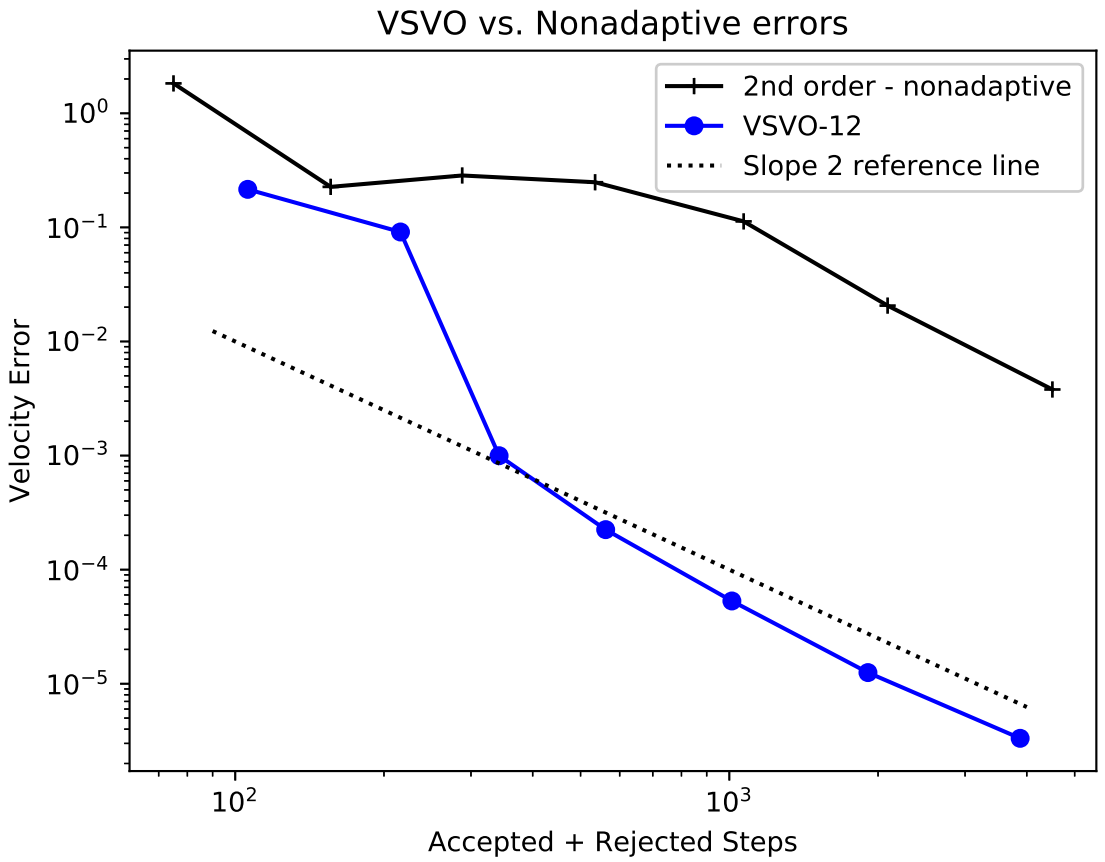
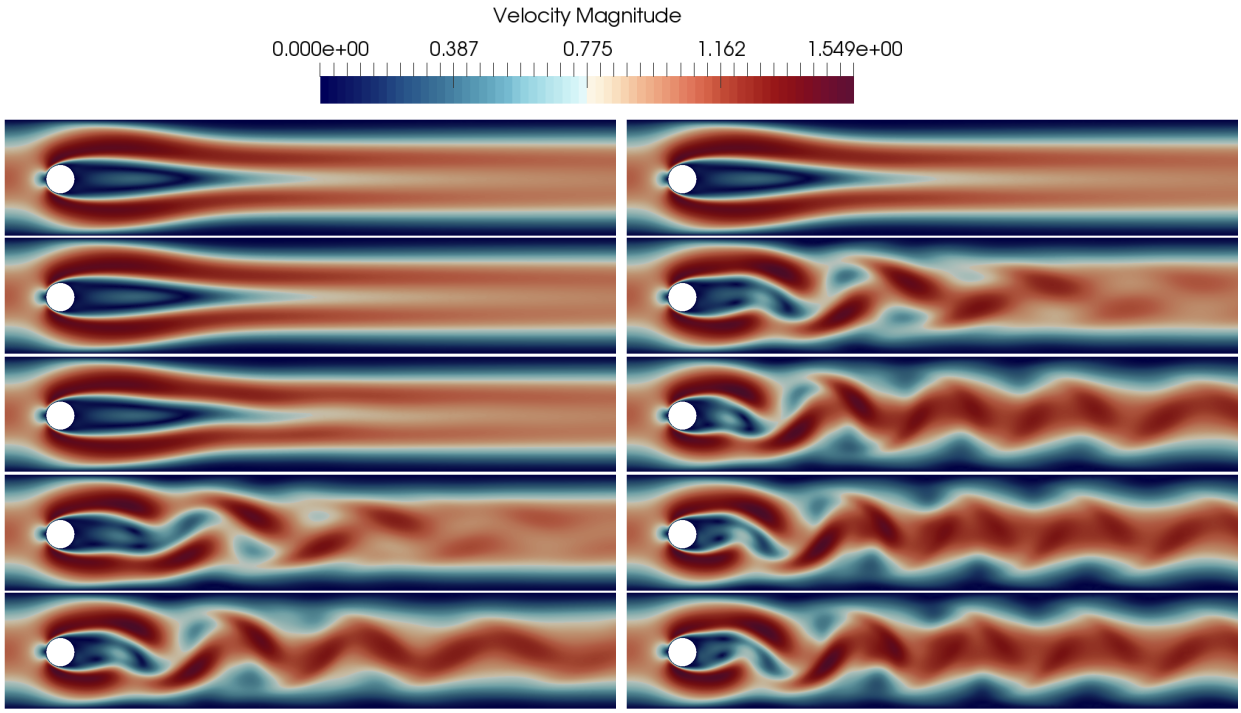


Figure 4: The VSVO-12 method performs three orders of magnitude better for the same amount of work compared to the nonadaptive 2nd order method for the test problem in Section 2.6.2. Each circle represents a different tolerance from $TOL = 10^{-1}$ to 10^{-7}.



(a) Backward Euler (b) Backward Euler Plus Filter

Figure 5: Flow snapshots at $t = 6$ with $\Delta t = 0.04$ (top), and Δt halving until $\Delta t = 0.0025$ (bottom). Backward Euler (left) destroys energy and suppresses oscillations, meaning that it can predict nearly steady state solutions when a time dependent one exists. The time filter (right) corrects this.

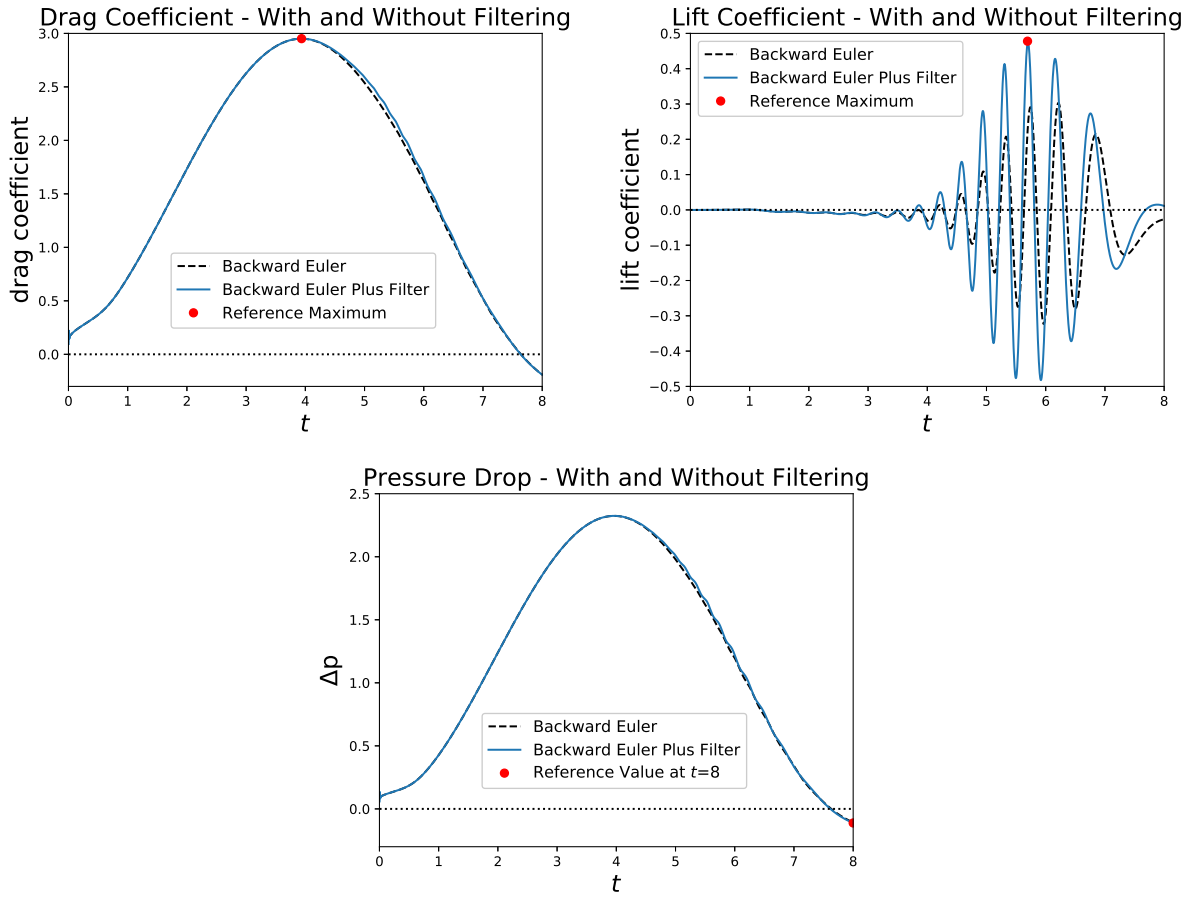


Figure 6: Lift of the Backward Euler solution and the filtered solution for $\Delta t = 0.0025$. The filtered solution correctly predicts both the time and magnitude of the maximum lift.

Table 1: Lift, drag, and pressure drop for cylinder problem

Δt	$t(c_{d,\max})$	$c_{d,\max}$	$t(c_{l,\max})$	$c_{l,\max}$	$\Delta p(8)$
Backward Euler					
0.04	3.92	2.95112558	0.88	0.00113655	-0.12675521
0.02	3.94	2.95064522	0.92	0.00117592	-0.12647232
0.01	3.93	2.95041574	7.17	0.02489640	-0.12433915
0.005	3.93	2.95031983	6.28	0.17588270	-0.10051423
0.0025	3.9325	2.95038901	6.215	0.30323034	-0.10699361
Backward Euler Plus Filter					
0.04	3.92	2.95021463	7.56	0.00438111	-0.12628328
0.02	3.94	2.95026781	6.14	0.20559211	-0.11146505
0.01	3.93	2.95060684	5.81	0.40244197	-0.09943203
0.005	3.935	2.95082513	5.72	0.46074771	-0.11111586
0.0025	3.935	2.95089028	5.7	0.47414096	-0.11193754
Backward Euler Plus Filter u and p					
0.04	3.92	2.95073993	7.52	0.00439864	-0.12642684
0.02	3.94	2.95039973	6.14	0.21101313	-0.11153593
0.01	3.93	2.95063962	5.81	0.40624697	-0.09945143
0.005	3.935	2.95083296	5.72	0.46192306	-0.11112049
0.0025	3.935	2.95089220	5.7	0.47444753	-0.11193859
Reference Values					
—	3.93625	2.950921575	5.693125	0.47795	-0.1116

3.0 Higher Order VSVO Methods

3.1 Introduction

In this chapter, we generalize the idea from Chapter 2 of using time filters on backward Euler to yield VSVO methods to the BDF family. We develop a general formula for the time filters that increase the order of any BDF method by one order. We also analyze and characterize the stability of these methods. In addition to time filters that increase the order of accuracy, we also develop stabilizing time filters.

While the end goal is the application to PDEs such as the Navier-Stokes equations, the theory is developed in a Numerical ODEs context, and will mostly use the notation of a more general IVP,

$$y'(t) = f(t, y(t)),$$
$$y(0) = y_0.$$

We recall the algorithm for constant stepsize,

Backward Euler	$\dfrac{y_1^{n+1} - y^n}{\Delta t} = f(t_{n+1}, y_1^{n+1})$
Time Filter	$y_2^{n+1} = y_1^{n+1} - \dfrac{1}{3}(y_1^{n+1} - 2y^n + y^{n-1})$

The approximation y_2^{n+1} is second order, A-stable and is implemented by adding one line to an existing backward Euler code. In this chapter, we answer the natural question: Can this be extended by more filters, using more y values, to produce an embedded family of methods and from that a VSVO algorithm of negligible additional complexity over backward Euler?

We prove in Theorem 3.3.2 an order barrier: backward Euler can only be made up to second order with linear time filters. Thus, to develop an embedded family of higher accuracy, we apply the time filter idea beginning with a third order method, BDF3. BDF methods are popular in CFD, e.g. BDF2 [35][2] [82], BDF3 [14][43], BDF4 [60], convex combinations of BDF methods [79] [72] [54], a predictor corrector scheme using BDF [70], VSVOBDF [47], and many others.

We develop time filters for every variable stepsize, variable coefficient BDFp method that increases their order of accuracy by one. We call this method Filtered BDFp+1 (FBDFp+1), and it is given in Algorithm 5. We then use the filtering idea to generate a VSVO algorithm of orders 2,3,4 with complexity comparable to BDF3. Starting with the approximation from BDF3, we apply a time filter to obtain FBDF4. We then develop a second filter, BDF3-Stab (3.22) to the BDF3 approximation to yield a second order A-stable (and G-stable) approximation (see Theorem 3.3.3).

Let the super script denote the order of the approximation, and the subscript denote the timestep. The resulting method, Multiple Order One Solve Embedded 234 (MOOSE234) for constant stepsize is

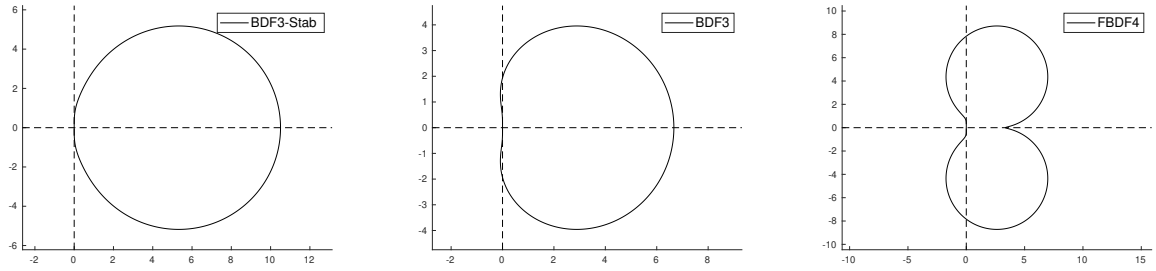
BDF3	$\dfrac{11y_3^{n+1} - 18y^n + 9y^{n-1} - 2y^{n-2}}{6\Delta t} = f(t_{n+1}, y_3^{n+1})$
BDF3-Stab	$y_2^{n+1} = y_3^{n+1} + \dfrac{9}{125}(y_3^{n+1} - 3y^n + 3y^{n-1} - y^{n-2})$
FBDF4	$y_4^{n+1} = y_3^{n+1} - \dfrac{3}{25}(y_3^{n+1} - 4y^n + 6y^{n-1} - 4y^{n-2} + y^{n-3})$
Error	$Est_2 = y_2^{n+1} - y_3^{n+1}$
Estimation	$Est_3 = y_4^{n+1} - y_3^{n+1}$

$$Est_4 = y_4^{n+1} - \frac{48}{25}y^n + \frac{36}{25}y^{n-1} - \frac{16}{25}y^{n-2} + \frac{3}{25}y^{n-3} - \frac{12}{25}\Delta t f(t_{n+1}, y_4^{n+1})$$

The algorithm for variable timestep is given in Section 3.4. This is an embedded family; its complexity is dominated by the nonlinear BDF3 solve. The remaining steps contribute negligible cost, require non-intrusive modifications to an existing BDF3 code, and, when used for a complex application, are single instruction, multiple data (SIMD) type, which adapt well to parallel architectures.

Each step computes a solution of different temporal orders of accuracy, so Est_2 gives an error estimator for y_2^{n+1}, and Est_3 gives an error estimator for y_3^{n+1}. Est_4 is an explicit approximation of the leading term of the local truncation error of y_4^{n+4}. Thus, the error estimates are embedded. Using standard strategies for timestep and order selection, the

family yields a VSVO algorithm, presented in Section 3.4. We demonstrate the time/order adaptivity of MOOSE234 on the Van der Pol test problem in Section 3.5.1. A linearly implicit version of MOOSE234 performed well on the incompressible Navier-Stokes equations, Sections 3.5.2-3.5.4.



(a) BDF3-Stab (b) BDF3 [$\alpha \approx 86.0^o$] (c) FBDF4 [$\alpha \approx 61.9^o$]

Figure 7: The regions of absolute stability are outside of the above curves. BDF3-Stab is A-Stable. The stability regions of BDF3 and FBDF4 still contain an infinite wedge of the left half of the plane, with α being the angle of the wedge measured from the real axis.

3.2 Preliminaries

The methods discussed herein will be shown to correspond to one-leg methods (OLMs), which require one function evaluation per timestep. OLMs are distinct from linear multistep methods (LMMs), have better nonlinear stability properties than their LMM counterparts and are well suited to time adaptation [24], [69], [53], [61]. Section 3.2.1 introduces the basic notation needed to state the methods in their full variable stepsize generality. Section 3.2.2 recalls a result of Dahlquist for OLMs needed herein.

3.2.1 Notation

Let $\Delta t_n = t_{n+1} - t_n$, and $\overline{\Delta t}_n$ be homogeneous of first degree in $\Delta t_n, \Delta t_{n+1}, ..., \Delta t_{n+m-1}$. An m step OLM is [24]

$$\sum_{i=0}^{m} \alpha_i y^{n+i} = \overline{\Delta t}_n f\left(\sum_{i=0}^{m} \beta_i t_{n+i}, \sum_{i=0}^{m} \beta_i y^{n+i}\right). \tag{3.1}$$

The corresponding LMM twin is

$$\sum_{i=0}^{m} \alpha_i y^{n+i} = \overline{\Delta t}_n \sum_{i=0}^{m} \beta_i f\left(t_{n+i}, y^{n+i}\right). \tag{3.2}$$

The first and second characteristic polynomials associated with both OLMs and LMMs are

$$\rho(r) = \sum_{j=0}^{m} \alpha_j r^j, \qquad \sigma(r) = \sum_{j=0}^{m} \beta_j r^j.$$

For variable timesteps, α_i and β_i depend on the stepsizes Δt_n, but we suppress additional subscripts for the sake of uncluttered notation. In the construction of the methods developed herein, it is convenient to absorb $\overline{\Delta t}_n$ into α_i through the change of variables $\bar{\alpha}_i = \alpha_i / \overline{\Delta t}_n$. Then (3.1) becomes

$$\sum_{i=0}^{m} \bar{\alpha}_i y^{n+i} = f\left(\sum_{i=0}^{m} \beta_i t_{n+i}, \sum_{i=0}^{m} \beta_i y^{n+i}\right). \tag{3.3}$$

The variable stepsize, variable coefficient BDF methods of order p (BDFp) using Newton interpolation are given as follows. Using the notation of [47, pg. 155], let δ^j be the jth order backward divided difference at t_{n+m},

$$\delta^j \phi = \phi[t_{n+m}, t_{n+m-1}, \cdots, t_{n+m-j}].$$

Let $m \geq p$. Then BDFp can be written

$$\sum_{j=m-p}^{m} \bar{\alpha}_j^{(p)} y^{n+j} := \sum_{j=1}^{p} \left[\prod_{i=1}^{j-1} (t_{n+m} - t_{n+m-i})\right] \delta^j y = f(t_{n+m}, y^{n+m})$$

where the $\bar{\alpha}_j^{(p)} s$ are given implicitly by the equation. The coefficients for variable stepsize BDF up to order four can also been written explicitly in terms of stepsize ratios in [81]. The divided differences can be expanded as

$$\delta^j \phi = \sum_{i=m-j}^{m} c_i^{(j)} \phi^{n+i} \tag{3.4}$$

with a procedure to generate the $c_i^{(j)}$s given in the appendix.

If the approximation at t_{n+m} is an intermediate approximation, we will denote it by $\tilde{y}^{n+m}$ to indicate a generic intermediate approximation, or y_p^{n+m} to indicate an approximation of order p. The intermediate divided differences are defined

$$\delta^j \tilde{y} = c_m^{(j)} \tilde{y}^{n+m} + \sum_{i=m-j}^{m-1} c_i^{(j)} y^{n+i}, \qquad \delta^j y^p = c_m^{(j)} y_p^{n+m} + \sum_{i=m-j}^{m-1} c_i^{(j)} y^{n+i}$$

Example. Recalling the variable stepsize backward Euler plus time filter in [46, pg. 307], we will rewrite it in terms of divided differences. With $\omega = \frac{\Delta t_{n+1}}{\Delta t_n}$ it is as follows,

$$\text{Backward Euler} \qquad \frac{y_1^{n+2} - y^{n+1}}{\Delta t_{n+1}} = f(t_{n+2}, y_1^{n+2})$$

$$\text{Time Filter} \qquad y^{n+2} = y_1^{n+2} - \frac{\omega(1+\omega)}{1+2\omega}\left(\frac{1}{1+\omega} y_1^{n+2} - y^{n+1} + \frac{\omega}{1+\omega} y^n\right)$$

Through algebraic manipulation, this can be written with divided differences,

$$\text{Backward Euler} \qquad \delta^1 y_1 = f(t_{n+2}, y_1^{n+2})$$

$$\text{Time Filter} \qquad y^{n+2} = y_1^{n+2} - \left(\frac{\Delta t_{n+1}}{\frac{1}{\Delta t_{n+1}} + \frac{1}{\Delta t_{n+1} + \Delta t_n}}\right) \delta^2 y_1 \tag{3.5}$$

$$= y_1^{n+2} - \left(\frac{\Delta t_{n+1}}{\frac{1}{\Delta t_{n+1}} + \frac{1}{\Delta t_{n+1} + \Delta t_n}}\right)\left(\frac{\frac{y_1^{n+2} - y^{n+1}}{\Delta t_{n+1}} - \frac{y^{n+1} - y^n}{\Delta t_n}}{\Delta t_{n+1} + \Delta t_n}\right)$$

The rearrangement (3.5) is a special case of Algorithm 5 with $p = 1$.

