Journal of Computational Physics 373 (2018) 940–974

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Iterated pressure-correction projection methods for the unsteady incompressible Navier–Stokes equations

Jean Aoussou [1], Jing Lin [1], Pierre F.J. Lermusiaux [*]

Department of Mechanical Engineering, Massachusetts Institute of Technology 77 Massachusetts Avenue, Cambridge, MA 02139, USA

A R T I C L E   I N F O

Article history:
Received 23 December 2017
Accepted 24 June 2018
Available online 3 July 2018

Keywords:
Incompressible Navier–Stokes
Projection methods
Pressure-correction schemes
Preconditioned iterative methods
High-order time-marching
IMEX schemes

A B S T R A C T

Iterated pressure-correction projection schemes for the unsteady incompressible Navier–Stokes equations are developed, analyzed and exemplified, in relation to preconditioned iterative methods and the pressure-Schur complement equation. Typical pressure-correction schemes perform only one iteration per stage or time step, and suffer from splitting errors that result in spurious numerical boundary layers and a limited order of convergence in time. We show that performing iterations not only reduces the effects of the splitting errors, but can also be more efficient computationally than merely reducing the time step. We devise stopping criteria to recover the desired order of temporal convergence, and to drive the splitting error below the time-integration error. We also develop and implement the iterated pressure corrections with both multi-step and multi-stage time integration schemes. Finally, to reduce further the computational cost of the iterated approach, we combine it with an Aitken acceleration scheme. Our theoretical results are validated and illustrated by numerical test cases for the Stokes and Navier–Stokes equations, using implicit–explicit (IMEX) backwards differences and Runge–Kutta time-integration solvers. The test cases comprise a now classical manufactured solution in the projection method community and a modified version of a more recently proposed manufactured solution. The different error types, stopping criterion, recovered orders of convergence, and acceleration rates are illustrated, as well as the effects of the rotational corrections and time-integration schemes. It is found that iterated pressure-correction schemes can retrieve the accuracy and temporal convergence order of fully-coupled schemes and are computationally more efficient than classic pressure-correction schemes.

1. Introduction

Solving the unsteady fully coupled incompressible Navier–Stokes equations is computationally demanding [40,59,26,62]. The projection method introduced by [14] and [56], which can decouple the momentum and continuity equations numerically by replacing the latter with a Poisson equation for the pressure, is often a preferred option. A non-divergence-free velocity field is first obtained by omitting the pressure term in the momentum equation, and is then corrected using the pressure Poisson equation (PPE). This method was reinterpreted as a fractional-step scheme by [32] and a number of vari-

[*] Corresponding author.
E-mail address: pierrel@mit.edu (P.F.J. Lermusiaux).
[1] JA and JL are co-first authors.

https://doi.org/10.1016/j.jcp.2018.06.062
0021-9991/© 2018 Elsevier Inc. All rights reserved.

ants of the original scheme have been developed, a review of which can be found in [24]. In this study, only the incremental pressure correction schemes are considered. This time splitting decouples the computation of velocity and pressure in the sense that instead of solving a large linear system for all velocity components and pressure simultaneously, it only solves $(d+1)$ smaller linear systems for each velocity component and pressure (or pressure correction) separately [62], where d is the spatial dimension of the flow. Because the cost of solving a linear system typically scales worse than linear, such breakdown is beneficial. Moreover, the decoupled linear systems all have the same structure as a discrete Laplacian, which is symmetric positive definite (SPD) and can be solved efficiently with many mature techniques. In contrast, the coupled system yields a saddle-point problem (symmetric indefinite) and is worse-conditioned because it involves both the momentum equations and the continuity equation, which behave very differently. However, this gain for time splitting comes at the price of a splitting error which manifests itself as an artificial boundary layer in space and as a limited order of temporal convergence [36,51,25,24].

