INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 17, Number 2, Pages 254–280

A TIME-ACCURATE, ADAPTIVE DISCRETIZATION FOR FLUID FLOW PROBLEMS

VICTOR DECARIA, WILLIAM LAYTON, AND HAIYUN ZHAO

Abstract. This report presents a low computational and cognitive complexity, stable, time accurate and adaptive method for the Navier-Stokes equations. The improved method requires a minimally intrusive modification to an existing program based on the fully implicit / backward Euler time discretization, does not add to the computational complexity, and is conceptually simple. The backward Euler approximation is simply post-processed with a two-step, linear time filter. The time filter additionally removes the overdamping of Backward Euler while remaining unconditionally energy stable, proven herein. Even for constant stepsizes, the method does not reduce to a standard / named time stepping method but is related to a known 2-parameter family of A-stable, two step, second order methods. Numerical tests confirm the predicted convergence rates and the improved predictions of flow quantities such as drag and lift.

Key words. Navier-Stokes, backward Euler, time filter, time discretization, finite element method.

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, [17], and can fail by overdamping a solution's dynamic behavior. For ODEs, adding a time filter to backward Euler, as in (1.3) below, yields two, embedded, A-stable approximations of first and second order accuracy, [20]. This report 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 1.1 below involves adding only 1 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 and tested in Section 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 (VSVO) method is presented in Section 2 and tested in Section 6.2.

Even for constant time-steps and constant order, the method herein does not reduce to a standard / named method. Algorithm 1.1 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 3.2, Section 3. Stability and velocity convergence of the (constant time step) general second

Received by the editors January 24, 2019, and accepted September 2, 2019.
2000 *Mathematics Subject Classification.* 65M99, 76M10.

order, two-step, A-stable method for the Navier-Stokes equations was proven already in [15], see equation (3.20) p. 185, and has been elaborated thereafter, e.g., [23]. Our *velocit*y stability and error analysis, while necessary for completeness, parallels this previous work and is thus collected in Appendix A. On the other hand, Algorithm 1.1 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:

$$\begin{aligned} &u_t + u\cdot\nabla u - \nu\Delta u + \nabla p = f, \text{ and } \nabla\cdot u = 0 \text{ in } \Omega,\\ &u = 0 \text{ on } \partial\Omega, \text{ and } \int_\Omega p\,dx = 0, \\ &u(x,0) = u_0(x) \text{ in } \Omega. \end{aligned} \tag{1}$$

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 1.1 (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)*

$$\begin{aligned} &\frac{\hat{u}^{n+1}-u^n}{\Delta t} + u^*\cdot\nabla\hat{u}^{n+1} - \nu\Delta\hat{u}^{n+1} + \nabla\hat{p}^{n+1} = f(t^{n+1}),\\ &\nabla\cdot\hat{u}^{n+1} = 0, \end{aligned} \tag{2}$$

Step 2: (Time Filter for velocity and pressure)

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

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 1.1A means Option A is used, and Algorithm 1.1B means Option B is used.

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 and error analysis for constant timestep in Appendix A. 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 [15] and [23]. We also give an analysis of the unfiltered pressure error, which does not have a parallel in [15] or [23]. The