The coefficients for higher order differences are lengthy to write out. While they may be hard-coded into a program, they are easily computed with recursion [47, pg. 175] or nested loops (see Appendix B.2).

3.2.2 Classical error results

Define the local truncation error (LTE)

$$\begin{aligned}\varepsilon_n &= \left(\overline{\Delta t}_n\right)^{-1} \sum_{i=0}^{m} \alpha_i y(t_{n+i}) - f\left(\sum_{i=0}^{m} \beta_i t_{n+i}, \sum_{i=0}^{m} \beta_i y(t_{n+i})\right) \\ &= \sum_{i=0}^{m} \bar{\alpha}_i y(t_{n+i}) - f\left(\sum_{i=0}^{m} \beta_i t_{n+i}, \sum_{i=0}^{m} \beta_i y(t_{n+i})\right)\end{aligned}$$

A characterization of the accuracy of the OLM can be stated in terms of the differentiation and interpolation error operators [25] [26].

Definition 3.2.1 (Differentiation and Interpolation Error)**.** *The differentiation error L_d and interpolation error L_i operators are defined*

$$(L_d\phi)\left(\sum_{i=0}^{m} \beta_i t_{n+i}\right) = \sum_{i=0}^{m} \bar{\alpha}_i \phi(t_{n+i}) - \phi'\left(\sum_{i=0}^{m} \beta_i t_{n+i}\right) \tag{3.6}$$

$$(L_i\phi)\left(\sum_{i=0}^{m} \beta_i t_{n+i}\right) = \sum_{i=0}^{m} \beta_i \phi(t_{n+i}) - \phi\left(\sum_{i=0}^{m} \beta_i t_{n+i}\right). \tag{3.7}$$

A Taylor series calculation gives

$$\textit{Leading term of } \varepsilon_n = \left(L_d y - f_y\left(\sum_{i=0}^{m} \beta_i t_{n+i}, y\left(\sum_{i=0}^{m} \beta_i t_{n+i}\right)\right) \cdot L_i y\right)\left(\sum_{i=0}^{m} \beta_i t_{n+i}\right). \tag{3.8}$$

Definition 3.2.2 (Proposed in [25, pg. 8])**.** *Let p_d be the largest integer such 3.6 is zero for all polynomials of degree p_d, and p_i the largest integer such that 3.7 is zero for all polynomials of degree p_i. The order of the one leg method is* $\min(p_d, p_i + 1)$.

3.3 Embedding BDFp in a New Family

We now develop a time filter that increases the order of consistency of BDFp by one. Let $m = p + 1$, and consider the following method.

Algorithm 5 (Filtered BDFp+1 (FBDFp+1))**.** *Given*

$$\{y^n, y^{n+1}, ..., y^{n+m-1}\},$$

find y^{n+m} *satisfying*

$$\sum_{j=1}^{p} \left[\prod_{i=1}^{j-1} (t_{n+m} - t_{n+m-i}) \right] \delta^j y^p = f(t_{n+m}, y_p^{n+m}). \tag{3.9}$$

$$\eta^{(p+1)} = \frac{\prod_{i=1}^{p} (t_{n+m} - t_{n+m-i})}{\sum_{j=1}^{p+1} (t_{n+m} - t_{n+m-j})^{-1}}. \tag{3.10}$$

$$y^{n+m} = y_p^{n+m} - \eta^{(p+1)} \delta^{p+1} y^p. \tag{3.11}$$

It will be shown in Theorem 3.3.1 that this method has consistency error of order $p + 1$. The proof requires reducing the above steps to an OLM. Specifically, the OLM approximates y' the same as BDF$p + 1$, but not $f(y)$.

Algorithm 6 (Equivalent one-leg BDF filter)**.** *Given*

$$\{y^n, y^{n+1}, ..., y^{n+m-1}\},$$

and $\eta^{(p+1)}$ *as in* (3.10)*, find* y^{n+m} *satisfying*

$$\sum_{j=1}^{p+1} \left[\prod_{i=1}^{j-1} (t_{n+m} - t_{n+m-i}) \right] \delta^j y = f\left(t_{n+m}, y^{n+m} + \frac{\eta^{(p+1)}}{1 - \eta^{(p+1)} c_m^{(p+1)}} \delta^{p+1} y \right). \tag{3.12}$$

The β_is for this method are given implicitly by

$$\sum_{j=0}^{m} \beta_j y^{n+j} = y^{n+m} + \frac{\eta^{(p+1)}}{1 - \eta^{(p+1)} c_m^{(p+1)}} \delta^{p+1} y.$$

Since $\delta^p t = 0$ for $p \geq 2$, this is indeed an OLM of the form (3.3).

Proposition 3.3.1. *Algorithms 5 and 6 are equivalent.*

Proof. Subtract $\eta^{(p+1)}c_m^{p+1}y^{n+m}$ from both sides of (3.11) and use (3.4).

$$(1-\eta^{(p+1)}c_m^{(p+1)})y^{n+m} = (1-\eta^{(p+1)}c_m^{(p+1)})y_p^{n+m} - \eta^{(p+1)}\delta^{p+1}y.$$

Solving for y_p^{n+m} gives

$$y_p^{n+m} = y^{n+m} + \frac{\eta^{(p+1)}}{1-\eta^{(p+1)}c_m^{(p+1)}}\delta^{p+1}y. \tag{3.13}$$

Substituting this into the BDFp step (3.9), the right hand side of (3.9) becomes the right hand side of (3.12) as desired. The left hand side of (3.9) becomes

$$\sum_{j=1}^{p}\left[\prod_{i=1}^{j-1}(t_{n+m}-t_{n+m-i})\right]\delta^j y \tag{3.14}$$
$$+\left\{\frac{\eta^{(p+1)}}{1-c_m^{(p+1)}\eta^{(p+1)}}\sum_{j=1}^{p}\left[\prod_{i=1}^{j-1}(t_{n+m}-t_{n+m-i})\right]c_m^{(j)}\right\}\delta^{p+1}y.$$

We next simplify the scalar, shown in braces, multiplying $\delta^{p+1}y$ in (3.14). First, note that a simple calculation (not shown) gives

$$c_m^{(j)} = \left(\prod_{i=1}^{j}(t_{n+m}-t_{n+m-i})\right)^{-1}.$$

Splitting the term in braces apart, we see that

$$\frac{\eta^{(p+1)}}{1-c_m^{(p+1)}\eta^{(p+1)}} = \frac{\frac{\prod_{i=1}^{p}(t_{n+m}-t_{n+m-i})}{\sum_{j=1}^{p+1}(t_{n+m}-t_{n+m-j})^{-1}}}{1-\left(\prod_{i=1}^{p+1}(t_{n+m}-t_{n+m-i})\right)^{-1}\frac{\prod_{i=1}^{p}(t_{n+m}-t_{n+m-i})}{\sum_{j=1}^{p+1}(t_{n+m}-t_{n+m-j})^{-1}}} =$$
$$= \frac{\prod_{i=1}^{p}(t_{n+m}-t_{n+m-i})}{\sum_{j=1}^{p+1}(t_{n+m}-t_{n+m-j})^{-1}-(t_{n+m}-t_n)^{-1}} = \frac{\prod_{i=1}^{p}(t_{n+m}-t_{n+m-i})}{\sum_{j=1}^{p}(t_{n+m}-t_{n+m-j})^{-1}}$$

and

$$\sum_{j=1}^{p}\left[\prod_{i=1}^{j-1}(t_{n+m}-t_{n+m-i})\right]c_m^{(j)}$$
$$=\sum_{j=1}^{p}\left[\prod_{i=1}^{j-1}(t_{n+m}-t_{n+m-i})\right]\left(\prod_{i=1}^{j}(t_{n+m}-t_{n+m-i})\right)^{-1} = \sum_{j=1}^{p}(t_{n+m}-t_{n+m-j})^{-1}.$$

Thus, the term in braces simplifies to

$$\sum_{j=1}^{p}\left[\prod_{i=1}^{j-1}(t_{n+m}-t_{n+m-i})\right]c_m^{(j)}\frac{\eta^{(p+1)}}{1-c_m^{(p+1)}\eta^{(p+1)}} = \prod_{i=1}^{p}(t_{n+m}-t_{n+m-i}). \tag{3.15}$$

Absorbing this into the sum in (3.14) gives the desired left hand side of (3.12). □

3.3.1 Stability and error analysis of FBDFp+1

By construction, the left hand side of FBDFp+1 is that of BDFp+1, and the argument of the function evaluation in FBDFp+1 is a consistent approximation to $y(t_{n+m})$. It is immediate that FBDFp+1 is 0-stable whenever BDFp+1 is 0-stable since they have the same first characteristic polynomial. Therefore, FBDF6, which is BDF5 plus a time filter, is the highest order 0-stable method for constant stepsize. The 0-stability of variable step methods is highly nontrivial, and conditions on the stepsize ratios to guarantee 0-stability for BDF methods are given [13], and [44] which improved the upper bound on the stepsize ratios for BDF3. A recent technique for analyzing 0-stability for general methods is shown in [77].

The consistency error analysis, presented next, is an application of Definition 3.2.2.

Theorem 3.3.1. *FBDFp+1 is consistent of order* $p+1$.

Proof. The differentiation error L_d (see Definition 3.2.1) of FBDFp+1 is the same as the local truncation error of BDFp+1 which annihilates polynomials up to order $p+1$, so $p_d = p+1$. The interpolation error L_i is

$$\begin{aligned}(L_i\phi)\left(\sum_{i=0}^{m}\beta_i t_{n+i}\right) &= \sum_{i=0}^{m}\beta_i\phi(t_{n+i}) - \phi\left(\sum_{i=0}^{m}\beta_i t_{n+i}\right)\\ &= \phi(t_{n+m}) + \frac{\eta^{(p+1)}}{1-\eta^{(p+1)}c_m^{(p+1)}}\delta^{p+1}\phi - \phi(t_{n+m}) = \frac{\eta^{(p+1)}}{1-\eta^{(p+1)}c_m^{(p+1)}}\delta^{p+1}\phi. \quad (3.16)\end{aligned}$$

If ϕ is smooth, then for some $\xi \in (t_n, t_{n+m})$,

$$\delta^{p+1}\phi = \phi[t_{n+m}, t_{n+m-1}, \cdots, t_n] = \frac{\phi^{(p+1)}(\xi)}{(p+1)!},$$

see e.g. [52]. Hence, $(L_i\phi)\left(\sum_{i=0}^{m}\beta_i t_{n+i}\right)$ is zero on polynomials of degree less than or equal to p, so $p_i = p$. Thus, the order of the the method is $\min(p_d, p_i+1) = p+1$. □

3.3.2 Error estimation

Let y_{p+1}^{n+m} the FBDFp+1 approximation. As a consequence FBDFp+1 being $\mathcal{O}(\Delta t^{p+1})$, we have that

$$Est_p = y_{p+1}^{n+m} - y_p^{n+m}$$

(see Algorithm 5) is an estimate for the local error in BDFp. In order to estimate the local error for FBDFp+1, we need to estimate (3.8) which involves the exact solution. To approximate $L_i y$, note from (3.13) and (3.16)

$$L_i y(t) = \frac{\eta^{(p+1)}}{1 - \eta^{(p+1)} c_m^{(p+1)}} \delta^{p+1} y(t) \approx y_p^{n+m} - y_{p+1}^{n+m} = -Est_p.$$

To approximate the rest of (3.8), use $y(t_{n+i}) \approx y^{n+i}$ a possible error estimate of FBDFp+1 is

$$Est_{p+1} = \left(\sum_{j=1}^{p+1} \left[\prod_{i=1}^{j-1} (t_{n+m} - t_{n+m-i}) \right] \delta^j y^{p+1} - f(t_{n+m}, y_{p+1}^{n+m}) \right. \\ \left. + f_y(t_{n+m}, y_{p+1}^{n+m}) \cdot Est_p \right) \Big/ \bar{\alpha}_m^{(p+1)} \tag{3.17}$$

However, plugging the approximate solution into (3.17) underpredicts the error, which lead to overly large stepsizes. This is because (3.17) is the leading term of the residual of the method, so it should be approximately zero for the discrete solution. A more successful estimator in our tests was to only estimate L_d with

$$Est_{p+1} = \left(\sum_{j=1}^{p+1} \left[\prod_{i=1}^{j-1} (t_{n+m} - t_{n+m-i}) \right] \delta^j y^{p+1} - f(t_{n+m}, y_{p+1}^{n+m}) \right) \Big/ \bar{\alpha}_m^{(p+1)}. \tag{3.18}$$

Using (3.18) is pessimistic because it does not allow for cancellation with the neglected term. It's success may be due to the fact that enforcing small Est_{p+1} also forces the y_{p+1}^{n+m} to make the BDFp+1 residual small. Thus, we are forcing the solution to be within ε of satisfying a nearby method of order $p+1$. (3.18) also does not require evaluating the directional derivative of f. Since $C\Delta t^{p+1} y^{(p+1)}$ is the leading error term of (3.18), this could potentially be estimated instead using a $p+1$ order difference, which requires storing one more solution, but no function evaluation.

3.3.3 Order barrier

We prove in this section that it is impossible to increase the order of the simplest method, BDF1, by more than one by filtering y^{n+m} alone. This motivates the development of time filtered VSVO methods centered at higher order methods. We show in Section 3.3.4 that time filters can create more stable, lower order approximations.

Theorem 3.3.2 (An Order Barrier)**.** *BDF1 followed by a linear time filter of arbitrary finite length,*

$$\tilde{y}^{n+m} - y^{n+m-1} = \Delta t f(\tilde{y}^{n+m})$$

$$y^{n+m} = \tilde{y}^{n+m} - (c_m \tilde{y}^{n+m} + c_{m-1} y^{n+m-1} + \cdots + c_0 y^n) \qquad c_m \neq 0 \tag{3.19}$$

is of no higher than second order consistency.

Proof. For simplicity, we assume the ODE is autonomous, and that the step size is constant with $\Delta t_n = \Delta t$ for all n. Without loss of generality, we need only consider the application of one time filter; a sequence of time filters can be reduced to one of possibly greater length by eliminating the intermediate values.

Solving (3.19) for $\tilde{y}^{n+m}$,

$$\tilde{y}^{n+m} = \frac{1}{1-c_m}(y^{n+m} + c_{m-1} y^{n+m-1} + \cdots + c_0 y^n).$$

For convenience, we perform the change of variables $\hat{c}_m = (1-c_m)^{-1}$, $\hat{c}_{m-1} = (c_{m-1})(1-c_m)^{-1} - 1$, and $\hat{c}_i = c_i(1-c_m)^{-1}$ for $0 \leq i \leq m-2$.

$$\tilde{y}^{n+m} = y^{n+m-1} + \sum_{j=0}^{m} \hat{c}_j y^{n+j}$$

Substituting this into the backward Euler step yields the equivalent OLM

$$\sum_{j=0}^{m} \hat{c}_j y^{n+j} = \Delta t_{n+m-1} f(y^{n+m-1} + \sum_{j=0}^{m} \hat{c}_j y^{n+j}).$$

L_d must be zero on polynomials of order 3, and L_i must be zero on polynomials of order 2. We derive conditions on $\hat{c}_i$ from the differential error operator applied to $1, t$, and t^2. To simplify further, set $n = 0$.

$$(L_d 1)(t_m) = (\Delta t)^{-1} \sum_{i=0}^{m} \hat{c}_i = 0 \Longrightarrow \sum_{i=0}^{m} \hat{c}_i = 0$$

$$(L_d t)(t_m) = (\Delta t)^{-1} \sum_{i=0}^{m} \hat{c}_i i \Delta t - 1 = \sum_{i=0}^{m} \hat{c}_i i - 1 = 0 \Longrightarrow \sum_{i=0}^{m} \hat{c}_i i = 1$$

$$(L_d t^2)(t_m) = (\Delta t)^{-1} \sum_{i=0}^{m} \hat{c}_i (i\Delta t)^2 - 2(mk) = (\Delta t)^{-1} \sum_{i=0}^{m} \hat{c}_i i^2 \Delta t^2 - 2m\Delta t$$
$$= \Delta t \sum_{i=0}^{m} \hat{c}_i i^2 - 2m\Delta t = 0 \Rightarrow \sum_{i=0}^{m} \hat{c}_i i^2 = 2m$$

Next, apply the interpolation error operator to t^2.

$$(L_i t^2)(t_m) = \sum_{i=0}^{m} \hat{c}_i (i\Delta t)^2 + ((m-1)\Delta t)^2 - (\Delta t m)^2 = \sum_{i=0}^{m} \hat{c}_i i^2 \Delta t^2 + (m-1)^2 \Delta t^2 - \Delta t^2 m^2$$
$$= \Delta t^2 \sum_{i=0}^{m} \hat{c}_i i^2 + (m^2 - 2m + 1)\Delta t^2 - \Delta t^2 m^2$$
$$= 2m\Delta t^2 + m^2 \Delta t^2 - 2m\Delta t^2 + \Delta t^2 - \Delta t^2 m^2 = \Delta t^2.$$

The operator does not vanish on quadratics, so by Definition 3.2.2, the method is no higher than second order.

□

3.3.4 Stabilizing time filters

Adaptive codes using non-A-stable methods will, when necessary, decrease timesteps to enforce stability rather than accuracy. We thus need an A-stable member in the embedded family. This is achieved automatically for BDF1 plus a time filter (FBDF2), but we are limited to second order since filters can increase the order of accuracy by only one (see Theorem 3.3.2). Thus, we construct time filters that create lower order, but A-stable, embedded methods from BDF3 for constant stepsize. The result (BDF3-Stab below) is second order, G-stable and therefore A-stable by [23].

Recall that a method is absolutely stable for the product $\Delta t\lambda$ if the discrete solution to $y' = \lambda y$, $\lambda \in \mathbb{C}$, does not grow unbounded. A method is A-stable if its region of absolute stability contains the negative real half of the complex plane. A-stability can be analyzed by the root locus method by drawing the curve that separates points where the method is stable or unstable. We prove A-Stability using the equivalent G-Stability theory, since regions of absolute stability can be misleading if it is not obvious that the locus curve passes the imaginary axis. However, the locus plot for the method parameter choice $\mu = 9/125$ is shown in Figure 7a.

We then generalize the filter for variable stepsize. We begin with the ansatz that such a filter should be a third order perturbation of BDF3.

Algorithm 7 (Constant stepsize stabilized BDF3, BDF3-Stab)**.**

$$\begin{aligned} &\textit{BDF3 Step} && \frac{11y_3^{n+3} - 18y^{n+2} + 9y^{n+1} - 2y^n}{6\Delta t} = f(t_{n+3}, y_3^{n+3}) \\ &\textit{BDF3-Stab Step} && y_2^{n+3} = y_3^{n+3} + \mu(y_3^{n+3} - 3y^{n+2} + 3y^{n+1} - y^n) \end{aligned} \quad (3.20)$$

The induced OLM after eliminating the intermediate variable is

$$\begin{aligned} \frac{11y^{n+3} - 18y^{n+2} + 9y^{n+1} - 2y^n}{6\Delta t} - \frac{11}{6\Delta t}\frac{\mu}{1+\mu}(y^{n+3} - 3y^{n+2} + 3y^{n+1} - y^n) \\ = f\left(t^{n+3}, y^{n+3} - \frac{\mu}{1+\mu}(y^{n+3} - 3y^{n+2} + 3y^{n+1} - y^n)\right) \end{aligned} \quad (3.21)$$

From (3.21), a Taylor series calculation shows that the method is second order consistent with

$$\text{leading term of } \varepsilon^n = -\frac{11}{6}\frac{\mu}{1+\mu}y'''\Delta t^2.$$

What remains is to show that it may be G-Stable for a range of μ.