Incremental pressure-correction methods use the pressure at the previous time step as a guess for the current time. This guess is then updated, and the scheme moves on to the next time step. In order to obtain a more accurate solution, instead of directly moving on to the next time step, it is possible to repeat the process and use the newly computed pressure as the guess. Indeed, doing so reduces the splitting error, and if the process is repeated enough times, it renders the splitting error negligible compared to the error due to the time-integration scheme. In fact, other methods, such as the Uzawa algorithm [40], the SIMPLE method and their variants [21], which as projection schemes decouple the system and replace the incompressibility condition with a pressure Poisson equation, perform similar inner loop iterations in addition to the time-integration outer loop. They are particularly efficient for stationary problems with pseudo time integration [21]. However, for unsteady problems, since the splitting error scales with the time step, it is not always clear whether performing those pressure correction iterations offers a real benefit as opposed to using smaller time steps in order to achieve the same accuracy. One of our objectives is to investigate the convergence properties of such iterated pressure-correction schemes and assess the benefit of performing more iterations as opposed to reducing the time step.

The pressure-correction method can also be viewed as one iteration towards solving the pressure-Schur complement equation. This is in fact also true for the Uzawa and SIMPLE schemes [40,41,46]. More precisely, those methods can be understood as preconditioned Richardson iterations and only differ in the preconditioner being used. In the case of the projection method, the preconditioner is the Laplacian operator and with a regular scheme, only one iteration is performed [60]. If more iterations are performed, the numerical solution converges to the solution of the pressure-Schur equation. With this approach [3], it is possible to estimate how fast the splitting error decreases with each iteration and thus assess how performing more iterations compares to reducing the time step in terms of computational cost and accuracy. When doing so, care is needed to differentiate among the different error types, namely the spatial, splitting, and time-marching errors.

A wide range of time-integration schemes are available to solve the incompressible Navier–Stokes equations, for both fully-coupled and projection-type methods. A common choice is the backward difference formulae (BDF) schemes [11]. Those fully-implicit schemes usually enjoy good stability for relatively large time steps. However, they require root-finding for a nonlinear system because of the nonlinearity of the advection term [60,43]. Implicit–explicit (IMEX) schemes thus offer a good alternative and have indeed been commonly used with projection-like schemes [32,37,11,49].

Nevertheless, BDF schemes are not self-starting and have smaller stability regions at high orders [50]. In particular, methods of order higher than 3 are not A-stable and methods of order higher than 6 are unstable [28]. Although this is not necessarily an issue for all types of problems [18] and other multi-step methods with better stability properties do exist [27], multi-stage methods, such as Runge–Kutta schemes seem to offer a viable alternative. The study of their application to projection methods has however been limited so far. A number of studies address the use of fully-explicit schemes [48,30,37,19]. Second-order RK-IMEX schemes were studied in [45] and [44], and higher order schemes in [71] and [16] for PPE-based methods. A pressure-free projection method for cases with periodic boundary conditions was proposed in [70]. To our knowledge, [62] outlines the most general IMEX-RK schemes for pressure-correction schemes.

In what follows, we first summarize the usual time discretization schemes for the Navier–Stokes equations and common strategies to deal with the nonlinear advection term (Sec. 2). We then outline common ways to handle the velocity–pressure coupling and discuss their inter-connections (Sec. 3). We develop and analyze the accuracy and convergence properties of the new iterated projection methods (Sec. 4). In particular, we show that in order to achieve a certain accuracy, it is often advantageous in terms of computational effort to perform more iterations than to reduce the time step. We obtain a time step size-dependent stopping criterion that allows the control of the temporal order of convergence of the remaining splitting error. We show how iterations can be accelerated using an Aitken relaxation scheme [39]. Next, in Sec. 5, we explain how the method can be practically embedded into existing pressure-correction schemes, for both IMEX linear multi-step and IMEX Runge–Kutta time-integration schemes. In Sec. 6, we present numerical results that illustrate and verify the properties inferred by applying the new methods to test cases of manufactured solutions for both the Stokes and Navier–Stokes equations. We illustrate and discuss the different error types, stopping criterion, recovered orders of convergence, suppression of the splitting errors, acceleration rates, and computational costs, as well as the effects of the rotational correction and time integration schemes. Finally, conclusions and future directions are provided (Sec. 7).

