Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 335 (2018) 327–346
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

An efficient and modular grad–div stabilization

J.A. Fiordilino[a,*], W. Layton[a], Y. Rong[a,b]

[a] University of Pittsburgh, Department of Mathematics, Pittsburgh, PA, 15260, USA
[b] Xi'an Jiaotong University, School of Mathematics and Statistics, Xi'an, Shaanxi, 710049, China

Received 1 December 2017; received in revised form 23 February 2018; accepted 24 February 2018
Available online 6 March 2018

Abstract

This paper presents two modular grad–div algorithms for calculating solutions to the Navier–Stokes equations (NSE). These algorithms add to an NSE code a minimally intrusive module that implements grad–div stabilization. The algorithms do not suffer from either solver breakdown or debilitating slow down for large values of grad–div parameters. Stability and optimal-order convergence of the methods are proven. Numerical tests confirm the theory and illustrate the benefits of these algorithms over a fully coupled grad–div stabilization.

Keywords: Navier–Stokes equations; Grad–div stabilization; Modular methods

1. Introduction

Grad–div stabilization of fluid flow problems has drawn attention due to its positive impact on solution quality. However, it also introduces new computational challenges. As the grad–div parameter γ increases, the condition number of the resulting linear system generally grows without bound [1]; consequently, iterative solvers can slow dramatically, Fig. 1. Unfortunately, appropriate values for γ can vary wildly depending on the application; proposed values include: $\mathcal{O}(h)$ [2], $\mathcal{O}(\nu)$ [3], and both local and global solution ratios $\mathcal{O}(\frac{|p|_m}{|v|_{k+1}})$ [4], among others [5,6]; values as high as $\gamma = 1{,}000$ and 10,000 produce good results for Rayleigh–Bénard convection for silicon oil [2]. Therefore, moderate or even large values of γ may be unavoidable. Moreover, grad–div stabilization increases coupling, decreases sparsity, and makes preconditioning more difficult. Research has addressed the former [7–11] and the latter [12–19], but full resolution is still open.

This paper presents two modular grad–div stabilizations resolving both issues, Algorithms 1 and 2 in Section 3. Algorithm 2 incorporates sparse grad–div ideas from [9] resulting in further storage reduction and efficiency gains. Each method adds a minimally intrusive stabilization module. The algorithms are simple to implement, retain the benefits of grad–div stabilization, and are resilient to solver breakdown as stabilization parameters increase, Fig. 1.

* Corresponding author.
E-mail addresses: jaf150@pitt.edu (J.A. Fiordilino), wjl@pitt.edu (W. Layton), rongyao@stu.xjtu.edu.cn (Y. Rong).

https://doi.org/10.1016/j.cma.2018.02.023
0045-7825/

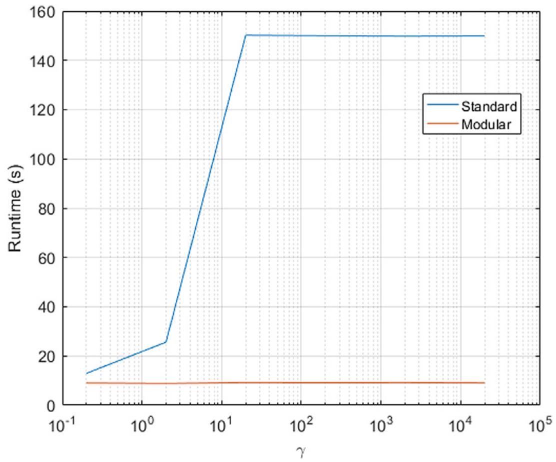

Fig. 1. Runtimes: Standard vs. modular grad-div for increasing γ.

Both the modularity and simplicity of these algorithms allow them to be easily introduced into legacy codes. They are, however, still endangered by the classical Poisson locking phenomenon. Moreover, we note that higher order, exactly divergence free elements, e.g., [6,20,21], offer an important alternative to grad–div stabilization not considered herein.

Let $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$ be a convex polyhedral domain with piecewise smooth boundary $\partial\Omega$. Given the fluid viscosity ν, $u(x, 0) = u^0(x)$, and the body force $f(x, t)$, the velocity $u(x, t) : \Omega \times (0, t^*] \to \mathbb{R}^d$ and pressure $p(x, t) : \Omega \times (0, t^*] \to \mathbb{R}$ satisfy

$$\begin{aligned} u_t + u \cdot \nabla u - \nu \Delta u + \nabla p = f \;\; in \; \Omega, \\ \nabla \cdot u = 0 \;\; in \; \Omega, \\ u = 0 \;\; on \; \partial\Omega. \end{aligned} \tag{1}$$

To explain how grad–div stabilization terms $-\beta\nabla\nabla \cdot u_t$ and $-\gamma\nabla\nabla \cdot u$ $(\beta \geq 0,\ \gamma \geq 0)$ are introduced, suppress the spatial discretization momentarily and consider the following simple example:

Algorithm 1. Step 1: Given u^n, find $\hat{u}^{n+1}$ and p^{n+1} satisfying

$$\frac{\hat{u}^{n+1} - u^n}{\Delta t} + u^n \cdot \nabla \hat{u}^{n+1} - \nu \Delta \hat{u}^{n+1} + \nabla p^{n+1} = f^{n+1} \; and \; \nabla \cdot \hat{u}^{n+1} = 0. \tag{2}$$

Step 2: Given $\hat{u}^{n+1}$, find u^{n+1} satisfying

$$u^{n+1} - (\beta + \gamma \Delta t)\nabla\nabla \cdot u^{n+1} = \hat{u}^{n+1} - \beta\nabla\nabla \cdot u^n. \tag{3}$$

Step 1 is obviously a consistent discretization of the NSE. Step 2 can be rewritten as

$$\frac{\hat{u}^{n+1} - u^{n+1}}{\Delta t} = -\beta\nabla\nabla \cdot \Big(\frac{u^{n+1} - u^n}{\Delta t}\Big) - \gamma\nabla\nabla \cdot u^{n+1}. \tag{4}$$

Rewriting the first term in Step 1 as $\frac{u^{n+1} - u^n}{\Delta t} + \frac{\hat{u}^{n+1} - u^{n+1}}{\Delta t}$ and using (4), we see that Steps 1 and 2 introduce the bold terms below:

$$\frac{u^{n+1} - u^n}{\Delta t} - \boldsymbol{\beta\nabla\nabla\cdot}\Big(\frac{\boldsymbol{u^{n+1} - u^n}}{\boldsymbol{\Delta} t}\Big) + u^n \cdot \nabla \hat{u}^{n+1} - \nu \Delta \hat{u}^{n+1} - \boldsymbol{\gamma\nabla\nabla\cdot u^{n+1}} = f^{n+1}.$$

After spatial discretization, Step 1 requires solution of a standard and well understood velocity–pressure system without the added coupling or ill conditioning of the grad–div terms while Step 2 is the same uncoupled, SPD, grad–div system at every timestep. Fig. 1, summarizing a timing test in Section 5, shows this separation into two