Theorem 3.3.3. *BDF3-Stab is G-Stable for $\mu \in$ [0.07143215,0.14285528]*

Proof. Multiplying (3.21) by $\frac{6\Delta t}{11}$ gives

$$\frac{y^{n+3}}{1+\mu} + \left(\frac{3\mu}{1+\mu} - \frac{18}{11}\right) y^{n+2} + \left(-\frac{3\mu}{1+\mu} + \frac{9}{11}\right) y^{n+1} + \left(\frac{\mu}{1+\mu} - \frac{2}{11}\right) y^{n} = \frac{6}{11}\Delta t f\left(t_{n+3}, \frac{y^{n+3}}{1+\mu} + \frac{3\mu}{1+\mu} y^{n+2} - \frac{3\mu}{1+\mu} y^{n+1} + \frac{\mu}{1+\mu} y^{n}\right).$$

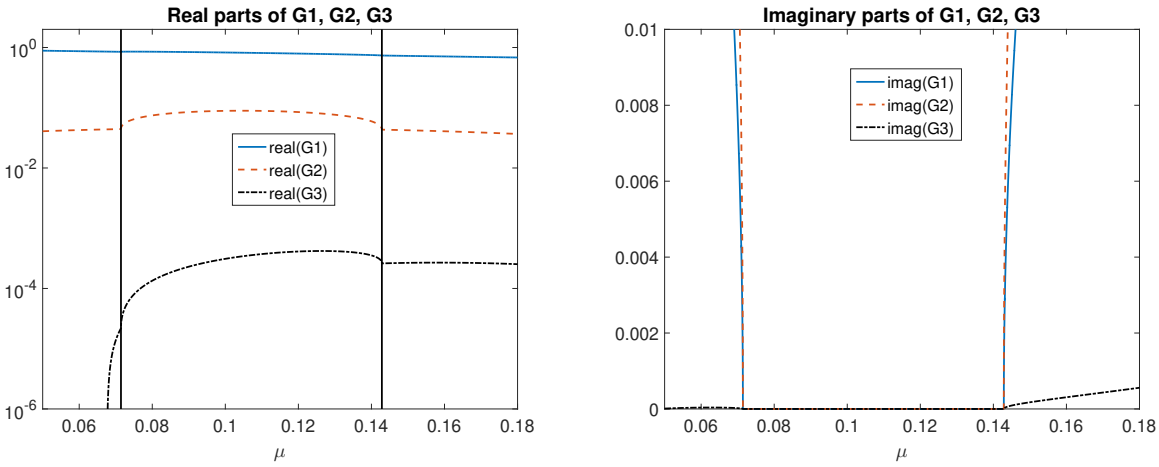
We show there exists a symmetric positive definite matrix $G = G_{3\times 3}$ and some constants a_3, a_2, a_1, a_0 such that

$$\begin{aligned}
&\left(\frac{y^{n+3}}{1+\mu} + \left(\frac{3\mu}{1+\mu} - \frac{18}{11}\right) y^{n+2} + \left(-\frac{3\mu}{1+\mu} + \frac{9}{11}\right) y^{n+1} + \left(\frac{\mu}{1+\mu} - \frac{2}{11}\right) y^{n}\right) \\
&\cdot \left(\frac{y^{n+3}}{1+\mu} + \frac{3\mu}{1+\mu} y^{n+2} - \frac{3\mu}{1+\mu} y^{n+1} + \frac{\mu}{1+\mu} y^{n}\right) \\
&= \left\| \begin{matrix} y^{n+3} \\ y^{n+2} \\ y^{n+1} \end{matrix} \right\|_G^2 - \left\| \begin{matrix} y^{n+2} \\ y^{n+1} \\ y^{n} \end{matrix} \right\|_G^2 + \|a_3 y^{n+3} + a_2 y^{n+2} + a_1 y^{n+1} + a_0 y^{n}\|^2.
\end{aligned}$$

We solved for g_{ij} and a_i in terms of μ symbolically with MATLAB, and write them explicitly in B.3.1. Using Sylvesters criterion, we seek an interval of μ for which the principal minors of G have positive determinant. From smallest to largest principal minors, we denote their determinants as G_1, G_2, and $G_3 = det(G)$. By plotting the real and imaginary parts of the determinants, we see that the real parts are positive, and the imaginary parts vanish within the interval $\mu \in$ [0.07143215,0.14285528], which implies that BDF3-Stab is G stable for μ in this interval.

□

Dahlquist's Second Barrier states that the leading error constant C of all A-Stable LMMs is, in magnitude, greater than or equal to $C_{TR} = 1/12$, which is attained by the trapezoid rule [22]. For the left endpoint of the interval given in Theorem 3.3.3, BDF3-Stab has a leading error constant of $\frac{1}{12} < C_{\text{BDF3-Stab}} \approx 0.1222 < \frac{2}{12}$. This is about 2.73 times smaller than $C_{BDF2} = \frac{4}{12}$ (compare with the optimized blended BDF2/BDF3 and BDF2/BDF3/BDF4



(a) (b)

Figure 8: The real parts of $G1, G2$ and $G3$ are positive and the imaginary parts vanish in the region of G stability, which is bounded by the vertical bars in Fig. 8a.

schemes in [79] which have respectively 2 and 2.64 times smaller leading constants than C_{BDF2}).

The filter is extended (nonuniquely) to variable stepsize by replacing $y_3^{n+3} - 3y^{n+2} + 3y^{n+1} - y^n$ with the third order divided difference $\delta^3 y$, and rescaling by the leading coefficient $c_3^{(3)}$.

$$y_2^{n+3} = y_3^{n+3} + \frac{\mu}{c_3^{(3)}} \delta^3 y_3, \tag{3.22}$$

This gives a variable stepsize BDF3 method a stable method to switch to rather than cutting the timestep.

3.4 The VSVO Algorithm

In Section 3.3, we derived a general embedded method FBDFp+1 which increases the order of any BDFp method by one, and an embedded stabilized BDF3 method (BDF3-Stab) that is second order and G-Stable. We combine FBDF4 and BDF3-Stab to create

an embedded implicit method that is second, third, and fourth order called Multiple Order One Solve Embedded 234 (MOOSE234). After a filter is applied, the new solution is used to estimate the error in the pre-filtered solution. We then pick the solution which allows for the largest stepsize to be taken. The variable stepsize algorithm is as follows.

Algorithm 8 (MOOSE234). *Let $m = 4$. Given ε, $\tilde{\gamma}$, γ, $\{y^n, \cdots, y^{n+3}\}$, $\mu \in$ [0.07143215,0.14285528], compute $y_2^{n+4}, y_3^{n+4}, y_4^{n+4}$ by solving*

BDF3
$$\sum_{j=1}^{3}\left[\prod_{i=1}^{j-1}(t_{n+4}-t_{n+4-i})\right]\delta^j y_3 = f(t_{n+4}, y_3^{n+4})$$

BDF3-Stab
$$y_2^{n+4} = y_3^{n+4} + \frac{\mu}{c_4^{(3)}}\delta^3 y_3$$

FBDF4
$$y_4^{n+4} = y_3^{n+4} - \eta^{(4)}\delta^4 y_3.$$

Put $Est_2 = y_3^{n+4} - y_2^{n+4}$, $Est_3 = y_4^{n+4} - y_3^{n+4}$, and

$$Est_4 = \left(\sum_{j=1}^{4}\left[\prod_{i=1}^{j-1}(t_{n+4}-t_{n+4-i})\right]\delta^j y_4 - f(t_{n+4}, y_4^{n+4})\right)\Big/\bar{\alpha}_4^{(4)} \tag{3.23}$$

Of the solutions that satisfy $|Est_i| < \varepsilon$, find j that would allow a maximum step size to be taken.

$$j = \underset{i\in\{2,3,4\}}{\arg\max}\left(\frac{\varepsilon}{|Est_i|}\right)^{\frac{1}{i+1}}, \qquad \Delta t_{n+4} = \gamma\Delta t_{n+3}\left(\frac{\varepsilon}{|Est_j|}\right)^{\frac{1}{j+1}} \tag{3.24}$$

Then set $y^{n+4} = y_j^{n+4}$. If none satisfy the tolerance, set

$$\Delta t_{n+3} := \max_j \tilde{\gamma}\Delta t_{n+3}\left(\frac{\varepsilon}{|Est_j|}\right)^{\frac{1}{j+1}},$$

and recompute the above steps.

The second equation in (3.24) is a standard formula for stepsize selection given an estimate for the local truncation error [76],[41],[1]. $\gamma \leq 1$ is a safety factor used to keep the next step size from growing too fast, and to increase the chance that the next solution will be accepted. If a solution is rejected, $\tilde{\gamma} \leq \gamma$ is a second factor used to nudge the stepsize in a direction that will make the recomputed solution more likely to be accepted. We took $\gamma = 0.9$, and $\tilde{\gamma} = 0.7$. Est_4 is an estimation of the local truncation error (3.18) of FBDF4 (justification in Section 3.3.2).

For step size control, we use the most popular criteria [76], [41] which is to require the local truncation error be less than a user supplied tolerance ε. Other criterion, not tested herein, include requiring the error per unit time interval be less than a tolerance, LTE $< \Delta t_n \varepsilon$ [19],[76], or adapting to control the global error [18]. In our implementation, we add a common heuristic that the step size can change by no more than a factor of two at a time [1], although this may be overly cautious since factors as large as five have been used for adaptive BDF methods [47]. Many other considerations for implementation and improvement of adaptive methods are discussed in the PhD thesis of Ahmad [1].

The algorithm above is of variable order two through four, but different methods can be obtained by taking a max in (3.24) over a subset of $\{2, 3, 4\}$, which is tested numerically in Section 3.5.

3.5 Applications to Nonlinear Evolution Equations

MOOSE234 is easily implemented for nonlinear evolution equations with decreased cost, increased assured accuracy and thereby increased predictive power. We give one test on a highly stiff and fluctuating ODE (the Van der Pol oscillator), and two tests for the Navier-Stokes equations, which describe a phenomena for which predictive accuracy is important and where memory limitations, accuracy requirements, cognitive and computational complexity are often in competition.

The Van der Pol test is given in Section 3.5.1. The NSE and the spatial discretization used herein are defined in Section 3.5.2. In Section 3.5.3 , the constant stepsize, constant

order methods are applied to a Taylor-Green vortex array. This is a common benchmark problem in CFD such as [16], and others. The VSVO method is tested for the same problem in Section 3.5.4.

We measure error versus work by the number of time steps taken plus the number of rejected solution in Section 3.5.1, and by compute time in Section 3.5.4. We test methods of different orders (2,23,234,3,34,4) by restricting the approximations that MOOSE234 is allowed to select. The method of order three is simply adaptive BDF3, 23 is adaptive BDF3 and BDF3-Stab, etc. If the method does not include 4, we do not evaluate the error estimator for FBDF4 so that it does not artificially inflate the runtime of the other methods. We chose the method parameter $\mu = \frac{9}{125} = 0.072$ somewhat arbitrarily. It comes close to minimizing the leading term of the local truncation error within the interval of μ for which it is G-Stable. A larger μ may be useful if more dissipation is needed.

3.5.1 Van der Pol oscillator

In this section, we test the methods on the Van der Pol oscillator, a common test problem for stiff ODE integrators. We do not compare runtime performance with well calibrated, existing ODE codes. We just demonstrate the adaptivity of the methods in a simpler setting before the PDE tests. The Van der Pol oscillator is

$$y_1' = y_2, \qquad y_2' = \bar{\mu}(1 - y_1^2)y_2 - y_1$$

with $\bar{\mu} = 1000$. We compute relative errors at $t = 3000$ by comparing with a reference solution from MATLAB's ode15s with an absolute tolerance of 1e-16, and a relative tolerance of 3e-14.

The error vs total work (number of steps taken plus rejected steps) is shown in Figure 9, and clearly shows that the higher order methods are most efficient for this problem. We tested many combinations of orders (2,23,234,3,34,4) to verify that the higher order methods reduced the total amount of work, although we do not plot the results of all these combinations. Specific to this test, we note that 23 performed essentially the same as adaptive BDF3, and 34 was essentially the same as MOOSE234. Adaptive FBDF4 appears to perform the

best, although it does not have an obvious trend. Other tests such as the one performed in Section 3.5.4 do show a notable increase in efficiency using the full MOOSE234 versus using a subset of the available orders.

The stepsize and order evolution of MOOSE234 is shown in Figure 10. While MOOSE234 chooses the BDF3 approximation for most of the simulation, the narrow sections where the fourth order method is used is enough to make it globally fourth order, shown in Figure 9.

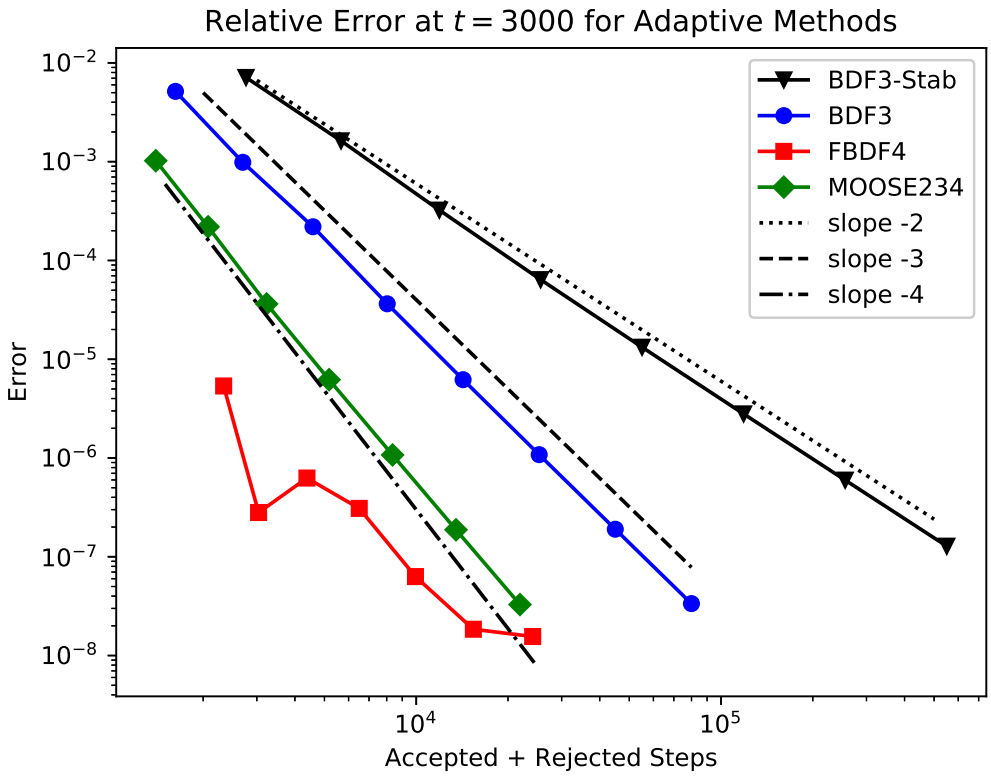


Figure 9: The fourth order methods produce the smallest errors in the least amount of steps.

3.5.2 Finite element formulation

Given the domain $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$, consider the problem

$$u_t + u \cdot \nabla u - \nu \Delta u + \nabla p = f \text{ and } \nabla \cdot u = 0 \text{ in } \Omega \times (0, T]$$
$$u = 0 \text{ on } \partial\Omega \text{ and } u(x, 0) = u_0(x).$$

To discretize in space, let (X^h, Q^h) be a finite element pair satisfying the inf-sup, or LBB_h condition. We suppress the spatial discretization on velocity and pressure to avoid excessive super and subscripts. Define the explicitly skew-symmeterized trilinear form $b^*(u, v, w) = (u \cdot \nabla v, w) + \frac{1}{2}((\nabla \cdot u)v, w)$.

The fully discrete BDF3 problem is as follows. Find $(u_3^{n+4}, p^{n+4}) \in (X^h, Q^h)$ such that for all $(v_h, q_h) \in (X^h, Q^h)$,

$$\left(\bar{\alpha}_4^{(3)} u_3^{n+4} + \sum_{j=1}^{3} \bar{\alpha}_j^{(3)} u^{n+j}, v_h \right) + b^*(u_3^{n+4}, u_3^{n+4}, v_h) + \nu(\nabla u_3^{n+4}, \nabla v_h)$$
$$-(p^{n+4}, \nabla \cdot v_h) = (f(t_{n+4}), v_h), \qquad \text{and} \qquad (\nabla \cdot u_3^{n+4}, q_h) = 0.$$

This method for constant stepsize was analyzed in [10] under a small data condition with the nonlinearity treated explicitly. We linearize the problem as follows. Let

$$\hat{u}_3^{n+4} = -\frac{1}{c_4^{(4)}} \sum_{i=0}^{3} c_i^{(4)} u^{n+i}$$

which is a fourth order extrapolation of u_3^{n+4} to preserve the order of consistency of FBDF4. Then the linearly implicit (sometimes called semi-implicit) method is obtained by replacing $b^*(u_3^{n+4}, u_3^{n+4}, v_h)$ with $b^*(\hat{u}_3^{n+4}, u_3^{n+4}, v_h)$.

Implicit-explicit [17][81] and linearly implicit methods [29][3] are a common way to reduce the computational complexity of time stepping nonlinear problems . The idea of Baker [9] to treat the convective term in Galerkin approximations of Navier-Stokes this way while preserving skew-symmetry has a long history of use and expansions [38, p. 185] [51] [72] [54][2].

Pressure is not a dynamic variable; only the pressure at the current time level is required, so applying the time filters to pressure will not affect the computed velocity solution. Thus,

we choose not to filter it for these tests. Therefore, Est_2, Est_3, and Est_4 are only estimates of the temporal velocity error. $|Est_i| = ||Est_i||$ is the $L^2(\Omega)$ norm.

The error for FBDF4 is estimated by Est_4, where Est_4 is the finite element discretization of (3.23), and is the solution of

$$\bar{\alpha}_4^{(4)}(Est_4, v_h) = (\bar{\alpha}_4^{(4)} u_4^{n+4} + \sum_{j=0}^{3} \bar{\alpha}_j^{(4)} u^{n+j}, v_h) + b^*(u_4^{n+4}, u_4^{n+4}, v_h)$$
$$-(p^{n+4}, \nabla \cdot v_h) - (f(t_{n+4}), v_h) \qquad \text{for all } v_h \in X^h.$$

This is the only non-embedded error estimator, and since it is a mass matrix with order one condition number and narrow band width, does not add significantly to the computational complexity. In our tests in Section 3.5.4 with 495,000 degrees of freedom, this system can be solved with about 10 iterations of the conjugate gradient method within a relative tolerance of 1e-6; this takes about 0.1 seconds on a desktop with a four core Intel i7 7700k cpu. The time taken to solve this system is included in the timing tests in Figure 12. All tests were performed with FEniCS [4].

In the adaptive tests, the stepsize ratios were limited to a maximum of two and a minimum of one half, which is a common heuristic [1]. All tests were performed on a square periodic domain using P5/P4 Lagrange elements with 32 elements per edge of the square resulting in 67,584 degrees of freedom.

3.5.3 Constant stepsize test

We test the case of constant step size on a Taylor-Green vortex array, a common benchmark problem in CFD.

We took Ω to be the periodic box with sides of length 2π, and $\nu = 1$. Define $F(t) = e^{-2\nu t}$. An exact solution is given by

$$u(x, y; t) = F(t)(\cos x \sin y, -\cos y \sin x), \qquad p(x, y; t) = -\frac{1}{4} F(t)^2 (\cos 2x + \cos 2y).$$

The final time was taken to be $T = 1$. Figure 11 shows the relative l2L2 velocity and pressure errors for different stepsize Δt. We achieve convergence rates in time for the velocity and pressure predicted by the ODE theory.

3.5.4 Variable stepsize variable order test

In this test, we allow the method to adapt, and run the above problem to a final time of $T = 1$. The same mesh from the constant stepsize test was used. All tests were initialized with exact solutions that were Δt =1E-3 apart. The safety factors were $\gamma = 0.9$, and $\tilde{\gamma} = 0.7$.

We tested many combinations of orders (2,23,234,3,34,4) to verify that variable order is necessary for an improvement in execution time. We do not show the results of all these combinations in the plots for clarity, but we note that the method of order 23 is slightly more efficient than adaptive BDF3. The method of order 34 performed better than FBDF4, but slightly worse than MOOSE234 for larger tolerances. Each test was timed starting at the outer time stepping loop of the program, and ending after the final time step. Various ε were tested from 1e-1 to 1e-8. Figure 12 shows the amount of time in seconds each method required to run to completion for different tolerances versus the relative l2L2 errors, with the tolerances decreasing from left to right.

For the smallest tolerances, we clearly see that the higher order methods are the most efficient with the full MOOSE234 method performing the best. MOOSE234 is about two times faster than adaptive BDF3 for the final tolerance.

3.6 Conclusion

We present MOOSE234, a new stiff VSVO solver of orders two, three, and four. The computational complexity is comparable to BDF3. In our tests on the Van der Pol oscillator and a standard spatial discretization of the Navier-Stokes equations, the VSVO methods of higher order give the most accurate approximations at least cost. We also developed FBDFp+1 of orders two through six, which uses computationally inexpensive time filters to increase the order of all variable stepsize BDFp methods with $p \leq 5$ by one.

Many open problems remain. Linearly implicit (tested herein for Navier-Stokes) and Implicit-Explicit versions need systematic development. Error analysis of the fully discrete method for NSE, and a deeper understanding of the pressure error are needed. There may

exist more optimal G-Stabilizing filters for BDF3. The idea of constructing second order G-stabilizing time filters can be applied to higher order BDF methods, and other multistep methods. MOOSE234 can be applied to other complex nonlinear applications.

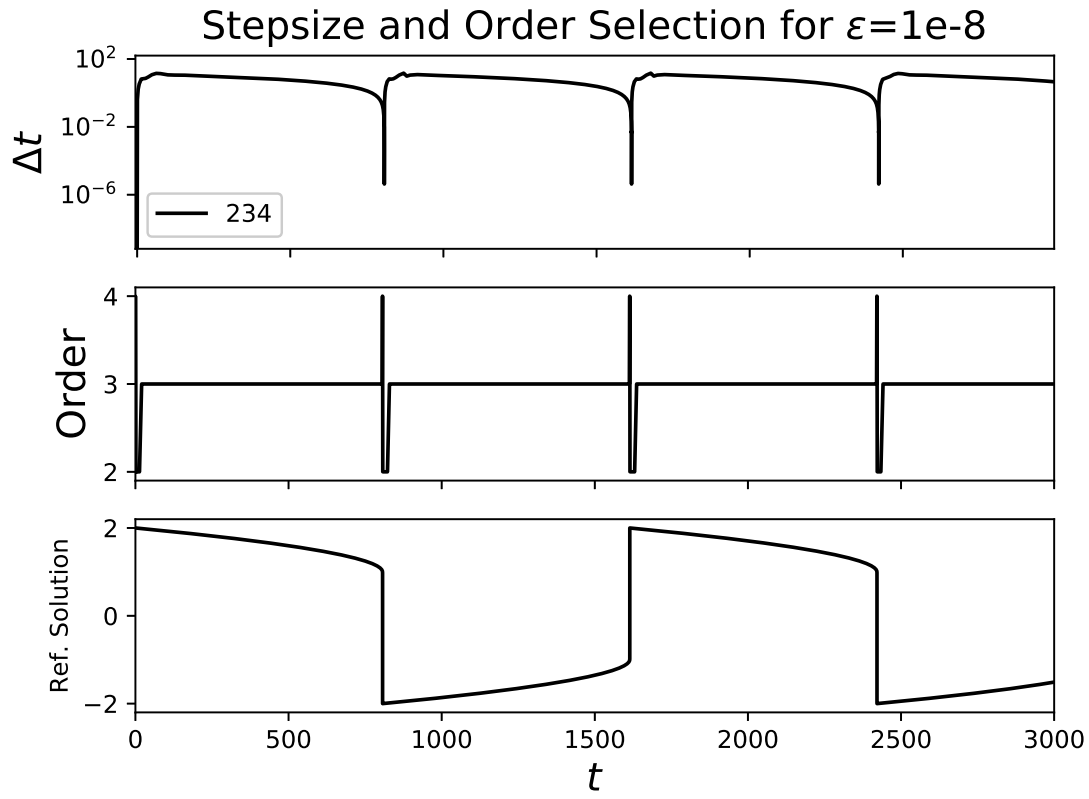


Figure 10: Stepsize and order adaptation for the Van der Pol oscillator.