2. Discretization and handling nonlinearity

The incompressible Navier–Stokes equations can be written as:

$$\begin{cases} \dfrac{\partial \boldsymbol{u}}{\partial t} = \nu\nabla^2\boldsymbol{u} - \nabla p + \boldsymbol{f} - \boldsymbol{u}\cdot\nabla\boldsymbol{u} & \text{(a)} \\ \nabla\cdot\boldsymbol{u} = 0 & \text{(b)} \end{cases} \tag{1}$$

where p is pressure divided by the constant fluid density (p is called pressure hereafter for conciseness), ν the kinematic viscosity, and $\boldsymbol{f}$ a given forcing term that does not depend on $\boldsymbol{u}$. Note that p enters the equations only through its gradient, so p is determined only up to an additive constant, which is uniform in space but can vary arbitrarily in time. In practice, this degree of freedom can be eliminated by imposing an extra condition on p either out of a realistic physical setup, e.g. anchoring p at a reference point: $p|_{\boldsymbol{x}=\boldsymbol{x}_{\text{ref}}} = p_{\text{ref}}$, or for numerical convenience, e.g. requiring p to have a vanishing spatial mean: $\int_\Omega p\,d\boldsymbol{x} = 0$.

The system (1) is different from a normal dynamic PDE (Partial Differential Equation) system in that although p is an unknown field, there is no prognostic equation for p and the only extra equation (1b) other than (1a) for $\boldsymbol{u}$ does not involve p at all. Therefore, p is determined by being a field such that $\boldsymbol{u}$ governed by (1a) always satisfies the divergence-free constraint (1b).[2]

Due to the special role of p and (1b), to numerically evolve $\boldsymbol{u}$ and p from time t_n to t_{n+1}, we can only apply a time-marching scheme to (1a) and treat the pressure implicitly,[3] such that both $\boldsymbol{u}^{n+1}$ and p^{n+1} are unknowns in the time-discrete counterpart of (1a). The field p^{n+1} is then determined such that $\boldsymbol{u}^{n+1}$ computed by the time-marching scheme given this p^{n+1} also satisfies $\nabla\cdot\boldsymbol{u}^{n+1} = 0$. This mimics the role played by p in the time-continuous setting.

Terms other than the pressure gradient can be treated either explicitly or implicitly, so a general time-marching scheme can be formulated as an IMEX (IMplicit–EXplicit) scheme. For a time-dependent PDE, the right hand side is divided into two parts, one is treated implicitly and the other explicitly:

$$\frac{\partial\boldsymbol{u}}{\partial t} = \boldsymbol{F}^{\text{im}}(\boldsymbol{u},t) + \boldsymbol{F}^{\text{ex}}(\boldsymbol{u},t)\,. \tag{2}$$

If no term is treated explicitly, (2) reduces to a fully-implicit scheme (backward difference formula (BDF), Adams–Moulton, etc.). The general s-step IMEX linear multi-step scheme [5] reads

$$\frac{\sum_{j=-1}^{s-1} a_j\boldsymbol{u}^{n-j}}{\Delta t} = \sum_{j=-1}^{s-1} c_j\boldsymbol{F}^{\text{im}}_{n-j} + \sum_{j=0}^{s-1} b_j\boldsymbol{F}^{\text{ex}}_{n-j}, \tag{3}$$

where $\boldsymbol{F}^{\text{im}}_{n-j} = \boldsymbol{F}^{\text{im}}(\boldsymbol{u}^{n-j}, t_{n-j})$, $\boldsymbol{F}^{\text{ex}}_{n-j} = \boldsymbol{F}^{\text{ex}}(\boldsymbol{u}^{n-j}, t_{n-j})$ and the a_j's, b_j's, and c_j's are constant coefficients of the specific IMEX scheme. For an IMEX linear multi-stage scheme such as IMEX-RK (Runge–Kutta), the scheme at each stage within a time step has the same form as (3), so we will assume the multi-step setting hereafter, except that the detailed algorithms for the multi-stage setting is described in Sec. 5 and the related numerical results shown in Sec. 6.

In the case of the incompressible Navier–Stokes equations, the diffusion term and the (possibly stiff) forcing are typically treated implicitly to avoid too stringent of a stability constraint ($\Delta t \le Ch^2/\nu$ for some $C > 0$), while the choice for the advection term is not as clear.

IMEX. If we treat the advection term explicitly, i.e.

$$\boldsymbol{F}^{\text{im}}_{n-j} = \nu\nabla^2\boldsymbol{u}^{n-j} - \nabla p^{n-j} + \boldsymbol{f}^{n-j}, \qquad \boldsymbol{F}^{\text{ex}}_{n-j} = -\boldsymbol{u}^{n-j}\cdot\nabla\boldsymbol{u}^{n-j},$$

and use an IMEX scheme, we avoid solving a nonlinear system but Δt is subject to a stability constraint ($\Delta t \le Ch/U_0$ for some $C > 0$, with U_0 a characteristic flow velocity). In this case, at each time-step, we solve a Stokes system

$$\begin{cases} \left(\dfrac{a}{\Delta t} - \nu\nabla^2\right)\boldsymbol{u}^{n+1} + \nabla p^{n+1} = \boldsymbol{f}^{n+1,n}_{\text{rhs}} & \text{(a)} \\ \nabla\cdot\boldsymbol{u}^{n+1} = 0 & \text{(b)} \end{cases} \tag{4}$$

where $a = a_{-1}/c_{-1}$ and $\boldsymbol{f}^{n+1,n}_{\text{rhs}}$ contains every term of (3) that is known before the system is evolved to t_{n+1}, i.e.

[2] In a variational formulation of (1) [9], the velocity is the solution to a constrained optimization problem and the pressure is the Lagrangian multiplier to enforce the divergence-free constraint.

[3] Here being explicit or implicit is with respect to only the momentum equation (1a) with the pressure term viewed as a given forcing. Some literature, such as Sec. 7.3.2 of [21], also present "fully explicit" schemes. Technically there is no genuine "fully explicit" scheme for (1), because solving a linear system is inevitable due to the way $\boldsymbol{u}$ and p are coupled. Even when a fully explicit time-marching scheme is used, we still solve a linear system for the Poisson equation for p^n to enforce the divergence-free constraint on $\boldsymbol{u}^{n+1}$. Therefore, treating the pressure "explicitly" in time marching saves no computational efforts and one might as well just treat it implicitly.

$$\boldsymbol{f}_{\text{rhs}}^{n+1,n}=\boldsymbol{f}^{n+1}+\frac{1}{c_{-1}}\left(-\frac{\sum_{j=0}^{s-1}a_j\boldsymbol{u}^{n-j}}{\Delta t}+\sum_{j=0}^{s-1}c_j\boldsymbol{F}_{n-j}^{\text{im}}+\sum_{j=0}^{s-1}b_j\boldsymbol{F}_{n-j}^{\text{ex}}\right).$$

With some spatial discretization schemes applied to (4), the fully-discrete linear system can be written in matrix notation:

$$\begin{bmatrix}\boldsymbol{A} & \boldsymbol{G}\\ -\boldsymbol{D} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}^{n+1}\\ p^{n+1}\end{bmatrix}=\begin{bmatrix}\boldsymbol{f}_{\text{rhs}}^{n+1,n}+\text{bc}_{\text{mom}}^{n+1}\\ \text{bc}_{\text{con}}^{n+1}\end{bmatrix}. \tag{5}$$