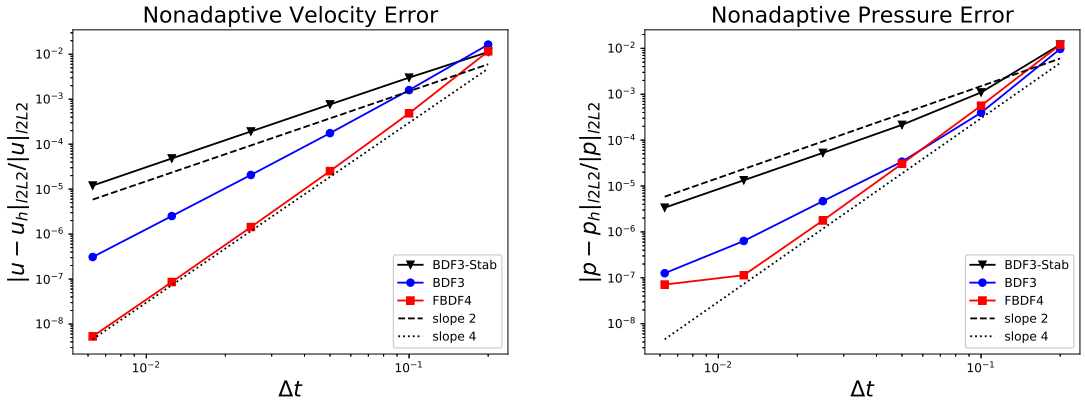


Figure 11: Velocity and pressure converge at the predicted rates.

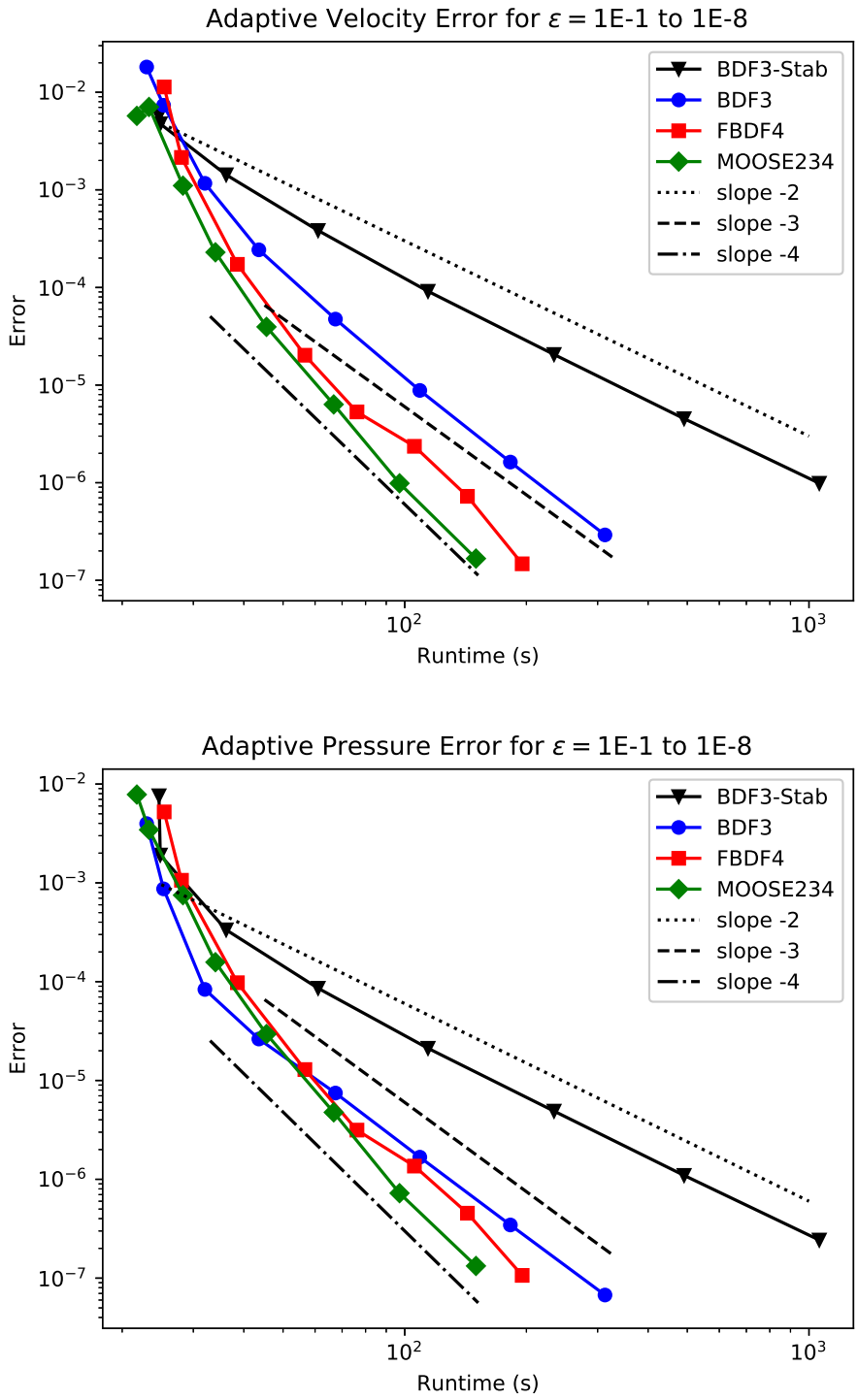


Figure 12: The greatest accuracy for the least compute time is from the higher order methods.

4.0 An Implicit-Explicit VSVO Method for Navier-Stokes

4.1 Introduction

In this chapter, we revisit the method developed in Chapter 2. We present with supporting analysis an adaptive time discretization of the Navier-Stokes equations (NSE) requiring, at each timestep the solution of a shifted stokes problem. Methods of this type are known to be inexpensive per step, but often have a severe timestep restriction due to the explicit treatment of the nonlinear term. As solvers have matured and memory increased, methods of this type have seen decreased development. However with the recent explosion of interest in uncertainty quantification and machine learning, along with newly emerging computational architectures methods requiring less memory, and lower computational complexity have become interesting tools again. We stress that our analysis is for a variable timestep. This is a unique feature that reduces the gap between the needs of practical CFD and what analysis can contribute.

This chapter is organized as follows. In Section 4.2 we present preliminary analysis which will be needed in the ensuing sections. The fully discrete algorithm we will analyze is outlined in Section 4.3. In Section 4.4 we present an energy stability analysis of the algorithm by analyzing stability of each order independently. The variable stepsize, first order algorithm is analyzed in Section 4.4.1. The constant stepsize version of the second order method is analyzed in Section 4.4.2.

4.1.1 Previous works

Variable timestep schemes have been studied extensively for linear multistep methods for ODES see [20, 27] and the references therein. Linear stability analysis for constant timestep BDF2-AB2 and CNLF applied to systems of linear evolution equations with skew symmetric couplings was conducted in [62]. It was shown under a timestep condition that both methods were long time energy stable. The CNLF scheme was proven to be unstable for

variable timestep in [65]. Recently, for the NSE adaptive time stepping schemes have been studied for a variety of second order implicit and linearly implicit methods [57, 59, 12]. It was demonstrated that time adaptivity increased the accuracy and efficiency of the schemes. A stability analysis of these methods for increasing and decreasing timestep ratio is still an open problem. Constant timestep IMEX schemes for the NSE have been studied for CN-AB2 [68, 58], a three-step backward extrapolating scheme in [10] and BEFE in [48].

4.2 Notation and Preliminaries

Let $\Omega \subset \mathbb{R}^d, d = 2, 3$, denote an open regular domain with boundary $\partial\Omega$ and let $[0, T]$ denote a time interval. We consider the incompressible Navier-Stokes equations (NSE)

$$\begin{cases} u_t + u \cdot \nabla u - \nu \Delta u + \nabla p = f(x,t) & \forall x \in \Omega \times (0,T] \\ \nabla \cdot u = 0 & \forall x \in \Omega \times (0,T] \\ u = 0 & \forall x \in \partial\Omega \times (0,T] \\ u(x,0) = u^0(x) & \forall x \in \Omega. \end{cases} \tag{4.1}$$

We denote by $\|\cdot\|$ and $(\cdot,\cdot)$ the $L^2(\Omega)$ norm and inner product, respectively, and by $\|\cdot\|_{L^p}$ and $\|\cdot\|_{W_p^k}$ the $L^p(\Omega)$ and Sobolev $W_p^k(\Omega)$ norms, respectively. $H^k(\Omega) = W_2^k(\Omega)$ with norm $\|\cdot\|_k$. The space $H^{-1}(\Omega)$ denotes the dual space of bounded linear functionals defined on $H_0^1(\Omega) = \{v \in H^1(\Omega) \,:\, v = 0 \text{ on } \partial\Omega\}$; this space is equipped with the norm

$$\|f\|_{-1} = \sup_{0 \neq v \in X} \frac{(f,v)}{\|\nabla v\|} \quad \forall f \in H^{-1}(\Omega).$$

The solutions spaces X for the velocity and Q for the pressure are respectively defined as

$$\begin{aligned} X :=& [H_0^1(\Omega)]^d = \{v \in [L^2(\Omega)]^d \,:\, \nabla v \in [L^2(\Omega)]^{d\times d} \text{ and } v = 0 \text{ on } \partial\Omega\} \\ Q :=& L_0^2(\Omega) = \Big\{q \in L^2(\Omega) \,:\, \int_\Omega q dx = 0\Big\}. \end{aligned}$$

A weak formulation of (4.1) is given as follows: find $u : (0,T] \to X$ and $p : (0,T] \to Q$ such that, for almost all $t \in (0,T]$, satisfy

$$\begin{cases} (u_t, v) + (u \cdot \nabla u, v) + \nu(\nabla u, \nabla v) - (p, \nabla \cdot v) = (f, v) & \forall v \in X \\ (\nabla \cdot u, q) = 0 & \forall q \in Q \\ u(x,0) = u^0(x). \end{cases} \tag{4.2}$$

The subspace of X consisting of weakly divergence-free functions is defined as

$$V := \{v \in X \,:\, (\nabla \cdot v, q) = 0 \ \forall q \in Q\} \subset X.$$

We denote conforming velocity and pressure finite element spaces based on a regular triangulation of Ω having maximum triangle diameter h by $X_h \subset X$ and $Q_h \subset Q$. We assume that the pair of spaces (X_h, Q_h) satisfy the discrete inf-sup (or LBB_h) condition required for stability of finite element approximations; we also assume that the finite element spaces satisfy the approximation properties

$$\begin{aligned} \inf_{v_h \in X_h} \|v - v_h\| &\le Ch^{s+1} \quad \forall v \in [H^{s+1}(\Omega)]^d \\ \inf_{v_h \in X_h} \|\nabla(v - v_h)\| &\le Ch^{s} \quad \forall v \in [H^{s+1}(\Omega)]^d \\ \inf_{q_h \in Q_h} \|q - q_h\| &\le Ch^{s} \quad \forall q \in H^{s}(\Omega), \end{aligned}$$

where C is a positive constant that is independent of h. The Taylor-Hood element pairs (P^s-P^{s-1}), $s \ge 2$, are one common choice for which the LBB_h stability condition and the approximation estimates hold [39, 45].

We will also assume that the mesh satisfies the following standard inverse inequalities

$$\|v_h\| \le Ch^{-1}\|\nabla v_h\| \quad \forall v_h \in X_h \tag{4.3}$$

$$\|v_h\|_\infty \le C|\ln h|^{1/2}\|\nabla v_h\| \quad \forall v_h \in X_h, \text{ and for } d = 2 \tag{4.4}$$

We define the trilinear form

$$b(u, v, w) = (u \cdot \nabla v, w) \qquad \forall u, v, w \in [H^1(\Omega)]^d$$

and the explicitly skew-symmetric trilinear form given by

$$b^*(u,v,w) := \frac{1}{2}(u\cdot\nabla v, w) - \frac{1}{2}(u\cdot\nabla w, v) \qquad \forall u,v,w \in [H^1(\Omega)]^d,$$

or equivalently,

$$b^*(u,v,w) := (u\cdot\nabla v, w) + \frac{1}{2}(\nabla\cdot u, v\cdot w) \qquad \forall u,v,w \in [H^1(\Omega)]^d.$$

This satisfies the bound [63]

$$b^*(u,v,w) \le C_{b^*}\|\nabla u\|\|\nabla v\|\|\nabla w\| \qquad \forall u,v,w \in X. \tag{4.5}$$

$$b^*(u,v,w) \le C_{b^*}(\|u\|\|\nabla u\|)^{1/2}\|\nabla v\|\|\nabla w\| \qquad \forall u,v,w \in X. \tag{4.6}$$

$$b^*(u,v,w) \le C_{b^*}\|\nabla u\|(\|v\|\|\nabla v\|)^{1/2}\|\nabla w\| \qquad \forall u,v,w \in X. \tag{4.7}$$

Additionally, we have the following bound.

Lemma 4.2.1.

$$b^*(u,v,w) \le C_{b^*}\|\nabla u\|\|\nabla v\|(\|w\|\|\nabla w\|)^{1/2} \qquad \forall u,v,w \in X \tag{4.8}$$

Proof. We have by repeated Hölders inequality that

$$\begin{aligned}(\nabla\cdot u, v\cdot w) &= \sum_{i=1}^d \int_\Omega (\nabla\cdot u) v_i w_i dx \le \sum_{i=1}^d \|\nabla\cdot u\|\|v_i\|_{L^6}\|w_i\|_{L^3} \\ &\le \sqrt{d}\|\nabla\cdot u\|\|v\|_{L^6}\|w\|_{L^3} \le C(d)\|\nabla u\|\|v\|_{L^6}\|w\|_{L^3}.\end{aligned}$$

Similarly, we have

$$\int_\Omega (u\cdot\nabla v)\cdot w dx \le C(d)\|u\|_{L^6}\|\nabla v\|\|w\|_{L^3}.$$

By Sobolev embedding theorems, $H^1 \hookrightarrow L^6$ and $H^{\frac{1}{2}} \hookrightarrow L^3$ for $d = 2, 3$. The result then follows from the interpolation inequality $\|w\|_{\frac{1}{2}} \le C\|w\|^{\frac{1}{2}}\|\nabla w\|^{\frac{1}{2}}$. □

We also define the discretely divergence-free space V_h as

$$V_h := \{v_h \in X_h \,:\, (\nabla\cdot v_h, q_h) = 0 \ \forall q_h \in Q_h\} \subset X.$$

4.3 Algorithm

The fully discrete constant stepsize scheme we consider is a Backward Euler in time with a second order Adams Bashforth treatment of the advection term (BE-AB2)

$$\begin{aligned}\left(\frac{u_h^{n+1}-u_h^n}{\Delta t}, v_h\right) + \nu(\nabla u_h^{n+1}, \nabla v_h) + b^*(2u_h^n - u_h^{n-1}, 2u_h^n - u_h^{n-1}, v_h) \\ -(p^{n+1}, \nabla \cdot v_h) = (f^{n+1}, v_h) \qquad \forall v_h \in X_h \\ (\nabla \cdot u_h^{n+1}, q_h) = 0 \qquad \forall q_h \in Q_h.\end{aligned} \tag{4.9}$$

For the case of nonuniform timesteps, let $\Delta t_n = t^{n+1} - t^n$. The stepsize ratios are $\omega_n = \frac{\Delta t_n}{\Delta t_{n-1}}$. The second order extrapolation of u_h^{n+1} becomes $E^{n+1}(u_h) := u_h^n + \omega_n(u_h^n - u_h^{n-1})$. We then have the variable stepsize BE-AB2 (VSS BE-AB2) method.

Algorithm 9 (Variable stepsize BE-AB2). *Given ε, Δt_n, (u_h^n, p_h^n),$(u^{n-1}{}_h \quad , p_h^{n-1})$, and a norm $\|\cdot\|_\#$, find (u_h^{n+1}, p_h^{n+1}) satisfying*

$$\begin{aligned}\left(\frac{u_h^{n+1}-u_h^n}{\Delta t_n}, v_h\right) + \nu(\nabla u_h^{n+1}, \nabla v_h) + b^*(E^{n+1}(u_h), E^{n+1}(u_h), v_h) \\ -(p^{n+1}, \nabla \cdot v_h) = (f^{n+1}, v_h) \qquad \forall v_h \in X_h \\ (\nabla \cdot u_h^{n+1}, q_h) = 0 \qquad \forall q_h \in Q_h.\end{aligned} \tag{4.10}$$

Let

$$Est = \frac{\omega_n}{2\omega_n + 1}\|u_h^{n+1} - E^{n+1}(u_h)\|_\#.$$

If *$Est \le \varepsilon$, accept (u_h^{n+1}, p_h^{n+1}) and let*

$$\Delta t_{n+2} = 0.9\Delta t_{n+1}\left(\frac{\varepsilon}{Est}\right)^{1/2}.$$

Else *Reject (u_h^{n+1}, p_h^{n+1}) and let*

$$\Delta t_{n+1} := 0.7\Delta t_{n+1}\left(\frac{\varepsilon}{Est}\right)^{1/2}.$$

Recalculate (u_h^{n+1}, p_h^{n+1}).

The numbers 0.9 and 0.7 are safety factors that make it more likely the next stepsize will give an accepted solution, which is common practice. They are chosen arbitrarily here. The error estimation is derived from the variable stepsize backward Euler time filter that results in a second order method. That is, if

$$u_h^{n+1,2} = u_h^{n+1} - \frac{\omega_n}{2\omega_n + 1}(u_h^{n+1} - (1+\omega_n)u_h^n + \omega_n u_h^{n-1}),$$

then $u_h^{n+1,2}$ is a second order approximation. Thus, $u_h^{n+1} - u_h^{n+1,2}$ provides an estimation for the error in the first order approximation. Rearranging this shows

$$\begin{aligned} u_h^{n+1} - u_h^{n+1,2} &= \frac{\omega_n}{2\omega_n + 1}(u_h^{n+1} - (1+\omega_n)u_h^n + \omega_n u_h^{n-1}) \\ &= \frac{\omega_n}{2\omega_n + 1}(u_h^{n+1} - E^{n+1}(u_h)). \end{aligned}$$

4.4 Stability

4.4.1 Energy stability for VSS BE-AB2

In this section we prove nonlinear conditional stability of (4.10). We begin with a general stability result.