Here $\boldsymbol{A}=(a/\Delta t)\boldsymbol{I}-\nu\boldsymbol{L}_{\boldsymbol{u}}$ is SPD (Symmetric Positive Definite) with $\boldsymbol{L}_{\boldsymbol{u}}$ being the discrete vector Laplacian and $\boldsymbol{A}$ is fixed in the whole simulation. $\boldsymbol{G}$ and $\boldsymbol{D}$ are the discrete gradient and divergence operators, respectively. They satisfy $\boldsymbol{G}^{\text{T}}=-\boldsymbol{D}$ and $\boldsymbol{L}_p=\boldsymbol{D}\boldsymbol{G}=-\boldsymbol{G}^{\text{T}}\boldsymbol{G}$ is the discrete scalar Laplacian. Note that we negate the continuity equation in (5) to make the linear system symmetric. The conditions $\text{bc}_{\text{mom}}^{n+1}$ relate to the discretization of the diffusion term in the momentum equations and account for the boundary values prescribed to the normal and tangential boundary conditions of velocity at t_{n+1}, while $\text{bc}_{\text{con}}^{n+1}$ relates to the divergence term in the continuity equation and involves the boundary values associated with only the normal boundary conditions of velocity at t_{n+1}. For wall bounded flows, i.e. without any inflow or outflow boundaries, we have $\boldsymbol{u}\cdot\boldsymbol{n}=0$ everywhere at the boundary, so $\text{bc}_{\text{con}}^{n+1}=\boldsymbol{0}$. Note that we use $\boldsymbol{u}$, p and $\boldsymbol{f}_{\text{rhs}}$ to denote both their continuous and discrete counterparts and which one is referred to should be clear from contexts.

Fully-implicit. If we also treat the advection implicitly, i.e.

$$\boldsymbol{F}_{n-j}^{\text{im}}=\nu\nabla^2\boldsymbol{u}^{n-j}-\nabla p^{n-j}+\boldsymbol{f}^{n-j}-\boldsymbol{u}^{n-j}\cdot\nabla\boldsymbol{u}^{n-j},\qquad \boldsymbol{F}_{n-j}^{\text{ex}}=0,$$

we usually eliminate the stability constraint so a larger Δt can be used, but now a nonlinear system needs to be solved at each time step, i.e.

$$\begin{cases}\left(\frac{a}{\Delta t}-\nu\nabla^2+\boldsymbol{u}^{n+1}\cdot\nabla\right)\boldsymbol{u}^{n+1}+\nabla p^{n+1}=\boldsymbol{f}_{\text{rhs}}^{n+1,n} & \text{(a)}\\ \nabla\cdot\boldsymbol{u}^{n+1}=0 & \text{(b)}\end{cases} \tag{6}$$

where

$$\boldsymbol{f}_{\text{rhs}}^{n+1,n}=\boldsymbol{f}^{n+1}+\frac{1}{c_{-1}}\left(-\frac{\sum_{j=0}^{s-1}a_j\boldsymbol{u}^{n-j}}{\Delta t}+\sum_{j=0}^{s-1}c_j\boldsymbol{F}_{n-j}^{\text{im}}\right).$$

There are various ways to solve (6) iteratively [58]. One way is to directly apply a general nonlinear root-finding algorithm such as Newton–Raphson (e.g. [43,34]). The $(k+1)$-th Newton–Raphson iteration then simply solves the system linearized around the solution from the previous iteration k, i.e.

$$\begin{cases}\left(\frac{a}{\Delta t}-\nu\nabla^2+\boldsymbol{u}_k^{n+1}\cdot\nabla+\left(\nabla\boldsymbol{u}_k^{n+1}\right)^{\text{T}}\cdot\right)\boldsymbol{u}_{k+1}^{n+1}+\nabla p_{k+1}^{n+1}=\boldsymbol{f}_{\text{rhs,NR}}^{n+1,n} & \text{(a)}\\ \nabla\cdot\boldsymbol{u}_{k+1}^{n+1}=0, & \text{(b)}\end{cases} \tag{7}$$

where $\boldsymbol{f}_{\text{rhs,NR}}^{n+1,n}=\boldsymbol{f}_{\text{rhs}}^{n+1,n}+\boldsymbol{u}_k^{n+1}\cdot\nabla\boldsymbol{u}_k^{n+1}$. The initial guess can be $\boldsymbol{u}_0^{n+1}=\boldsymbol{u}^n$. Note that