Theorem 4.4.1. *[General Stability of VSS BE-AB2] Consider the method* (4.10) *and let* $\Omega \subset \mathbb{R}^d, d = 2, 3$. *Suppose that*

$$1 - \frac{C_{stab}\Delta t_n(1+\omega_n^2)}{\nu h}\|\nabla E^{n+1}(u_h)\|^2 \geq 0. \tag{4.11}$$

Then for any $N > 1$

$$\begin{aligned} &\frac{1}{2}\|u_h^N\|^2 + \frac{1}{4}\|u_h^N - u_h^{N-1}\|^2 + \frac{\nu}{4}\sum_{n=1}^{N-1}\Delta t_n\|\nabla u_h^{n+1}\|^2 \\ &+ \sum_{n=1}^{N-1}\frac{1}{8(1+\omega_n^2)}\|u_h^{n+1} - u_h^n + \omega_n(u_h^n - u_h^{n-1})\|^2 \leq \sum_{n=1}^{N-1}\frac{\Delta t_n}{\nu}\|f^{n+1}\|_{-1}^2 \\ &+ \frac{1}{2}\|u_h^1\|^2 + \frac{1}{4}\|u_h^1 - u_h^0\|^2. \end{aligned} \tag{4.12}$$

Proof. Setting $v_h = u_h^{n+1}$ and multiplying by Δt_n we have

$$\frac{1}{2}\|u_h^{n+1}\|^2 - \frac{1}{2}\|u_h^n\|^2 + \frac{1}{2}\|u_h^{n+1} - u_h^n\|^2 + \Delta t_n \nu \|\nabla u_h^{n+1}\|^2$$
$$+\Delta t_n b^*(E^{n+1}(u_h), E^{n+1}(u_h), u_h^{n+1}) = \Delta t_n (f^{n+1}, u_h^{n+1})$$

Applying Young's inequality to the right hand side then gives

$$\frac{1}{2}\|u_h^{n+1}\|^2 - \frac{1}{2}\|u_h^n\|^2 + \frac{1}{2}\|u_h^{n+1} - u_h^n\|^2 + \Delta t_n \nu \|\nabla u_h^{n+1}\|^2$$
$$\Delta t_n b^*(u_h^n + \omega_n(u_h^n - u_h^{n-1}), u_h^n + \omega_n(u_h^n - u_h^{n-1}), u_h^{n+1})$$
$$\leq \frac{\nu \Delta t_n}{4}\|\nabla u_h^{n+1}\|^2 + \frac{\Delta t_n}{\nu}\|f^{n+1}\|_{-1}^2.$$

Next we deal with the nonlinearity. Applying (4.8), using the skew symmetry of the nonlinearity, applying the Cauchy-Schwarz-Young, Poincaré-Friedrichs and inverse inequalities we have

$$\begin{aligned}
&\Delta t_n b^*(E^{n+1}(u_h), u_h^n + \omega_n(u_h^n - u_h^{n-1}), u_h^{n+1}) \\
&= \Delta t_n b^*(E^{n+1}(u_h), u_h^{n+1}, u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})) \\
&\leq C\Delta t_n h^{-\frac{1}{2}} \|\nabla E^{n+1}(u_h)\| \|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\| \|\nabla u_h^{n+1}\| \\
&\leq C\frac{\Delta t_n^2(1+\omega_n^2)}{h}\|\nabla E^{n+1}(u_h)\|^2 \|\nabla u_h^{n+1}\|^2 \\
&+ \frac{1}{8(1+\omega_n^2)}\|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\|^2.
\end{aligned}$$

For the last term we have by the parallelogram law

$$\begin{aligned}
&\frac{1}{8(1+\omega_n^2)}\|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\|^2 \\
&= \frac{1}{4(1+\omega_n^2)}\|u_h^{n+1} - u_h^n\|^2 + \frac{\omega_n^2}{4(1+\omega_n^2)}\|u_h^n - u_h^{n-1}\|^2 \\
&\quad - \frac{1}{8(1+\omega_n^2)}\|u_h^{n+1} - u_h^n + \omega_n(u_h^n - u_h^{n-1})\|^2 \\
&\leq \frac{1}{4}\|u_h^{n+1} - u_h^n\|^2 + \frac{1}{4}\|u_h^n - u_h^{n-1}\|^2 \\
&\quad - \frac{1}{8(1+\omega_n^2)}\|u_h^{n+1} - u_h^n + \omega_n(u_h^n - u_h^{n-1})\|^2
\end{aligned}$$

Combining like terms we then have

$$\frac{1}{2}\|u_h^{n+1}\|^2 - \frac{1}{2}\|u_h^n\|^2 + \frac{1}{4}\|u_h^{n+1} - u_h^n\|^2 - \frac{1}{4}\|u_h^n - u_h^{n-1}\|^2$$
$$+\frac{\nu\Delta t_n}{4}\|\nabla u_h^{n+1}\|^2 + \frac{\nu\Delta t_n}{2}\left(1 - \frac{C\Delta t_n(1+\omega_n^2)}{\nu h}\|\nabla E^{n+1}(u_h)\|^2\right)\|\nabla u_h^{n+1}\|^2$$
$$+\frac{1}{8(1+\omega_n^2)}\|u_h^{n+1} - u_h^n + \omega_n(u_h^n - u_h^{n-1})\|^2 \leq \frac{\Delta t_n}{\nu}\|f^{n+1}\|_{-1}^2.$$

Now using condition (4.11) letting $C = C_{stab}$ and summing from $n = 1$ to $N-1$ the result follows. □

There are several cases where the time step condition can be improved by using a different embedding for the nonlinear term. When $\Omega \subset \mathbb{R}^2$ the discrete Sobolev embedding will give a superior timestep condition to that in Theorem 4.4.1.

Theorem 4.4.2. *[2d Stability of VSS BE-AB2] Consider the method* (4.10) *and let* $\Omega \subset \mathbb{R}^2$. *Suppose that*

$$1 - \frac{C_{stab}\Delta t_n(1+\omega_n^2)|\ln h|}{\nu}\|\nabla E^{n+1}(u_h)\|^2 \geq 0. \tag{4.13}$$

Then the energy inequality (4.12) *from Theorem 4.4.1 holds.*

Proof. The proof is similar to that of Theorem 4.4.1, the key difference being in the treatment of the nonlinearity. Using Holders inequality for the nonlinear term we have

$$\begin{aligned}
&\Delta t_n b^*(E^{n+1}(u_h), u_h^{n+1}, u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})) \\
&\leq C\Delta t_n\|E^{n+1}(u_h)\|_\infty\|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\|\|\nabla u_h^{n+1}\| \\
&+ C\frac{\Delta t_n}{2}\|\nabla\cdot E^{n+1}(u_h)\|\|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\|\|u_h^{n+1}\|_\infty.
\end{aligned}$$

Then applying (4.4) and Cauchy-Schwarz-Young

$$\begin{aligned}
&C\Delta t_n\|E^{n+1}(u_h)\|_\infty\|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\|\|\nabla u_h^{n+1}\| \\
&+ C\frac{\Delta t_n}{2}\|\nabla\cdot E^{n+1}(u_h)\|\|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\|\|u_h^{n+1}\|_\infty \\
&\leq C|\ln h|^{1/2}\Delta t_n\|\nabla E^{n+1}(u_h)\|\|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\|\|\nabla u_h^{n+1}\| \\
&\leq C\Delta t_n^2(1+\omega_n^2)|\ln h|\|\nabla E^{n+1}(u_h)\|^2\|\nabla u_h^{n+1}\|^2 \\
&+ \frac{1}{8(1+\omega_n^2)}\|u_h^{n+1} - u_h^n - \omega_n(u_h^n - u_h^{n-1})\|^2.
\end{aligned}$$

The result then follows from Theorem 4.4.1. □

4.4.2 Energy stability for constant timestep Filtered-BE-AB2

In this section we give a stability result for the constant timestep filtered version of Algorithm 9.

Algorithm 10 (Constant stepsize filtered-BE-AB2)**.** *Given* Δt, (u_h^n, p_h^n), (u_h^{n-1}, p_h^{n-1}), *find* $(\hat{u}_h^{n+1}, p_h^{n+1})$ *satisfying*

$$\begin{aligned}\left(\frac{\hat{u}_h^{n+1}-u_h^n}{\Delta t}, v_h\right) + \nu(\nabla \hat{u}_h^{n+1}, \nabla v_h) + b^*(2u_h^n - u_h^{n-1}, 2u_h^n - u_h^{n-1}, v_h) \\ -(p_h^{n+1}, \nabla \cdot v_h) = (f^{n+1}, v_h) \qquad \forall v_h \in X_h \\ (\nabla \cdot \hat{u}_h^{n+1}, q_h) = 0 \qquad \forall q_h \in Q_h.\end{aligned} \tag{4.14}$$

Then compute

$$u_h^{n+1} = \hat{u}_h^{n+1} - \frac{1}{3}(\hat{u}_h^{n+1} - 2u_h^n + u_h^{n-1}) \tag{4.15}$$

Equivalently this can be written as

$$\begin{aligned}\left(\frac{\frac{3}{2}u_h^{n+1} - 2u_h^n + \frac{1}{2}u_h^{n-1}}{\Delta t}, v_h\right) + \nu\left(\nabla\left(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}\right), \nabla v_h\right) \\ +b^*(2u_h^n - u_h^{n-1}, 2u_h^n - u_h^{n-1}, v_h) - (p^{n+1}, \nabla \cdot v_h) = (f^{n+1}, v_h) \\ (\nabla \cdot \hat{u}_h^{n+1}, q_h) = 0\end{aligned} \tag{4.16}$$

In order to prove stability we will need to use the identity

Lemma 4.4.1. *The following identity holds*

$$\begin{aligned}\left(\frac{3}{2}a - 2b + \frac{1}{2}c, \frac{3}{2}a - b + \frac{1}{2}c\right) = \\ \left(\frac{\|a\|^2}{4} + \frac{\|2a-b\|^2}{4} + \frac{\|a-b\|^2}{4}\right) - \left(\frac{\|b\|^2}{4} + \frac{\|2b-c\|^2}{4} + \frac{\|b-c\|^2}{4}\right) \\ +\frac{3}{4}\|a - 2b + c\|^2\end{aligned}$$

We then have the following general conditional stability result. This result can be improved further in some cases, such as the 2D case. Stability and convergence of the VSS version of this method is currently an open problem.

Theorem 4.4.3. *Consider the method* (4.16) *and suppose that*

$$1 - \frac{C\Delta t}{\nu h}\|\nabla(2u_h^n - u_h^{n-1})\|^2 \geq 0. \tag{4.17}$$

Then for any $N > 1$

$$\begin{aligned}
&\frac{1}{4}\|u_h^N\|^2 + \frac{1}{4}\|2u_h^N - u_h^{N-1}\|^2 + \frac{1}{4}\|u_h^N - u_h^{N-1}\|^2 \\
&\quad + \frac{\nu\Delta t}{4}\sum_{n=1}^{N-1}\|\nabla(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1})\|^2 \\
&\leq \sum_{n=1}^{N-1}\frac{\Delta t}{\nu}\|f^{n+1}\|_{-1}^2 + \frac{1}{4}\|u_h^1\|^2 + \frac{1}{4}\|2u_h^1 - u_h^0\|^2 + \frac{1}{4}\|u_h^1 - u_h^0\|^2
\end{aligned}$$

Proof. Setting $v_h = \frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}$ multiplying by Δt, using lemma 4.4.1, and applying Young's inequality to the right hand side

$$\begin{aligned}
&\frac{1}{4}(\|u_h^{n+1}\|^2 + \|2u_h^{n+1} - u_h^n\|^2 + \|u_h^{n+1} - u_h^n\|^2) - \\
&\quad \frac{1}{4}(\|u_h^n\|^2 + \|2u_h^n - u_h^{n-1}\|^2 + \|u_h^n - u_h^{n-1}\|^2) + \\
&\frac{3}{4}\|u_h^{n+1} - 2u_h^n + u_h^{n-1}\|^2 + \Delta t\nu\|\nabla(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1})\|^2 + \\
&\quad \Delta t b^*(2u_h^n - u_h^{n-1}, 2u_h^n - u_h^{n-1}, \frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}) \\
&\quad \leq \frac{\nu\Delta t}{4}\|\nabla(\frac{3}{2}u_h^{n+1} - 2u_h^n + \frac{1}{2}u_h^{n-1})\|^2 + \frac{\Delta t}{\nu}\|f^{n+1}\|_{-1}^2.
\end{aligned}$$

Next dealing with the nonlinear term we use the skew symmetry of b^*, Poincar inequality, inequality (4.5), the inverse inequality and Young's inequality

$$\begin{aligned}
&\Delta t b^*(2u_h^n - u_h^{n-1}, 2u_h^n - u_h^{n-1}, \frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}) \\
&\quad = -\Delta t b^*(2u_h^n - u_h^{n-1}, \frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}, -2u_h^n + u_h^{n-1}) \\
&\quad = -\frac{3}{2}\Delta t b^*(2u_h^n - u_h^{n-1}, \frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1}, u_h^{n+1} - 2u_h^n + u_h^{n-1}) \\
&\quad \leq \frac{3}{2}C_{b^*}\Delta t\|\nabla(2u_h^n - u_h^{n-1})\|\|\nabla(\frac{3}{2}u_h^{n+1} - u_h^n - \frac{1}{2}u_h^{n-1})\| \\
&\|\nabla(u_h^{n+1} - 2u_h^n - u_h^{n-1})\|^{1/2}\|(u_h^{n+1} - 2u_h^n - u_h^{n-1})\|^{1/2} \\
&\quad \leq C\Delta t h^{-\frac{1}{2}}\|\nabla(2u_h^n - u_h^{n-1})\|\nabla(\frac{3}{2}u_h^{n+1} - u_h^n - \frac{1}{2}u_h^{n-1})\|\|(u_h^{n+1} - 2u_h^n - u_h^{n-1})\| \\
&\quad \leq C\frac{\Delta t^2}{h}\|\nabla(2u_h^n - u_h^{n-1})\|^2\|\nabla(\frac{3}{2}u_h^{n+1} - u_h^n - \frac{1}{2}u_h^{n-1})\|^2 \\
&\quad + \frac{3}{4}\|u_h^{n+1} - 2u_h^n + u_h^{n-1}\|^2.
\end{aligned}$$

Combining like terms we then have

$$\frac{1}{4}(\|u_h^{n+1}\|^2 + \|2u_h^{n+1} - u_h^n\|^2 + \|u_h^{n+1} - u_h^n\|^2) -$$
$$\frac{1}{4}(\|u_h^n\|^2 + \|2u_h^n - u_h^{n-1}\|^2 + \|u_h^n - u_h^{n-1}\|^2) +$$
$$\frac{\Delta t\nu}{4}\|\nabla(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1})\|^2 +$$
$$\frac{\nu\Delta t}{2}\left(1 - \frac{C\Delta t}{\nu h}\|\nabla(2u_h^n - u_h^{n-1})\|^2\right)\|\nabla(\frac{3}{2}u_h^{n+1} - u_h^n + \frac{1}{2}u_h^{n-1})\|^2$$
$$\leq \frac{\Delta t}{\nu}\|f^{n+1}\|_{-1}^2.$$

Now using condition (4.17) and summing from $n = 1$ to $N - 1$ the result follows. □

4.5 Concluding Remarks

We have given a VSVO IMEX algorithm of up to second order for the incompressible Navier-Stokes equations. This is the first attempt at analyzing such an algorithm to our knowledge. Such analysis is feeling around the boundary of what is currently possible/known for such methods, and as a result, there are many open problems. An error analysis of the variable stepsize first order method, not included in this dissertation, is underway. The most challenging question that we have not been able to answer with analysis is probably the stability of the variable stepsize second order method. Indeed, energy stability analysis of any variable stepsize two step is highly nontrivial to begin with. It is even more nontrivial for a nonlinear problem with mixed implicit and explicit treatments of the equation. Finally, there is the question of numerical testing, which is in progress.

5.0 Stability of Variable Stepsize BDF2

An essential step in the analysis of the VSVO methods developed in Chapter 3 is to analyze the variable stepsize stability of each standalone method. In particular, we are interested in variable stepsize stability of the BDF methods which form the basis of the methods. The difficulty of the analysis goes from trivially easy for BDF1 to extremely difficult for BDF2. Stability of variable stepsize BDF2 is the subject of many papers, and in this chapter, we expand on this knowledge by answering the following. What limit on the stepsize ratios ω is required to maintain A_0 stability.

Consider the following diffusion equation

$$u_t = -Au. \tag{5.1}$$

where A is SPD. We will assume A is an SPD matrix for the entire chapter, but extensions to PDEs will be obvious. Taking the inner product with u yields

$$\frac{d}{dt}\frac{1}{2}\|u\|^2 = -\|A^{1/2}u\|^2,$$

which implies l^2 stability. On the other hand, multiplying by u_t (as if for a gradient flow) and rearranging gives

$$\frac{d}{dt}\frac{1}{2}\|A^{1/2}u\|^2 = -\|u_t\|^2,$$

giving stability in the $A^{1/2}$ norm, which implies l^2 stability by coercivity. This is the core idea of our proof.

Discrete energy stability in the $A^{1/2}$ norm is known for BDF1-3 for constant stepsizes. For variable stepsizes, the stability results are naturally stated in terms of stepsize ratios $\omega_n = \Delta t_n/\Delta t_{n-1}$, and there has been steady progress at extending the upper bound of ω_n for BDF2 applied to a nonautonomous version of (5.1). For stability, results include Becker's bound of $(2+\sqrt{13})/3 \approx 1.86$ [11]. The largest upper bound to my knowledge is by Emmrich [32] in 2005 where he proved stability for stepsize ratios bounded by 1.91. The analysis for nonlinear diffusion is even more restrictive, and Emmrich additionally showed the error may depend badly on the fluctuation of adjacent stepsize ratios [33]. Nevertheless,

we are interested in analyzing stability properties in the worst cases of ω_n since practical computations may require large amplitudes and fluctuations of ω_n. See also [77] for recent results on 0-stability on non uniform grids. Unlike Emmrich, we restrict ourselves to the autonomous case which is the case that A_0-Stability pertains to.

Recall variable stepsize BDF2 as it is normally presented in the literature. Let $\Delta t_n = t^{n+1} - t^n$, $\omega_n = \frac{\Delta t_n}{\Delta t_{n-1}}$.

$$\frac{1}{\Delta t_n}\left(\frac{1+2\omega_n}{1+\omega_n}u^{n+1} - (1+\omega_n)u^n + \frac{\omega_n^2}{1+\omega_n}u^{n-1}\right) = -Au^{n+1}. \tag{5.2}$$

The BDFp methods are naturally given by weighted sums of divided differences up to order p, and it is insightful to rewrite BDF2 as follows.

Proposition 5.0.1. *Method* (5.2) *is equivalent to*

$$\begin{aligned}\frac{1}{\Delta t_n(1+\omega_n)}\Big(\omega_n^2\left(u^{n+1}-u^n-(u^n-u^{n-1})\right) + (-\omega_n^2+2\omega_n+1)\left(u^{n+1}-u^n\right)\Big) \\ = -Au^{n+1}.\end{aligned} \tag{5.3}$$

Taking the inner product with $u^{n+1} - u^n$ gives

$$\begin{aligned}\frac{1}{\Delta t_n(1+\omega_n)}\Big(&\frac{\omega_n^2}{2}\left(\|u^{n+1}-u^n\|^2 - \|u^n-u^{n-1}\|^2 + \|u^{n+1}-2u^n+u^{n-1}\|^2\right) \\ &+(-\omega_n^2+2\omega_n+1)\|u^{n+1}-u^n\|^2\Big) \\ = -\frac{1}{2}\|A^{1/2}u^{n+1}\|^2 &+ \frac{1}{2}\|A^{1/2}u^n\|^2 - \frac{1}{2}\|A^{1/2}(u^{n+1}-u^n)\|^2.\end{aligned} \tag{5.4}$$

We can highlight the essential difficulties of the proof. The time differences in the top line, which correspond to numerical dispersion, will *not* telescope for variable ω_n. The term on the second line, which corresponds to *physical* dissipation, will be positive provided $\omega_n \leq 1+\sqrt{2}$, which is the known limit for 0-Stability. Fortunately, the right hand side is independent of ω_n and Δt_n. Also, the numerical dispersion is actually *stabilizing* as it will allow the physical dissipation to be negative as long as it's balanced by the dispersion.

5.1 Stability

We prove an energy equality which is true independent of ω_n, but does not necessarily imply stability in Section 5.1.1. We then derive *sufficient* conditions for which the energy equality implies stability of the solution in the $A^{1/2}$ norm, and therefore stability in l^2.

5.1.1 Energy equality

Theorem 5.1.1 (Energy Equality)**.** *Let*

$$\alpha^{n+1} = \frac{-\omega_n^2 + 2\omega_n + 1}{\Delta t_n(1+\omega_n)} \qquad \beta^{n+1} = \frac{\omega_n^2}{\Delta t_n(1+\omega_n)}. \tag{5.5}$$

Then for all $N \geq 3$, the following equality holds,

$$\begin{aligned} &\frac{1}{2}\|A^{1/2}u^N\|^2 + \left(\alpha^N + \frac{\beta^N}{2}\right)\|u^N - u^{N-1}\|^2 \\ &+ \sum_{n=1}^{N-1}\frac{1}{2}\|A^{1/2}(u^{n+1}-u^n)\|^2 + \sum_{n=2}^{N-1}\left(\alpha^n + \frac{1}{2}\beta^n - \frac{1}{2}\beta^{n+1}\right)\|u^n - u^{n-1}\|^2 \\ &+ \sum_{n=1}^{N-1}\frac{1}{2}\beta^{n+1}\|u^{n+1} - 2u^n + u^{n-1}\|^2 = \frac{1}{2}\|A^{1/2}u^0\|^2 + \frac{\beta^2}{2}\|u^1 - u^0\|^2. \end{aligned} \tag{5.6}$$

For clarity, we prove an identity as a separate Lemma as it may have more universal application.

Lemma 5.1.1. *For every sequence $\{a_n\}_n$, $\{b_n\}_n$, and $\{c_n\}_n$, and $N \geq 3$, we have the identity*

$$\begin{aligned} &\sum_{n=1}^{N-1}\left(a^{n+1}c^{n+1} + b^{n+1}(c^{n+1}-c^n)\right)c^{n+1} \\ &= \left(a^N + \frac{b^N}{2}\right)(c^N)^2 + \sum_{n=2}^{N-1}\left(a^n + \frac{1}{2}b^n - \frac{1}{2}b^{n+1}\right)(c^n)^2 \\ &\quad + \sum_{n=1}^{N-1} b^{n+1}(c^{n+1}-c^n)^2 - \frac{b^2}{2}(c^1)^2. \end{aligned} \tag{5.7}$$

Proof. We have for the $n = 1$ term

$$\left(a^2c^2 + b^2(c^2 - c^1)\right)c^2 = a^2(c^2)^2 + b^2\left(\frac{1}{2}(c^2)^2 - \frac{1}{2}(c^1)^2 + \frac{1}{2}(c^2 - c^1)^2\right).$$

Adding this equation for $n = 1$ and $n = 2$ and grouping like terms yields the $N = 3$ case

$$\begin{aligned}
&\sum_{n=1}^{3-1}\left(a^{n+1}c^{n+1} + b^{n+1}(c^{n+1} - c^n)\right)c^{n+1} \\
&= \left(a^3 + \frac{b^3}{2}\right)(c^3)^2 + \left(a^2 + \frac{1}{2}b^2 - \frac{1}{2}b^3\right)(c^2)^2 - \frac{b^2}{2}(c^1)^2 \\
&\quad + \frac{1}{2}b^2(c^2 - c^1)^2 + \frac{1}{2}b^3(c^3 - c^2)^2 \\
&= \left(a^3 + \frac{b^3}{2}\right)(c^3)^2 + \sum_{n=2}^{3-1}\left(a^n + \frac{1}{2}b^n - \frac{1}{2}b^{n+1}\right)(c^n)^2 \\
&\quad + \sum_{n=1}^{3-1} b^{n+1}(c^{n+1} - c^n)^2 - \frac{b^2}{2}(c^1)^2.
\end{aligned}$$

An induction argument yields the result. □

Now we prove Theorem 5.1.1.

Proof of Theorem 5.1.1. Taking the inner product of (5.3) with $u^{n+1} - u^n$ gives

$$\begin{aligned}
&\frac{1}{2}\|A^{1/2}u^{n+1}\|^2 - \frac{1}{2}\|A^{1/2}u^n\|^2 + \frac{1}{2}\|A^{1/2}(u^{n+1} - u^n)\|^2 \\
&\quad + \Bigg(\frac{1}{\Delta t_n(1+\omega_n)}\Bigg(\omega_n^2\left(u^{n+1} - u^n - (u^n - u^{n-1})\right) \\
&\quad + (-\omega_n^2 + 2\omega_n + 1)\left(u^{n+1} - u^n\right)\Bigg), u^{n+1} - u^n\Bigg) = 0.
\end{aligned}$$

Summing from $n = 1$ to $N - 1$ gives

$$\begin{aligned}
&\frac{1}{2}\|A^{1/2}u^N\|^2 + \sum_{n=1}^{N-1}\frac{1}{2}\|A^{1/2}(u^{n+1} - u^n)\|^2 \\
&+ \sum_{n=1}^{N-1}\Bigg(\frac{1}{\Delta t_n(1+\omega_n)}\Bigg(\omega_n^2\left(u^{n+1} - u^n - (u^n - u^{n-1})\right) \\
&+ (-\omega_n^2 + 2\omega_n + 1)\left(u^{n+1} - u^n\right)\Bigg), u^{n+1} - u^n\Bigg) = \frac{1}{2}\|A^{1/2}u^1\|^2.
\end{aligned}$$

Recalling (5.5), let $a^{n+1} = \alpha^{n+1}$, $b^{n+1} = \beta^{n+1}$, and $c^{n+1} = u^{n+1} - u^n$. Then

$$\frac{1}{2}\|A^{1/2}u^N\|^2 + \sum_{n=1}^{N-1}\frac{1}{2}\|A^{1/2}(u^{n+1}-u^n)\|^2$$
$$+\sum_{n=1}^{N-1}\left(a^{n+1}c^{n+1} + b^{n+1}(c^{n+1}-c^n)\right)c^{n+1} = \frac{1}{2}\|A^{1/2}u^1\|^2.$$

If $N \geq 3$, we can apply Lemma 5.1.1, which completes the proof. □

5.1.2 Main result

The energy equality derived in Section 5.1 is not sufficient for stability without applying some restrictions on ω_n.

Theorem 5.1.2. *Let A be SPD, and the stepsize ratios $\omega_n < (3+\sqrt{13})/2$. Then solutions to variable stepsize BDF2 are bounded uniformly in n by initial data.*

Proof. From Theorem 5.1.1, we observe that a sufficient condition for boundedness is that

$$\alpha^n + \frac{1}{2}\beta^n - \frac{1}{2}\beta^{n+1} \geq 0 \qquad \text{for all } n \geq 1.$$

Indeed, since β^n is always positive, this implies positivity of every term in the energy equality (5.6), and coercivity of A implies boundedness of $\|u\|$ in both the l^2 and $A^{1/2}$ norms.

By definition (5.5), this means

$$\frac{-\omega_n^2 + 2\omega_n + 1}{\Delta t_n(1+\omega_n)} + \frac{\omega_n^2/2}{\Delta t_n(1+\omega_n)} - \frac{\omega_{n+1}^2/2}{\Delta t_{n+1}(1+\omega_{n+1})} \geq 0.$$

Multiplying by Δt_n,

$$\frac{-\omega_n^2 + 2\omega_n + 1}{1+\omega_n} + \frac{\omega_n^2/2}{1+\omega_n} - \frac{\omega_{n+1}/2}{1+\omega_{n+1}} \geq 0.$$

and combining like terms,

$$\frac{2\omega_n + 1 - \omega_n^2/2}{1+\omega_n} - \frac{\omega_{n+1}/2}{1+\omega_{n+1}} \geq 0.$$

Multiplying by 2 and rearranging gives

$$\frac{4\omega_n + 2 - \omega_n^2}{1+\omega_n} \geq \frac{\omega_{n+1}}{1+\omega_{n+1}}.$$

Multiplying by $(1+\omega_{n+1})(1+\omega_n)$ (which is always positive) and rearranging gives

$$4\omega_n + 2 - \omega_n^2 \geq \omega_{n+1}(\omega_n^2 - 3\omega_n - 1).$$

$\omega_n^2 - 3\omega_n - 1$ has roots $(3-\sqrt{13})/2 \approx -.3$ and $(3+\sqrt{13})/2 \approx 3.3$. Therefore, $\omega_n^2 - 3\omega_n - 1$ is negative for $\omega_n < (3+\sqrt{13})/2$, and

$$\omega_{n+1} \geq \frac{4\omega_n + 2 - \omega_n^2}{\omega_n^2 - 3\omega_n - 1}.$$

$4\omega_n+2-\omega_n^2$ has roots $2\pm\sqrt{6}$, and is therefore positive in that interval. Since $[0, (3+\sqrt{13})/2) \in (2-\sqrt{6}, 2+\sqrt{6})$, we actually have

$$\omega_{n+1} \geq 0 \geq \frac{4\omega_n + 2 - \omega_n^2}{\omega_n^2 - 3\omega_n - 1},$$

which is trivially satisfied by ω_{n+1}. □

This result is independent of A and the dimension. In fact, observing the second line of (5.6), and using $\|A^{1/2}u\|^2 \geq \lambda\|u\|^2$ where $\lambda = \lambda_{min}$ shows that we could use a weaker assumption

$$\alpha^n + \frac{1}{2}\beta^n - \frac{1}{2}\beta^{n+1} + \frac{\lambda}{2} \geq 0 \qquad \text{for all } n \geq 1$$

In fact, one can show that solutions are bound by initial data if eventually all $\Delta t_n \geq \frac{1}{\lambda}$!

5.2 Damping Rate

We derive and briefly analyze a specific modified equation for VSSBDF2 applied to the Dahlquist problem $y' = \lambda y$. That is, we seek an ODE depending on Δt of the form $y' = \lambda + \Delta t^2 g(y)$ for which VSSBDF2 is actually third order consistent.

For VSSBDF2 applied to $y' = \lambda y$, we have

$$\begin{aligned}
&\frac{1}{\Delta t_n}\left(\frac{1+2\omega_n}{1+\omega_n}y(t^{n+1}) - (1+\omega_n)y(t^n) + \frac{\omega_n^2}{1+\omega_n}y(t^{n-1})\right) - y(t^{n+1}) \\
&\quad = \mathcal{O}((\Delta t_n + \Delta t_{n-1})^3) - \lambda y(t^{n+1}) + \frac{1}{\Delta t_n}\Bigg(\frac{1+2\omega_n}{1+\omega_n}y^{n+1} - \\
&-(1+\omega_n)\left((y(t^{n+1}) - \Delta t_n y'(t^{n+1}) + \frac{\Delta t_n^2}{2}y''(t^{n+1}) - \frac{\Delta t^3}{6}y'''(t^{n+1})\right) \\
&\quad + \frac{\omega_n^2}{1+\omega_n}\Bigg((y(t^{n+1}) - (\Delta t_n + \Delta t_{n-1})y'(t^{n+1}) \\
&\quad + \frac{(\Delta t_n + \Delta t_{n-1})^2}{2}y''(t^{n+1}) - \frac{(\Delta t_n + \Delta t_{n-1})^3}{6}y'''(t^{n+1})\Bigg) \\
&= y'(t^{n+1}) - \frac{\Delta t_n^2(1+\omega_n)}{6\omega_n}y'''(t^{n+1})) - \lambda y(t^{n+1}) + \mathcal{O}((\Delta t_n + \Delta t_{n-1})^3).
\end{aligned}$$

Therefore, we see that VSSBDF2 applied to $y' = \lambda y$ results in a *third* order consistent approximation to

$$y' - \frac{\Delta t_n^2(1+\omega_n)}{6\omega_n}y''' - \lambda y = 0.$$

We can reduce the order of the modified equation by taking two derivatives, and solving for y''',

$$y''' = \frac{\Delta t_n^2(1+\omega_n)}{6\omega_n}y^{(5)} + \lambda y''$$

Plugging this back into the modified equation gives

$$\begin{aligned}
0 &= y' - \frac{\Delta t_n^2(1+\omega_n)}{6\omega_n}\left(\frac{\Delta t_n^2(1+\omega_n)}{6\omega_n}y^{(5)} + \lambda y''\right) - \lambda y \\
&= y' - \frac{\lambda \Delta t_n^2(1+\omega_n)}{6\omega_n}y'' - \lambda y + \mathcal{O}((\Delta t_n + \Delta t_{n-1})^4),
\end{aligned}$$

so that another third order consistent modified equation is

$$y' - \frac{\lambda \Delta t_n^2(1+\omega_n)}{6\omega_n}y'' - \lambda y = 0. \tag{5.8}$$

Multiplying by y' and assuming $\lambda < 0$ gives

$$\frac{d}{dt}\left(\frac{1}{2}|\lambda|y^2 + \frac{|\lambda|\Delta t_n^2(1+\omega_n)}{12\omega_n}(y')^2\right) = -(y')^2,$$

This provides an intuitive explanation for the difficulty of proving a decrease of energy for VSSBDF2. The modified equation has a potential energy $\frac{1}{2}\lambda(y)^2$ and a pseudo kinetic energy

$$\text{Numerical Kinetic Energy} = \frac{|\lambda|\Delta t_n^2(1+\omega_n)}{12\omega_n}(y')^2$$

that can cause oscillations if Δt is not small enough to control λ, and if a lot of energy is stored in y'.

To quantify this further, note (5.8) can also be written

$$y'' - \frac{6\omega_n}{\lambda\Delta t_n^2(1+\omega_n)}y' + \frac{6\omega_n}{\Delta t_n^2(1+\omega_n)} = 0. \tag{5.9}$$

which is a damped harmonic oscillator with frequency $\mathcal{F}_n$ and damping ratio ξ_n given by

$$\mathcal{F}_n = \frac{1}{\Delta t_n}\sqrt{\frac{6\omega_n}{1+\omega_n}} \qquad \xi_n = \frac{-1}{\lambda\Delta t_n}\sqrt{\frac{3\omega_n}{2(1+\omega_n)}}$$

The system returns to steady state without oscillation if $\xi_n \geq 1$, which is the desired behavior of the original system. This can be written as a CFL like condition,

$$-\frac{1}{\lambda}\sqrt{\frac{3}{2}} \geq \Delta t_n\sqrt{\frac{\omega_n}{1+\omega_n}} \qquad \left(-\frac{\sqrt{3}}{\lambda} \geq \Delta t \text{ for constant stepsize}\right).$$

Otherwise we expect oscillations (overshooting) on the return to steady state.

Appendix A

Second Order Method Supplementary Material

A.1 Velocity Error Analysis for Backward Euler Plus Filter

A.1.1 Proof of Lemma 2.4.1

Proof. By Taylor's theorem with the integral remainder,

$$\begin{aligned} D[u(t^{n+1})] - \Delta t u_t(t^{n+1}) &= \frac{3}{2}u(t^{n+1}) - \Delta t u_t(t^{n+1}) \\ -2\left(u(t^{n+1}) - \Delta t u_t(t^{n+1}) + \frac{\Delta t^2}{2}u_{tt}(t^{n+1})) + \frac{1}{2}\int_{t^{n+1}}^{t^n} u_{ttt}(t)(t^n - t)^2 dt\right) \\ +\frac{1}{2}\left(u(t^{n+1}) - 2\Delta t u_t(t^{n+1}) + 2\Delta t^2 u_{tt}(t^{n+1})) + \frac{1}{2}\int_{t^{n+1}}^{t^{n-1}} u_{ttt}(t)(t^{n-1} - t)^2 dt\right) \\ = \int_{t^n}^{t^{n+1}} u_{ttt}(t^n - t)^2 dt - \frac{1}{4}\int_{t^{n-1}}^{t^{n+1}} u_{ttt}(t^{n-1} - t)^2 dt. \end{aligned}$$

These terms are first estimated by Cauchy-Schwarz.

$$\left(\int_{t^n}^{t^{n+1}} u_{ttt}(t)(t^n - t)^2 dt\right)^2 \leq \int_{t^n}^{t^{n+1}} u_{ttt}^2 dt \int_{t^n}^{t^{n+1}} (t^n - t)^4 dt = \frac{\Delta t^5}{5}\int_{t^n}^{t^{n+1}} u_{ttt}^2 dt.$$

$$\frac{1}{16}\left(\int_{t^{n-1}}^{t^{n+1}} u_{ttt}(t)(t^{n-1} - t)^2 dt\right)^2 \leq \frac{1}{16}\int_{t^{n-1}}^{t^{n+1}} u_{ttt}^2 dt \int_{t^{n-1}}^{t^{n+1}} (t^{n-1} - t)^4 dt = \frac{2\Delta t^5}{5}\int_{t^{n-1}}^{t^{n+1}} u_{ttt}^2 dt.$$

Thus,

$$\left(\frac{D[u(t^{n+1})]}{\Delta t} - u_t(t^{n+1})\right)^2 \leq \frac{6}{5}\Delta t^3 \int_{t^{n-1}}^{t^{n+1}} u_{ttt}^2 dt.$$

Integrating with respect to x yields the first inequality. Next,

$$I[u(t^{n+1})] - u(t^{n+1}) = \frac{1}{2}u(t^{n+1}) - u(t^n) + \frac{1}{2}u(t^{n-1})$$
$$= \frac{1}{2}\int_{t^n}^{t^{n+1}} u_{tt}(t)(t^{n+1}-t)dt + \frac{1}{2}\int_{t^n}^{t^{n-1}} u_{tt}(t)(t^{n-1}-t)dt.$$

By similar steps,

$$\left(\int_{t^n}^{t^{n+1}} u_{tt}(t)(t^{n+1}-t)dt\right)^2 \leq \frac{\Delta t^3}{3}\int_{t^n}^{t^{n+1}} u_{tt}^2 dt.$$

$$\left(\int_{t^{n-1}}^{t^n} u_{tt}(t)(t^{n-1}-t)dt\right)^2 \leq \frac{\Delta t^3}{3}\int_{t^{n-1}}^{t^n} u_{tt}^2 dt.$$

Therefore,

$$\left(I[u(t^{n+1})] - u(t^{n+1})\right)^2 \leq \frac{\Delta t^3}{6}\int_{t^{n-1}}^{t^{n+1}} u_{tt}^2 dt. \tag{A.1}$$

The last inequality can be proved using the same strategy. □

A.1.2 Proof of Theorem 2.4.2

Proof. We prove this for Option A. A parallel proof exists for Option B. At $t^{n+1} = (n+1)\Delta t$, the true solution of (2.1) satisfies,

$$\begin{aligned}&\left(\frac{D[u(t^{n+1})]}{\Delta t}, v_h\right) + b\left(I[u(t^{n+1})], I[u(t^{n+1})], v_h\right)\\ &+ \nu\left(\nabla I[u(t^{n+1})], \nabla v_h\right) - \left(p(t^{n+1}), \nabla\cdot v_h\right)\\ &= \left(\mathbf{f}^{n+1}, v_h\right) + \tau^{n+1}(u,p;v_h) \quad \forall v_h \in X_h.\end{aligned} \tag{A.2}$$

Subtracting (2.11) from $(A.2)$ yields

$$\begin{aligned}&\left(\frac{D[e_u^{n+1}]}{\Delta t}, v_h\right) + b\left(I[e_u^{n+1}], I[u(t^{n+1})], v_h\right)\\ &+ b\left(I[u_h^{n+1}], I[e_u^{n+1}], v_h\right) + \nu\left(\nabla I[e_u^{n+1}], \nabla v_h\right)\\ &- \left(e_p^{n+1}, \nabla\cdot v_h\right) = \tau^{n+1}(u,p;v_h).\end{aligned} \tag{A.3}$$

Decompose the error equation for velocity

$$u(t_{n+1}) - u_h^{n+1} = (u^{n+1} - \tilde{u}_h^{n+1}) + (\tilde{u}_h^{n+1} - u_h^{n+1}) = \eta^{n+1} + \phi_h^{n+1}. \tag{A.4}$$

where $\tilde{u}_h^{n+1}$ is the best approximation of $u(t^{n+1})$ in X_h.

Set $v_h = I[\phi_h^{n+1}]$. Using the identity (2.9) with $a = \phi_h^{n+1}$, $b = \phi_h^n$, $c = \phi_h^{n-1}$, (A.4), and applying $(\lambda_h, \nabla \cdot \phi_h) = 0$ for all $\lambda_h \in V_h$, equation $(A.3)$ can be written

$$\begin{aligned}
&\frac{1}{4\Delta t}(\|\phi_h^{n+1}\|^2 + \|2\phi_h^{n+1} - \phi_h^n\|^2 + \|\phi_h^{n+1} - \phi_h^n\|^2) \\
&- \frac{1}{4\Delta t}(\|\phi_h^n\|^2 + \|2\phi_h^n - \phi_h^{n-1}\|^2 + \|\phi_h^n - \phi_h^{n-1}\|^2) \\
&+ \frac{3}{4\Delta t}\|\phi_h^{n+1} - 2\phi_h^n + \phi_h^{n-1}\|^2 + \nu\|\nabla I[\phi_h^{n+1}]\|^2 \\
&= -\left(\frac{D[\eta^{n+1}]}{\Delta t}, I[\phi_h^{n+1}]\right) - b\left(I[\phi_h^{n+1}], I[u(t^{n+1})], I[\phi_h^{n+1}]\right) \\
&- b\left(I[u_h^{n+1}], I[\eta^{n+1}], I[\phi_h^{n+1}]\right) - b\left(I[\eta^{n+1}], I[u(t^{n+1})], I[\phi_h^{n+1}]\right) \\
&+ \left(p(t^{n+1}) - \lambda_h^{n+1}, \nabla \cdot I[\phi_h^{n+1}]\right) - \nu\left(\nabla I[\eta^{n+1}], \nabla I[\phi_h^{n+1}]\right) \\
&+ \tau^{n+1}(u, p; I[\phi_h^{n+1}]).
\end{aligned} \tag{A.5}$$

The next step in the proof is to bound all the terms on the right hand side of $(A.5)$ and absorb terms into the left hand side. For arbitrary $\varepsilon > 0$, the first term on the right hand side of (A.5) is bounded in the following way,

$$-\left(\frac{D[\eta^{n+1}]}{\Delta t}, I[\phi_h^{n+1}]\right) \le \frac{1}{4\varepsilon}\left\|\frac{D[\eta^{n+1}]}{\Delta t}\right\|_{-1}^2 + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2. \tag{A.6}$$

The first nonlinear term can be bounded as

$$\begin{aligned}
&- b\left(I[\phi_h^{n+1}], I[u(t^{n+1})], I[\phi_h^{n+1}]\right) \le C\|I[\phi_h^{n+1}]\|\|I[u(t^{n+1})]\|_2\|\nabla I[\phi_h^{n+1}]\| \\
&\le \frac{C^2}{4\varepsilon}\|I[\phi_h^{n+1}]\|^2\|I[u(t^{n+1})]\|_2^2 + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2.
\end{aligned} \tag{A.7}$$

The second nonlinear term is estimated by rewriting it using (A.4) as follows

$$\begin{aligned}
&- b\left(I[u_h^{n+1}], I[\eta^{n+1}], I[\phi_h^{n+1}]\right) = -b\left(I[u(t^{n+1})], I[\eta^{n+1}], I[\phi_h^{n+1}]\right) \\
&+ b\left(I[\eta^{n+1}], I[\eta^{n+1}], I[\phi_h^{n+1}]\right) + b\left(I[\phi_h^{n+1}], I[\eta^{n+1}], I[\phi_h^{n+1}]\right).
\end{aligned} \tag{A.8}$$

then find bounds for all terms on the right hand side of (A.8). We bound the third nonlinear term in (A.5) the same way as the first nonlinear term in (A.8).

$$\begin{aligned}
&- b\left(I[u(t^{n+1})], I[\eta^{n+1}], I[\phi_h^{n+1}]\right) \\
&\le C\|\nabla I[u(t^{n+1})]\|\|\nabla I[\eta^{n+1}]\|\|\nabla I[\phi_h^{n+1}]\| \\
&\le \frac{C^2}{4\varepsilon}\|u\|_{\infty,1}^2\|\nabla I[\eta^{n+1}]\|^2 + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2,
\end{aligned} \tag{A.9}$$

and

$$b\left(I[\eta^{n+1}], I[\eta^{n+1}], I[\phi_h^{n+1}]\right) \leq \frac{C^2}{4\varepsilon}\|\nabla I[\eta^{n+1}]\|^4 + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2. \tag{A.10}$$

Next, by the inverse inequality and approximation assumptions,

$$\begin{aligned}
&b\left(I[\phi_h^{n+1}], I[\eta^{n+1}], I[\phi_h^{n+1}]\right) \\
&\leq C\|\nabla I[\phi_h^{n+1}]\|\|\nabla I[\eta^{n+1}]\|\|\nabla I[\phi_h^{n+1}]\| \\
&\leq Ch^{-1}\|I[\phi_h^{n+1}]\|\|\nabla I[\eta^{n+1}]\|\|\nabla I[\phi_h^{n+1}]\| \\
&\leq C\|I[\phi_h^{n+1}]\|\|I[u(t^{n+1})]\|_2\|\nabla I[\phi_h^{n+1}]\| \\
&\leq \frac{C^2}{4\varepsilon}\|I[\phi_h^{n+1}]\|^2\|I[u(t^{n+1})]\|_2^2 + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2.
\end{aligned} \tag{A.11}$$

The pressure can be bounded as follows

$$\left(p(t^{n+1}) - \lambda_h^{n+1}, \nabla \cdot I[\phi_h^{n+1}]\right) \leq \frac{C^2}{4\varepsilon}\|p(t^{n+1}) - \lambda_h^{n+1}\|^2 + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2. \tag{A.12}$$

Then we can bound the term after the pressure,

$$-\nu\left(\nabla I[\eta^{n+1}], \nabla(I[\phi_h^{n+1}])\right) \leq \frac{\nu}{2}\|\nabla I[\eta^{n+1}]\|^2 + \frac{\nu}{2}\|\nabla I[\phi_h^{n+1}]\|^2. \tag{A.13}$$