$$\text{D}(\boldsymbol{u}\cdot\nabla\boldsymbol{u})|_{\boldsymbol{u}_k^{n+1}}(\cdot)=\boldsymbol{u}_k^{n+1}\cdot\nabla(\cdot)+\left(\nabla\boldsymbol{u}_k^{n+1}\right)^{\text{T}}\cdot(\cdot)$$

is the Fréchet derivative of the advection term at $\boldsymbol{u}_k^{n+1}$. In (7a), the term $\boldsymbol{u}_k^{n+1}\cdot\nabla\boldsymbol{u}_{k+1}^{n+1}$ is a linear advection term and $\left(\nabla\boldsymbol{u}_k^{n+1}\right)^{\text{T}}\cdot\boldsymbol{u}_{k+1}^{n+1}$ a linear "reaction term". The fully-discrete counterpart of (7) can also be written in the form of (5), but $\boldsymbol{A}$ is then no longer symmetric and varies at every iteration. Newton–Raphson has quadratic convergence if the initial guess is close to the fixed point, which is the case if Δt is small enough.

Since the "reaction term" $\left(\nabla\boldsymbol{u}_k^{n+1}\right)^{\text{T}}\cdot\boldsymbol{u}_{k+1}^{n+1}$ does not appear in the original equation (1), implementing (7) requires extra coding efforts. Moreover, the reaction term can worsen the conditioning of the linear system in some cases [58]. Therefore, often a simpler fixed-point iteration where only the linear advection term retained is used, which yields an Oseen system

$$\begin{cases}\left(\frac{a}{\Delta t}-\nu\nabla^2+\boldsymbol{u}_k^{n+1}\cdot\nabla\right)\boldsymbol{u}_{k+1}^{n+1}+\nabla p_{k+1}^{n+1}=\boldsymbol{f}_{\text{rhs}}^{n+1,n} & \text{(a)}\\ \nabla\cdot\boldsymbol{u}_{k+1}^{n+1}=0, & \text{(b)}\end{cases} \tag{8}$$

whose fully-discrete form still has a varying and non-symmetric $\boldsymbol{A}$. In principle, this iteration only converges linearly, e.g. [34].

Of course, instead of a linearization, we can also just use the solution from the previous iteration to evaluate the whole advection term and thus each iteration requires solving only a Stokes system

$$\begin{cases} \left(\frac{a}{\Delta t} - \nu\nabla^2\right)\boldsymbol{u}_{k+1}^{n+1} + \nabla p_{k+1}^{n+1} = \boldsymbol{f}_{\text{rhs}}^{n+1,n} - \boldsymbol{u}_k^{n+1}\cdot\nabla\boldsymbol{u}_k^{n+1} & \text{(a)} \\ \nabla\cdot\boldsymbol{u}_{k+1}^{n+1} = 0, & \text{(b)} \end{cases} \tag{9}$$

whose fully-discrete form has a fixed SPD matrix $\boldsymbol{A}$, as in the case of IMEX.

Note that in some literature, the advection term is linearized around the solution at the previous time step and the time-marching scheme is computed without iteration. This is equivalent to solving (7) or (8) only once. Such treatment changes the aggregate time-marching scheme and the order of accuracy can be compromised [58] if the order of the time-marching scheme is higher than the order of the linearization applied to the advection term. If the goal is only the final steady state (e.g. using pseudo time marching to solve a steady flow problem), this compromise is fine, but if the goal is the transient flow development, such approximation should be avoided.

The requirement of iteration in each time step may sound unsatisfying but this is inevitable if we want to treat the nonlinear advection term implicitly. However, if some solution procedure of (1) requires iterating on $\boldsymbol{u}$ no matter whether the advection is treated explicitly or implicitly, such as the pressure-correction projection methods to be discussed in the next section, then the iteration for the implicitly-treated advection can actually get a "free" ride by being combined with the main iteration. That is, we use the same iteration to achieve two purposes. Although in the following two sections, we will focus almost exclusively on the case of IMEX (4) because it allows us to perform rigorous analysis of convergence and efficiency of the proposed iterated pressure-correction schemes, we will discuss how to deal with fully-implicit time-marching schemes with our proposed methods in Sec. 5.3.