Next we will bound all components of the consistency error $\tau^{n+1}(u, p; I[\phi_h^{n+1}])$.

$$\begin{aligned}
&\left(\frac{D[u(t^{n+1})]}{\Delta t} - u_t(t^{n+1}), I[\phi_h^{n+1}]\right) \\
&\leq C\|\frac{D[u(t^{n+1})]}{\Delta t} - u_t(t^{n+1})\|\|\nabla I[\phi_h^{n+1}]\| \\
&\leq \frac{C^2}{4\varepsilon}\|\frac{D[u(t^{n+1})]}{\Delta t} - u_t(t^{n+1})\|^2 + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2.
\end{aligned} \tag{A.14}$$

$$\begin{aligned}
&\nu\left(\nabla(I[u(t^{n+1})] - u(t^{n+1})), \nabla I[\phi_h^{n+1}]\right) \\
&\leq \frac{C^2}{4\varepsilon}\|\nabla(I[u(t^{n+1})] - u(t^{n+1}))\|^2 + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2.
\end{aligned} \tag{A.15}$$

The nonlinear term in $\tau^{n+1}(u, p; I[\phi_h^{n+1}])$ is then estimated as follows,

$$\begin{aligned}
&b\left(I[u(t^{n+1})], I[u(t^{n+1})], I[\phi_h^{n+1}]\right) - b(u(t^{n+1}), u(t^{n+1}), I[\phi_h^{n+1}]) \\
&= b\left(I[u(t^{n+1})] - u(t^{n+1}), I[u(t^{n+1})], I[\phi_h^{n+1}]\right) + b(u(t^{n+1}), I[u(t^{n+1})] - u(t^{n+1}), I[\phi_h^{n+1}]) \\
&\leq C\|\nabla(I[u(t^{n+1})] - u(t^{n+1}))\|\|\nabla I[\phi_h^{n+1}]\|\Big(\|\nabla I[u(t^{n+1})]\| + \|\nabla u(t^{n+1})\|\Big) \\
&\leq \frac{C^2}{4\varepsilon}\|\nabla(I[u(t^{n+1})] - u(t^{n+1}))\|^2\Big(\|\nabla I[u(t^{n+1})]\|^2 + \|\nabla u(t^{n+1})\|^2\Big) + \varepsilon\|\nabla I[\phi_h^{n+1}]\|^2.
\end{aligned} \tag{A.16}$$

Set $\varepsilon = \frac{\nu}{40}$. Using $(A.6)$ to $(A.13)$ in (A.5) yields

$$
\begin{aligned}
&\frac{1}{4\Delta t}(\|\phi_h^{n+1}\|^2 + \|2\phi_h^{n+1} - \phi_h^n\|^2 + \|\phi_h^{n+1} - \phi_h^n\|^2) + \frac{\nu}{4}\|\nabla I[\phi_h^{n+1}]\|^2 \\
&- \frac{1}{4\Delta t}(\|\phi_h^{n}\|^2 + \|2\phi_h^{n} - \phi_h^{n-1}\|^2 + \|\phi_h^{n} - \phi_h^{n-1}\|^2) + \frac{3}{4\Delta t}\|\phi_h^{n+1} - 2\phi_h^n + \phi_h^{n-1}\|^2 \\
&\le C\nu^{-1}\Big(\|\frac{D[\eta^{n+1}]}{\Delta t}\|_{-1}^2 + \|I[\phi_h^{n+1}]\|^2\|I[u(t^{n+1})]\|_2^2 \\
&+ \|\|u\|\|_{\infty,1}\|\nabla I[\eta^{n+1}]\|^2 + \|\nabla I[\eta^{n+1}]\|^4 + \|p(t^{n+1}) - \lambda_h^{n+1}\|^2 \\
&+ \nu^2\|\nabla I[\eta^{n+1}]\|^2 + \|\frac{D[u(t^{n+1})]}{\Delta t} - u_t(t^{n+1})\|^2 \\
&+ \|\nabla(I[u(t^{n+1})] - u(t^{n+1}))\|^2 \\
&+ \|\nabla(I[u(t^{n+1})] - u(t^{n+1}))\|^2(\|\nabla I[u(t^{n+1})]\|^2 + \|\nabla u(t^{n+1})\|^2)\Big).
\end{aligned}
\tag{A.17}
$$

Let $\kappa = C\frac{\|u\|_{\infty,2}^2}{\nu}$. Assume $\Delta t < \frac{1}{\kappa}$, summing from $n = 1$ to $n = N-1$ and applying the discrete Gronwall lemma we obtain

$$
\begin{aligned}
&\|\phi_h^N\|^2 + \|2\phi_h^N - \phi_h^{N-1}\|^2 + \|\phi_h^N - \phi_h^{N-1}\|^2 \\
&+ \sum_{n=1}^{N-1} 3\|\phi_h^{n+1} - 2\phi_h^n + \phi_h^{n-1}\|^2 + \nu\Delta t\sum_{n=1}^{N-1}\|\nabla I[\phi_h^{n+1}]\|^2 \\
&\le e^{\left(\frac{\Delta t\kappa(N-1)}{1-\Delta t\kappa}\right)}\Big(\|\phi_h^1\|^2 + \|2\phi_h^1 - \phi_h^0\|^2 + \|\phi_h^1 - \phi_h^0\|^2 + C\Delta t\sum_{n=1}^{N-1}\|\frac{D[\eta^{n+1}]}{\Delta t}\|_{-1}^2 \\
&+ C\Delta t(\|u\|_{\infty,1}^2 + \nu^2)\sum_{n=1}^{N-1}\|\nabla I[\eta^{n+1}]\|^2 + C\Delta t\sum_{n=1}^{N-1}\|\nabla I[\eta^{n+1}]\|^4 \\
&+ C\Delta t\sum_{n=1}^{N-1}\|p(t^{n+1}) - \lambda_h^{n+1}\|^2 + C\Delta t\sum_{n=1}^{N-1}\|\frac{D[u(t^{n+1})]}{\Delta t} - u_t(t^{n+1})\|^2 \\
&+ C\Delta t\sum_{n=1}^{N-1}\|\nabla(I[u(t^{n+1})] - u(t^{n+1}))\|^2 \\
&+ C\Delta t\sum_{n=1}^{N-1}\|\nabla(I[u(t^{n+1})] - u(t^{n+1}))\|^2(\|\nabla I[u(t^{n+1})]\|^2 + \|\nabla u(t^{n+1})\|^2)\Big).
\end{aligned}
\tag{A.18}
$$

The first three terms can be bounded as

$$
\begin{aligned}
&\|\phi_h^1\|^2 + \|2\phi_h^1 - \phi_h^0\|^2 + \|\phi_h^1 - \phi_h^0\|^2 \\
&\le C\Big(\|u(t_1) - u_h^1\|^2 + \|(u(t_0) - u_h^0)\|^2\Big) + Ch^{2k+2}\|\|u\|\|_{\infty,k+1}.
\end{aligned}
\tag{A.19}
$$

We bound the fourth term in (A.18) as follows

$$\begin{aligned}\Delta t\sum_{n=1}^{N-1}\|\frac{D[\eta^{n+1}]}{\Delta t}\|_{-1}^2 &= \Delta t\sum_{n=1}^{N-1}\|\frac{\frac{3}{2}(\eta^{n+1}-\eta^n)-\frac{1}{2}(\eta^n-\eta^{n-1})}{\Delta t}\|_{-1}^2\\ &\le C\sum_{n=0}^{N}\int_{t^{n-1}}^{t^{n+1}}\|\eta_t\|^2 ds \le Ch^{2k+2}\|u_t\|_{2,k+1}^2,\end{aligned} \tag{A.20}$$

and

$$\begin{aligned}&\Delta t(\|u\|_{\infty,1}^2+\nu^2)\sum_{n=1}^{N-1}\|\nabla I[\eta^{n+1}]\|^2\\ &\le C\Delta t(\|u\|_{\infty,1}^2+\nu^2)\max\left\{\frac{9}{4},1,\frac{1}{4}\right\}\sum_{n=1}^{N-1}3\left(\|\nabla\eta^{n+1}\|^2+\|\nabla\eta^n\|^2+\|\nabla\eta^{n-1}\|^2\right)\\ &\le C\Delta t\sum_{n=0}^{N}h^{2k}\|u^{n+1}\|_{k+1}^2 = Ch^{2k}|||u|||_{2,k+1}^2.\end{aligned} \tag{A.21}$$

Similarly to (A.21), we also have

$$\Delta t\sum_{n=1}^{N-1}\|\nabla I[\eta^{n+1}]\|^4 \le C\Delta t\sum_{n=0}^{N}h^{4k}\|u(t^{n+1})\|_{k+1}^4 = Ch^{4k}|||u|||_{4,k+1}^4. \tag{A.22}$$

Observe that

$$\Delta t\sum_{n=1}^{N}\|p(t^{n+1})-\lambda_h^{n+1}\|^2 \le Ch^{2s+2}|||p|||_{2,s+1}^2. \tag{A.23}$$

The terms from consistency error are bounded using Lemma 2.4.1.

$$\Delta t\sum_{n=1}^{N-1}\|\frac{D[u(t^{n+1})]}{\Delta t}-u_t(t^{n+1})\|^2 \le C\Delta t^4\sum_{n=0}^{N-1}\int_{t^{n-1}}^{t^{n+1}}\|u_{ttt}\|^2dt \le C\Delta t^4\|u_{ttt}\|_{2,0}^2. \tag{A.24}$$

$$\Delta t\sum_{n=1}^{N-1}\|\nabla(I[u(t^{n+1})]-u(t^{n+1}))\|^2 \le C\Delta t^4\sum_{n=1}^{N-1}\int_{t^{n-1}}^{t^{n+1}}\|\nabla u_{tt}\|^2dt \le C\Delta t^4\|\nabla u_{tt}\|_{2,0}^2. \tag{A.25}$$

$$\begin{aligned}&\Delta t\sum_{n=1}^{N-1}\|\nabla(I[u(t^{n+1})]-u(t^{n+1}))\|^2(\|\nabla I[u(t^{n+1})]\|^2+\|\nabla u(t^{n+1})\|^2)\\ &\le C\Delta t\sum_{n=1}^{N-1}(\|\nabla I[u(t^{n+1})]\|^2+\|\nabla u(t^{n+1})\|^2)\Delta t^3\int_{t^{n-1}}^{t^{n+1}}\|\nabla u_{tt}\|^2dt\\ &\le C\Delta t^4\|\nabla u\|_{\infty,0}\sum_{n=1}^{N-1}\|\nabla u_{tt}\|^2dt = C\Delta t^4|||u_{tt}|||_{2,1}^2.\end{aligned} \tag{A.26}$$

Combining $(A.19)$ - $(A.26)$ gives

$$\begin{aligned}
&\|\phi_h^N\|^2 + \|2\phi_h^N - \phi_h^{N-1}\|^2 + \|\phi_h^N - \phi_h^{N-1}\|^2 + \sum_{n=1}^{N-1} 3\|\phi_h^{n+1} - 2\phi_h^n + \phi_h^{n-1}\|^2 \\
&+ \nu\Delta t \sum_{n=1}^{N-1} \|\nabla I[\phi_h^{n+1}]\|^2 \\
&\leq C\Big(\|u(t_1) - u_h^1\|^2 + \|(u(t_0) - u_h^0)\|^2 + h^{2k+2}\||u|\|_{\infty,k+1}^2 \\
&+ h^{2k+2}\|u_t\|_{2,k+1}^2 + h^{2k}\||u|\|_{2,k+1}^2 + h^{4k}\||u|\|_{4,k+1}^4 + h^{2s+2}\||p|\|_{2,s+1}^2 \\
&+ \Delta t^4(\|u_{ttt}\|_{2,0}^2 + \|\nabla u_{tt}\|_{2,0}^2\Big).
\end{aligned} \tag{A.27}$$

We add both sides of $(A.27)$ with

$$\begin{aligned}
&\|\eta^N\|^2 + \|2\eta^N - \eta^{N-1}\|^2 + \|\eta^N - \eta^{N-1}\|^2 + \sum_{n=1}^{N-1} 3\|\eta^{n+1} - 2\eta^n + \eta^{n-1}\|^2 \\
&+ \nu\Delta t \sum_{n=1}^{N-1} \|\nabla(\frac{3}{2}\eta^{n+1} - \eta^n + \frac{1}{2}\eta^{n-1})\|^2.
\end{aligned} \tag{A.28}$$

and apply triangle inequality to get (2.20). □

A.2 Second Order Error Estimator

This section justifies the use of EST_2 as an error estimator for the second order approximation. A Taylor series calculation shows that the second order approximation $y_{(2)}^{n+1}$ in Algorithm 2 has the local truncation error (LTE) (for constant stepsize)

$$LTE = -\Delta t^3 \left(\frac{1}{3}y''' + \frac{1}{2}f_y y''\right) + \mathcal{O}(\Delta t^4).$$

Consider the addition of a second time filter,

$$\begin{aligned}
&\text{Step 1} \ : && \tfrac{y_{(1)}^{n+1} - y^n}{\Delta t} = f(t_{n+1}, y_{(1)}^{n+1}), \\
&\text{Step 2} \ : && y_{(2)}^{n+1} = y_{(1)}^{n+1} - \tfrac{1}{3}\left\{y_{(1)}^{n+1} - 2y^n + y^{n-1}\right\} \\
&\text{Step 3} \ : && y^{n+1} = y_{(2)}^{n+1} - \tfrac{2}{11}\left\{y_{(2)}^{n+1} - 3y^n + 3y^{n-1} - y^{n-2}\right\}
\end{aligned} \tag{A.29}$$

Another Taylor series calculation shows that the induced method has the LTE of

$$LTE = -\Delta t^3 \frac{1}{2} f_y y'' + \mathcal{O}(\Delta t^4),$$

Thus, y^{n+1} yields a more accurate (still second order) approximation, and

$$EST_2 = y_{(2)}^{n+1} - y^{n+1} = \frac{2}{11}\left\{y_{(2)}^{n+1} - 3y^n + 3y^{n-1} - y^{n-2}\right\}$$

gives an estimate for the error of y^{n+1}. This is extended to variable stepsize using Newton interpolation, and written with stepsize ratios in Algorithm 2.

This is a nonstandard approach since one would normally use a higher order approximation to estimate the error. However, this is simple since it requires no additional function evaluations or Jacobians, and does not require solving a system of equations. Interestingly, (A.29) remains energy stable, and could be useful as a standalone constant stepsize method.

Appendix B

MOOSE234 Supplementary Material

B.1 Alternate Implementation of MOOSE234

We now state formulas for the coefficients used in the algorithm with stepsize ratios $\tau_n = \frac{k_n}{k_{n-1}}$ rather than divided differences. We will fix $\bar{k}_n := k_n$. Then BDFp is written

$$\frac{1}{k_{n+3}}\left(\sum_{i=0}^{p} \alpha_i^{(p)} y^{n+i}\right) = f(t_{n+p}, y_{n+p})$$

The coefficients for this are well known [82]. For completeness, we include the coefficients for variable stepsize BDF3 and BDF4 in this section. We will also show how the time filter coefficients can be expressed in terms of the BDF coefficients.

BDF3
$$\frac{1}{k_{n+3}}\left(\alpha_4^{(3)} y_3^{n+4} + \sum_{i=1}^{3} \alpha_i^{(3)} y^{n+i}\right) = f(t_{n+4}, y_3^{n+4})$$

BDF3-Stab
$$y_2^{n+4} = y_3^{n+4} + C_4 y_3^{n+4} + \sum_{i=1}^{3} C_i y^{n+i}$$

FBDF4
$$y_4^{n+4} = y_3^{n+4} + D_4 y_3^{n+4} + \sum_{i=0}^{3} D_i y^{n+i}$$

Put

$$\begin{aligned}
\mathrm{Est}_2 &= \quad y_3^{n+4} - y_2^{n+4} \\
\mathrm{Est}_3 &= \quad y_4^{n+4} - y_3^{n+4} \\
\mathrm{Est}_4 &= \left(\alpha_4^{(4)} y_4^{n+4} + \sum_{i=1}^{3} \alpha_i^{(4)} y^{n+i} - k_{n+3} f(t_{n+4}, y_4^{n+4})\right) \Big/ \alpha_4^{(4)}
\end{aligned} \tag{B.1}$$

The coefficients for below BDF3, below, are padded with an extra zero to make the formulas for the filter coefficients more clear. These coefficients were generated using Mathematica.

BDF3 coefficients ($\alpha_i^{(3)}$)

$$
\begin{aligned}
\alpha_0^{(3)} &= 0 \\
\alpha_1^{(3)} &= -\frac{\tau_{n+2}^3 \tau_{n+3}^2 \left(1+\tau_{n+3}\right)}{\left(1+\tau_{n+2}\right)\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right)} \\
\alpha_2^{(3)} &= \tau_{n+2}\tau_{n+3}^2 + \frac{\tau_{n+3}^2}{1+\tau_{n+3}} \\
\alpha_3^{(3)} &= -1-\tau_{n+3} - \frac{\tau_{n+2}\tau_{n+3}\left(1+\tau_{n+3}\right)}{1+\tau_{n+2}} \\
\alpha_4^{(3)} &= 1+\frac{\tau_{n+3}}{1+\tau_{n+3}} + \frac{\tau_{n+2}\tau_{n+3}}{1+\tau_{n+2}\left(1+\tau_{n+3}\right)}
\end{aligned}
$$

BDF4 coefficients ($\alpha_i^{(4)}$)

$$
\begin{aligned}
\alpha_0^{(4)} =& \frac{\tau_{n+1}^4 \tau_{n+2}^3 \tau_{n+3}^2 \left(1+\tau_{n+3}\right)\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right)}{\left(1+\tau_{n+1}\right)\left(1+\tau_{n+1}\left(1+\tau_{n+2}\right)\right)\left(1+\tau_{n+1}\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right)\right)} \\
\alpha_1^{(4)} =& -\frac{\tau_{n+1}\tau_{n+2}^3\tau_{n+3}^2\left(1+\tau_{n+3}\right)}{1+\tau_{n+2}} - \frac{\tau_{n+2}^3\tau_{n+3}^2\left(1+\tau_{n+3}\right)}{\left(1+\tau_{n+2}\right)\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right)} \\
\alpha_2^{(4)} =& \tau_{n+2}\tau_{n+3}^2 + \frac{\tau_{n+3}^2}{1+\tau_{n+3}} + \frac{\tau_{n+1}\tau_{n+2}\tau_{n+3}^2\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right)}{1+\tau_{n+1}} \\
\alpha_3^{(4)} =& -1-\tau_{n+3} - \frac{\tau_{n+2}\tau_{n+3}\left(1+\tau_{n+3}\right)}{1+\tau_{n+2}} \\
& - \frac{\tau_{n+1}\tau_{n+2}\tau_{n+3}\left(1+\tau_{n+3}\right)\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right)}{\left(1+\tau_{n+2}\right)\left(1+\tau_{n+1}\left(1+\tau_{n+2}\right)\right)} \\
\alpha_4^{(4)} =& 1+\tau_{n+3}\left(\frac{1}{1+\tau_{n+3}} + \frac{\tau_{n+2}}{1+\tau_{n+2}\left(1+\tau_{n+3}\right)}\right) + \frac{\tau_{n+1}\tau_{n+2}\tau_{n+3}}{1+\tau_{n+1}\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right)}
\end{aligned}
$$

BDF3-Stab coefficients (C_i)

$$
\begin{aligned}
C_0 =& 0 \\
C_1 =& -\frac{\mu\tau_{n+2}^2\tau_{n+3}\left(1+\tau_{n+3}\right)}{1+\tau_{n+2}} \\
C_2 =& \mu\tau_{n+3}\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right) \\
C_3 =& -\frac{\mu\left(1+\tau_{n+3}\right)\left(1+\tau_{n+2}\left(1+\tau_{n+3}\right)\right)}{1+\tau_{n+2}} \\
C_4 =& \mu
\end{aligned}
$$

FDF4 filter coefficients (D_i)

$$\gamma_i = \frac{\alpha_i^{(3)} - \alpha_i^{(4)}}{\alpha_4^{(3)}}$$

$$D_4 = \frac{\gamma_4}{1-\gamma_4}$$
$$D_i = \gamma_i(1+D_4) \qquad \text{for } 0 \le i \le 3.$$

The time filter coefficients D_i for FBDF4 can be implemented with knowledge of the BDF3 and BDF4 coefficients alone. We now show the derivation of the formulas for D_i. The goal as usual is to eliminate the time filter step to yield an equivalent method. We start with the equation for the time filter omitting the super script 4,

$$y^{n+4} = y_3^{n+4} + D_4 y_3^{n+4} + \sum_{i=0}^{3} D_i y^{n+i}$$

Add $D_4 y_4^{n+4}$ to both sides,

$$y^{n+4} + D_4 y_4^{n+4} = y_3^{n+4} + D_4 y_3^{n+4} + \sum_{i=0}^{4} D_i y^{n+i}.$$

Solving for y_3^{n+4},

$$y_3^{n+4} = y^{n+4} - \frac{1}{1+D_4} \sum_{i=0}^{4} D_i y^{n+i}.$$

Substituting this into the BDF3 step yields the equivalent method, the *left hand* side of which is

$$\frac{1}{k_{n+3}} \left(\sum_{i=1}^{4} \alpha_i^{(3)} y^{n+i} - \frac{\alpha_4^{(3)}}{1+D_4} \sum_{i=0}^{4} D_i y^{n+i} \right). \tag{B.2}$$

Recall that the time filter was chosen so that the left hand side of the induced FBDF4 method is equal to the left hand side of BDF4. Thus, D_i must satisfy

$$\alpha_i^{(3)} - \frac{\alpha_4^{(3)}}{1+D_4} D_i = \alpha_i^{(4)}$$

for $0 \le i \le 4$. This completely determines D_i. Let

$$\gamma_i = \frac{\alpha_i^{(3)} - \alpha_i^{(4)}}{\alpha_4^{(3)}}.$$

Solving the $i = 4$ first for D_4, then substituting this into the other equations gives the coefficients as stated above.