3. Handling velocity–pressure coupling

The strategy to handle the velocity–pressure coupling is independent of the choice of the time-marching scheme. In other words, any of the following strategies can be applied to any fully-implicit or implicit–explicit (IMEX) time-marching schemes, although we will base our analysis on the case of IMEX.

3.1. Fully-coupled systems

A straightforward way to handle the velocity–pressure coupling is to treat the coupling as it is and numerically solve (1a) and (1b) simultaneously in each time step. Schemes based on this strategy are referred to as fully-coupled schemes.

The fully-coupled system (5) is a symmetric indefinite linear system which is also called a saddle-point problem. Its solution is known to be computationally expensive [9]. Over the years, a variety of techniques to decouple the computation of velocity and of pressure have been devised for a reasonable trade-off between accuracy and computational cost. They include pressure-Schur complement methods, SIMPLE-based methods, pressure-correction projection methods, and velocity-correction projection methods. Next, we compare and contrast them, and importantly, provide some new inter-connections.

3.2. Pressure-Schur complement methods

In (5), we can formally manipulate the first equation to obtain an expression for $\boldsymbol{u}^{n+1}$ in terms of p^{n+1}:

$$\boldsymbol{u}^{n+1} = \boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1} - \boldsymbol{G}p^{n+1}) \tag{10}$$

Inserting this in the second equation in (5), we obtain an equation that involves only pressure but no velocity:

$$\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}p^{n+1} = \boldsymbol{D}\boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1}) + \text{bc}_{\text{con}}^{n+1}. \tag{11}$$

It is thus possible to first solve for the pressure using (11) and recover the velocity with (10). This technique is called the Schur complement [69] method because $\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}$ is exactly the Schur complement of $\boldsymbol{A}$ in (5). This is commonly used for physical problems with a conservation equation coupled with a constitutive equation (e.g. see [15] for a nodal analysis in electrical engineering).

For the incompressible Navier–Stokes equations, $\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}$ is referred to as the *pressure-Schur complement* and (11) as the pressure-Schur complement equation [60]. Eqs. (10) and (11) are simply the Uzawa's scheme expressed in another analytically equivalent form. The main difficulty of the Schur complement method lies in solving the linear system of $\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}$. Direct methods are infeasible because $\boldsymbol{A}^{-1}$ is dense, while iterative methods, whether stationary or Krylov subspace methods, require a good preconditioner of $\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}$ for fast convergence [66]. For example, a Richardson iteration applied to the pressure-Schur complement equation with preconditioner $\boldsymbol{P}$ is given by,

$$p_{k+1}^{n+1} = p_k^{n+1} - \boldsymbol{P}^{-1}\left(\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}p_k^{n+1} - \boldsymbol{b}^{n+1}\right), \tag{12}$$

where $\boldsymbol{b}^{n+1} = \boldsymbol{D}\boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1}) + \text{bc}_{\text{con}}^{n+1}$.

3.3. Pressure-correction projection methods

Commonly-used techniques for velocity–pressure decoupling are the projection methods introduced by [14] and [56], and reviewed in [24]. Here, we only consider the incremental pressure-correction schemes, in both non-rotational and rotational form. The schemes are well-known and are outlined here mostly to introduce some concepts and the notation.