Remark B.1.1. *These formulas are easily generalized to the other FBDF methods. Given the coefficients of BDFp and BDFp+1, similarly compact formulas for the filter coefficients can be derived following the same steps above.*

B.2 Code to Calculate BDF Coefficients

Algorithm 11 (BDF and filter coefficients)**.**

$\boldsymbol{d}^j = \boldsymbol{e}_{m+1-j} \in \mathbb{R}^{m+1}$ *for* $j \in \{0, 1, \cdots, m\}$.

FUNCTION *BACKDIFF*$(t_n, ..., t_{n+m})$

$(c_0^{(0)}, c_1^{(0)}, \cdots, c_m^{(0)}) = \boldsymbol{d}^0$

FOR q = 1:m

FOR j = 0:m-q

$\boldsymbol{d}^j = (\boldsymbol{d}^j - \boldsymbol{d}^{j+1})/(t_{n+m-j} - t_{n+m-q-j})$

END FOR

$(c_0^{(q)}, c_1^{(q)}, \cdots, c_m^{(q)}) = \boldsymbol{d}^0$

END FOR

RETURN $\{c_i^{(j)}\}_{i,j=0}^m$

//The function below calculates the coefficients for BDF*p****,***

// $\eta^{(p+1)}$***, and the coefficients of the divided differences*** $c_i^{(j)}$

FUNCTION *BDFANDFILTCOEFF*$(t_n, ..., t_{n+m}, p)$

$\{c_i^{(j)}\}_{i,j=0}^m$ = *BACKDIFF*$(t_n, ..., t_{n+m})$

$\eta^{(p+1)} = \left(\prod_{i=1}^{p}(t_{n+m} - t_{n+m-i})\right) / \left(\sum_{j=1}^{p+1}(t_{n+m} - t_{n+m-j})^{-1}\right)$

FOR k = m-p:m

$\bar{\alpha}_k = \sum_{j=1}^{p}\left[\prod_{i=1}^{j-1}(t_{n+m} - t_{n+m-i})\right] c_k^{(j)}$

END FOR

RETURN $\{\bar{\alpha}_k\}_{k=m-p}^m, \eta^{(p+1)},$ and $\{c_i^{(j)}\}_{i,j=0}^m$

B.3 Python Implementation

We also include an implementation of the above functions in Python. The "\" character is the continue line command.

```
import numpy as np


def first_difference(T,Y):
        """
        T is a vector of times, greatest to least
        Y is a vector older differences, [d(j),d(j+1)]
        See algorthm in chapter
        """
        return (Y[0]-Y[1])/(T[0] - T[1])


def backward_differences(T):
        """
        Generate the divided difference coefficients.
        T is a vector of times from least to greatest, e.g.
        T = [t_n, t_n+1, ..., t_n+m]
        """
        numOfTimes = len(T)
        #the number of steps in the method
        m = numOfTimes - 1
        #generate the initial differences, which
        #is just the standard basis.
        D = np.array([ [np.float64((i+1)==(numOfTimes-j))\
          for i in xrange(numOfTimes)] \
            for j in xrange(numOfTimes)])
        differences = np.zeros_like(D)
```

```
        differences[0] = D[0]

        for q in xrange(1,numOfTimes):
         for j in xrange(numOfTimes - q):
                D[j] = first_difference\
                        ([T[m-j],T[m-j-q]],[D[j],D[j+1]])
                differences[q] = D[0]
        return differences

def bdf_coefficients_and_differences(T,order):
        differences = backward_differences(T)
        m = len(T)-1
        #calculate filter coefficient for increasing order
        eta = np.prod([T[m]-T[m-i] \
         for i in xrange(1,m)])/np.sum(1./(T[m] - T[m-j]) \
          for j in xrange(1,m+1))

        return [np.sum(np.prod([T[m]-T[m-i] \
         for i in xrange(1,j)])*differences[j] \
          for j in xrange(1,order+1)), differences,eta]
```

B.3.1 Coefficients of G matrix

Define the G-matrix as

$$\begin{pmatrix} g33 & g32 & g31 \\ g32 & g22 & g21 \\ g31 & g21 & g11 \end{pmatrix},$$

$$g33 = (\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)) + (-42\mu^2 + \mu + 21)/(22(\mu^2 + 2\mu + 1)); \tag{B.3}$$

$$g32 = (42\mu^2 + 13\mu - 7)/(11(\mu^2 + 2\mu + 1)) - (\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1) /(22(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)); \tag{B.4}$$

$$g31 = (\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)) - (21\mu^2 + 8\mu - 2)/(11(\mu^2 + 2\mu + 1)); \tag{B.5}$$

$$g22 = (\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(11(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)) - (120 \mu^2 + 23\mu - 9)/(22(\mu^2 + 2\mu + 1)); \tag{B.6}$$

$$g21 = (51\mu^2 + 5\mu - 2)/(22(\mu^2 + 2\mu + 1)) - (\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1) /(22(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)); \tag{B.7}$$

$$g11 = (-9\mu^2 + 2\mu)/(11(\mu^2 + 2\mu + 1)) + (\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1) /(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)); \tag{B.8}$$

$$a3 = (22 * \mu^2((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} \tag{B.9}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{3}{2}}$$
$$-42\mu^2((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}}$$
$$+\mu((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}}$$
$$+44\mu((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{3}{2}}$$
$$-((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}}$$
$$+22((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu$$
$$+1)))^{\frac{3}{2}}/(20\mu - 2)$$

$$a2 = -(22\mu^2((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} \tag{B.10}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{3}{2}}$$
$$-42\mu^2((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}}$$
$$+11\mu((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}}$$
$$+44\mu((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{3}{2}}$$
$$-2((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}}$$
$$+22((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}}$$
$$+41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu$$
$$+1)))^{\frac{3}{2}}/(10\mu - 1)$$

$$
\begin{aligned}
a1 = {} & (22\mu^2((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} \\
& + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{3}{2}} \\
& - 42\mu^2((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} \\
& + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}} \\
& + 41\mu((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} \\
& + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}} \\
& + 44\mu((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} \\
& + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{3}{2}} \\
& - 5((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} \\
& + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}} \\
& + 22((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} \\
& + 41\mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu \\
& + 1)))^{\frac{3}{2}}/(20\mu - 2);
\end{aligned}
\tag{B.11}
$$

$$
\begin{aligned}
a0 = {} & -((\mu + 3^{\frac{1}{2}}((7\mu-1)(6\mu-5)(14\mu-1)(\mu+1)^5)^{\frac{1}{2}} + 41 \\
& \mu^2 + 83\mu^3 + 42\mu^4 + 1)/(44(\mu^4 + 4\mu^3 + 6\mu^2 + 4\mu + 1)))^{\frac{1}{2}};
\end{aligned}
\tag{B.12}
$$

Bibliography

[1] Saghir Ahmad. *Design and construction of software for general linear methods*. PhD thesis, Massey University, 2016.

[2] M. Akbas, S. Kaya, and L. G. Rebholz. On the stability at all times of linearly extrapolated BDF2 timestepping for multiphysics incompressible flow problems. *Numerical Methods for Partial Differential Equations*, 33(4):999–1017.

[3] Georgios Akrivis and Christian Lubich. Fully implicit, linearly implicit and implicit—explicit backward difference formulae for quasi-linear parabolic equations. *Numer. Math.*, 131(4):713–735, December 2015.

[4] M. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. Rognes, and G. Wells. The FEniCS project version 1.5. *Archive of Numerical Software*, 3(100), 2015.

[5] Ariyadasa Aluthge, Scott Sarra, and Roger Estep. Filtered leapfrog time integration with enhanced stability properties. *Journal of Applied Mathematics and Physics*, 04:1354–1370, 01 2016.

[6] D. N. Arnold, F. Brezzi, and M. Fortin. A stable finite element for the Stokes equations. *CALCOLO*, 21(4):337–344, Dec 1984.

[7] R.A. Asselin. Frequency filter for time integrations. *Mon. Weather Rev*, (100):487–490, 1972.

[8] Richard Asselin. Frequency filter for time integrations. *Monthly Weather Review*, 100(6):487–490, 1972.

[9] G.A. Baker. Galerkin approximations for the Navier-Stokes equations. Technical report, Harvard University, 1976.

[10] Garth A. Baker, Vassilios A. Dougalis, and Ohannes A. Karakashian. On a higher order accurate fully discrete Galerkin approximation to the Navier-Stokes equations. *Mathematics of Computation*, 39(160):339–375, 1982.

[11] J. Becker. A second order backward difference method with variable steps for a parabolic problem. *BIT Numerical Mathematics*, 38(4):644–662, Dec 1998.

[12] Michael Besier and Rolf Rannacher. Goal-oriented spacetime adaptivity in the finite element Galerkin method for the computation of nonstationary incompressible flow. *International Journal for Numerical Methods in Fluids*, 70(9):1139–1166, January 2012.

[13] M. Calvo, T. Grande, and R. D. Grigorieff. On the zero stability of the variable order variable stepsize BDF-Formulas. *Numerische Mathematik*, 57(1):39–50, Dec 1990.

[14] Sergey Charnyi, Timo Heister, Maxim A. Olshanskii, and Leo G. Rebholz. On conservation laws of Navier-Stokes Galerkin discretizations. *Journal of Computational Physics*, 337:289 – 308, 2017.

[15] A. J. Chorin. The numerical solution of the Navier-Stokes equations for an incompressible fluid. *Bull. Amer. Math. Soc.*, 73(6):928–931, 11 1967.

[16] Alexandre Joel Chorin. Numerical solution of the Navier-Stokes equations. *Mathematics of Computation*, 22(104):745–762, 1968.

[17] E.M. Constantinescu and A. Sandu. Extrapolated implicit-explicit time stepping. *SIAM Journal of Scientific Computing*, 31(6):4452–4477, 2010.

[18] Emil M. Constantinescu. Generalizing global error estimation for ordinary differential equations by using coupled time-stepping methods. *J. Comput. Appl. Math.*, 332:140–158, 2018.

[19] S.D. Conte and Carl de Boor. *Elementary Numerical Analysis*. McGraw-Hill, 3rd edition, 1980.

[20] M. Crouzeix and F. J. Lisbona. The convergence of variable-stepsize, variable-formula, multistep methods. *SIAM Journal on Numerical Analysis*, 21(3):512–534, 1984.

[21] M. Crouzeix and P. A. Raviart. Approximation d'équations d'évolution linéaires par des méthodes multipas. *Etude Numérique des Grands Systèmes*, 1976.

[22] Germund Dahlquist. A special stability problem for linear multistep methods. *BIT Numerical Mathematics*, 3(1):27–43, Mar 1963.

[23] Germund Dahlquist. G-stability is equivalent to A-stability. *BIT Numerical Mathematics*, 18(4):384–401, Dec 1978.

[24] Germund Dahlquist. Some properties of linear multistep and one-leg methods for ordinary differential equations. Technical report, Royal Institute of Technology, Stockholm, 1979.

[25] Germund Dahlquist. On the local and global errors of one-leg methods. Technical Report TRITA-NA-8110, Royal Institute of Technology, Stockholm, 1981.

[26] Germund Dahlquist. On one-leg multistep methods. *SIAM Journal on Numerical Analysis*, 20(6):1130–1138, 1983.

[27] Germund G. Dahlquist, Werner Liniger, and Olavi Nevanlinna. Stability of two-step methods for variable integration steps. *SIAM Journal on Numerical Analysis*, 20(5):1071–1085, 1983.

[28] Victor DeCaria, William Layton, and Michael McLaughlin. A conservative, second order, unconditionally stable artificial compression method. *Computer Methods in Applied Mechanics and Engineering*, 325(Supplement C):733 – 747, 2017.

[29] Jim Douglas and Todd Dupont. Galerkin methods for parabolic equations. *SIAM Journal on Numerical Analysis*, 7(4):575–626, 1970.

[30] Etienne Emmrich. Error of the two-step BDF for the incompressible Navier-Stokes problem. *ESAIM: M2AN*, 38(5):757–764, 2004.

[31] Etienne Emmrich. Stability and convergence of the two-step BDF for the incompressible Navier-Stokes Problem. 5:199–209, 01 2004.

[32] Etienne Emmrich. Stability and error of the variable two-step bdf for semilinear parabolic problems. *Journal of Applied Mathematics and Computing*, 19(1):33–55, Mar 2005.

[33] Etienne Emmrich. Convergence of the variable two-step bdf time discretisation of nonlinear evolution problems governed by a monotone potential operator. *BIT Numerical Mathematics*, 49(2):297–323, Jun 2009.

[34] J. A. Fiordilino. On pressure estimates for the Navier-Stokes equations. *ArXiv e-prints*, March 2018.

[35] Joseph Fiordilino. A second order ensemble timestepping algorithm for natural convection. 56:816837, 08 2017.

[36] K. J. Galvin. New subgrid artificial viscosity Galerkin methods for the Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 200(1):242 – 250, 2011.

[37] T. Geveci. On the convergence of a time discretization scheme for the Navier-Stokes equations. *Mathematics of Computation*, 53(187):43–53, 1989.

[38] Vivette Girault and Pierre-Arnaud Raviart. *Finite Element Approximation of the Navier-Stokes Equations*. Springer-Verlag Berlin Heidelberg, 1979.

[39] Vivette Girault and Pierre-Arnaud Raviart. An optimally accurate discrete regularization for second order timestepping methods for navier–stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 310:388–405, 2016.

[40] P. M. Gresho and R. L. Sani. *Incompressible Flow and the Finite Element Method*. John Wiley & Sons, Inc., 1998.

[41] David F. Griffiths and Desmond J. Higham. *Numerical Methods for Ordinary Differential Equations*. Springer-Verlag London Limited, 2010.

[42] D.F. Griffiths and D.J. Higham. Numerical methods for ordinary differential equations. *Springer*, 2010.

[43] Jean-Luc Guermond and Peter Minev. High-order time stepping for the incompressible navier–stokes equations. 37:A2656–A2681, 01 2015.

[44] N. Guglielmi and M. Zennaro. On the zero-stability of variable stepsize multistep methods: the spectral radius approach. *Numerische Mathematik*, 88(3):445–458, May 2001.

[45] Max D Gunzburger. *Finite Element Methods for Viscous Incompressible Flows: A guide to theory, practice, and algorithms*. Elsevier, 2012.

[46] Ahmet Guzel and William Layton. Time filters increase accuracy of the fully implicit method. *BIT Numerical Mathematics*, 58(2):301–315, Jun 2018.

[47] A. Hay, S. Etienne, D. Pelletier, and A. Garon. hp-adaptive time integration based on the bdf for viscous flows. *Journal of Computational Physics*, 291(Supplement C):151 – 176, 2015.

[48] Y. He. The euler implicit/explicit scheme for the 2d time-dependent navier-stokes equations with smooth or non-smooth initial data. *Mathematics of Computation*, 77(264):2097–2124, 2008.

[49] J. Heywood and R. Rannacher. Finite element approximation of the nonstationary Navier-Stokes problem. part iv. error analysis for second-order time discretization. *SIAM Journal on Numerical Analysis*, 27(2):353384, 1990.

[50] Adrian Hill and Terence J. T. Norton. Filters for time-stepping methods. Technical report, 09 2017. International Conference on Scientific Computation and Differential Equations at Bath, UK.

[51] Ross Ingram. Unconditional convergence of high-order extrapolations of the crank-nicolson, finite element method for the navier-stokes equations. 10, 01 2013.

[52] E. Isaacson and H.B. Keller. *Analysis of Numerical Methods*. John Wiley and Sons, 1966.

[53] M Janssen and Pascal Van Hentenryck. Precisely A(α)-stable one-leg multistep methods. *BIT Numerical Mathematics*, 43:761–774, 11 2003.

[54] Nan Jiang. A second-order ensemble method based on a blended backward differentiation formula timestepping scheme for time-dependent Navier-Stokes equations. *Numerical Methods for Partial Differential Equations*, 33(1):34–61.

[55] Nan Jiang, Muhammad Mohebujjaman, Leo G. Rebholz, and Catalin Trenchea. An optimally accurate discrete regularization for second order timestepping methods for Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 310:388 – 405, 2016.

[56] V. John. Reference values for drag and lift of a two-dimensional time-dependent flow around a cylinder. *International Journal for Numerical Methods in Fluids*, 44(7):777–788, 2004.

[57] V. John and J. Rang. Adaptive time step control for the incompressible navier–stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 199(9):514 – 524, 2010.

[58] H. Johnston and J. Liu. Accurate, stable and efficient navierstokes solvers based on explicit treatment of the pressure term. *Journal of Computational Physics*, 199(1):221 – 259, 2004.

[59] David Kay, Philip M. Gresho, David Griffiths, and David Silvester. Adaptive time-stepping for incompressible flow part ii: Navier-Stokes equations. 32:111128, 01 2010.

[60] B. Klein, F. Kummer, M. Keil, and M. Oberlack. An extension of the SIMPLE based discontinuous Galerkin solver to unsteady incompressible flows. *International Journal for Numerical Methods in Fluids*, 77(10):571–589.

[61] Gennady Yu. Kulikov and Sergey K. Shindin. One-leg variable-coefficient formulas for ordinary differential equations and local–global step size control. *Numerical Algorithms*, 43(1):99–121, Sep 2006.

[62] W. Layton and C. Trenchea. Stability of two IMEX methods, CNLF and BDF2-AB2, for uncoupling systems of evolution equations. *Applied Numerical Mathematics*, 62:112–120, 2012.

[63] William Layton. *Introduction to the numerical analysis of incompressible viscous flows*, volume 6. Siam, 2008.

[64] William Layton, Nathaniel Mays, Monika Neda, and Catalin Trenchea. Numerical analysis of modular regularization methods for the BDF2 time discretization of the Navier-Stokes equations. *ESAIM: M2AN*, 48(3):765–793, 2014.

[65] William Layton, Aziz Takhirov, and Myron Sussman. Instability of crank-nicolson leap-frog for nonautonomous systems.

[66] Y. Li and C. Trenchea. A higher-order Robert-Asselin type time filter. *J. Comput. Phys.*, 259:23–32, 2014.

[67] Jian-Guo Liu, Jie Liu, and Robert L. Pego. Stable and accurate pressure approximation for unsteady incompressible viscous flow. *Journal of Computational Physics*, 229(9):3428 – 3453, 2010.

[68] Martine Marion and Roger Temam. Navier-stokes equations: Theory and approximation. In *Numerical Methods for Solids (Part 3) Numerical Methods for Fluids (Part 1)*, volume 6 of *Handbook of Numerical Analysis*, pages 503 – 689. Elsevier, 1998.

[69] Olavi Nevanlinna and Werner Liniger. Contractive methods for stiff differential equations part i. *BIT Numerical Mathematics*, 18(4):457–474, Dec 1978.

[70] Alessandra Nigro, Carmine De Bartolo, Francesco Bassi, and Antonio Ghidoni. Up to sixth-order accurate A-stable implicit schemes applied to the Discontinuous Galerkin discretized Navier-Stokes equations. *Journal of Computational Physics*, 276:136 – 162, 2014.

[71] J. Rang and L. Angermann. New Rosenbrock W-Methods of order 3 for partial differential algebraic equations of index 1. *BIT Numerical Mathematics*, 45(4):761–787, Dec 2005.

[72] S. S. Ravindran. An analysis of the blended three-step backward differentiation formula time-stepping scheme for the Navier-Stokes-type system related to Soret convection. *Numerical Functional Analysis and Optimization*, 36(5):658–686, 2015.

[73] S. S. Ravindran. An extrapolated second order backward difference time-stepping scheme for the magnetohydrodynamics system. *Numerical Functional Analysis and Optimization*, 37(8):990–1020, 2016.

[74] Andre J. Robert. *An evaluation of the behaviour of planetary waves in an atmospheric model based on spherical harmonics*. PhD thesis, McGill, 1965.

[75] M Schäfer and S. Turek. Benchmark computations of laminar flow around a cylinder. In Hirschel E.H., editor, *Flow Simulation with High-Performance Computers II. Notes on Numerical Fluid Mechanics*, volume 48, pages 547–566. Vieweg+Teubner Verlag, 1996.

[76] L. F. Shampine. Error estimation and control for odes. *Journal of Scientific Computing*, 25(1):3–16, Oct 2005.

[77] Gustaf Söderlind, Imre Fekete, and István Faragó. On the zero-stability of multistep methods on smooth nonuniform grids. *BIT*, 58(4):1125–1143, 2018.

[78] R. Temam. *Navier-Stokes Equations and Nonlinear Functional Analysis*. Society for Industrial and Applied Mathematics, 1995.

[79] Veer N. Vatsa, Mark H. Carpenter, and David P. Lockard. Re-evaluation of an optimized second order backward difference (BDF2OPT) scheme for unsteady flow applications. Technical Report AIAA Paper 2010-0122, National Aeronautics and Space Administration, 2010.

[80] R. Verfürth. Error estimates for a mixed finite element approximation of the Stokes equations. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 18(2):175–182, 1984.

[81] Dong Wang and Steven J. Ruuth. Variable step-size implicit-explicit linear multistep methods for time-dependent partial differential equations. *Journal of Computational Mathematics*, 26(6):838–855, 2008.

[82] Xiaoming Wang. An efficient second order in time scheme for approximating long time statistical properties of the two dimensional Navier-Stokes equations. 121:753–779, 08 2011.

[83] Paul D. Williams. A proposed modification to the Robert-Asselin time filter. *Monthly Weather Review*, 137:2538–2546, 08 2009.

[84] Paul D. Williams. The RAW filter: An improvement to the Robert-Asselin filter in semi-implicit integrations. *Monthly Weather Review*, 139(6):1996–2007, 2011.

[85] Paul D. Williams. Achieving seventh-order amplitude accuracy in leapfrog integrations. *Monthly Weather Review*, 141(9):3037–3051, 2013.