Again, we assume that an IMEX time-marching scheme (4) is used and the pressure-correction scheme involves the following four steps:

$$\left(\frac{a}{\Delta t} - \nu\nabla^2\right)\boldsymbol{u}_*^{n+1} + \nabla p^n = \boldsymbol{f}_{\text{rhs}}^{n+1,n}, \tag{13a}$$

$$\nabla^2 q = \frac{a}{\Delta t}\nabla\cdot\boldsymbol{u}_*^{n+1}, \tag{13b}$$

$$p^{n+1} = p^n + q \underbrace{-\,\nu\nabla\cdot\boldsymbol{u}_*^{n+1}}_{\text{Rotational corr}}, \tag{13c}$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}_*^{n+1} - \frac{\Delta t}{a}\nabla q\,. \tag{13d}$$

Here p^n is used as a guess of p^{n+1} to predict a non-divergence-free velocity field $\boldsymbol{u}_*^{n+1}$ with the momentum equation. After $\boldsymbol{u}_*^{n+1}$ is obtained from the prediction step (13a), the auxiliary variable q related to pressure correction is computed by solving the Poisson equation (13b), which ensures the corrected velocity $\boldsymbol{u}^{n+1}$ in (13d) to be divergence-free. For wall-bounded flows, (13d) implies that a zero Neumann boundary condition $\partial q/\partial\boldsymbol{n} = 0$ should be imposed to this Poisson equation. Once q is known, the pressure is updated to p^{n+1} through the pressure correction (13c), which lends its name to the method. Finally, (13d) yields a divergence-free corrected velocity. Note that in the Stokes case with the advection term absent, all terms are treated implicitly and the time-integration scheme becomes fully-implicit.

The term with an under-brace in (13c), introduced in [57] and popularized by [25] under the name "rotational correction", is only present when the rotational form is used and is otherwise absent. This rotational correction serves two important purposes. First, the rotational correction avoids an unphysical normal derivative of pressure at the boundary. Second, it renders the aggregate scheme (13) consistent with the fully-coupled scheme (4), up to the tangential velocity boundary conditions. Further details are given in Appendix A.

So far, the boundary condition for q has not been discussed and indeed, this is a recurrent issue with projection methods. There is no single "correct" boundary condition for q, and there are different choices [36], all of which result in some inconsistency in the boundary values of one or more variables. In fact, similar problems exist for other methods where a Laplacian operator is applied to the momentum equation (1a) [52].

Inconsistencies in the boundary conditions manifest themselves in a splitting error that limits the overall order of temporal convergence of the scheme, regardless of the order of the time integration scheme used. It can be shown that when the rotational correction is applied, the splitting error only results in an inconsistent tangential boundary condition on $\boldsymbol{u}^{n+1}$ and that the L_2-norm error in $\boldsymbol{u}^{n+1}$ is of order $O(\Delta t^2)$ and the L_2-norm error in p^{n+1} of order $O(\Delta t^{3/2})$, if a second- or higher-order time-marching scheme is used [25]. These are the best convergence orders for a projection method in general, although in some special cases, it is possible to achieve higher order temporal convergence [70].

3.4. Pressure correction as one iteration of the pressure-Schur complement

The algorithmic description (13) of the projection method involves various intermediate variables, but these intermediate variables can be eliminated when (13) is rewritten using the discrete matrix operators $\boldsymbol{A}$, $\boldsymbol{G}$ and $\boldsymbol{D}$. After the elimination (see Appendix B for derivations), we obtain

$$\boldsymbol{u}_*^{n+1} = \boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1} - \boldsymbol{G}p^n) \tag{14a}$$

$$p^{n+1} = p^n - \left(\frac{a}{\Delta t}\boldsymbol{L}_p^{-1} \underbrace{-\,\nu\boldsymbol{I}}_{\text{Rot}}\right)\left(\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}p^n - \boldsymbol{b}^{n+1}\right) \tag{14b}$$

$$\boldsymbol{u}^{n+1} = (\boldsymbol{I} - \boldsymbol{G}\boldsymbol{L}_p^{-1}\boldsymbol{D})\boldsymbol{u}_*^{n+1} - \boldsymbol{G}\boldsymbol{L}_p^{-1}\text{bc}_{\text{con}}^{n+1} \tag{14c}$$

with $\boldsymbol{b}^{n+1} = \boldsymbol{D}\boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1}) + \text{bc}_{\text{con}}^{n+1}$ as before. Here $\boldsymbol{L}_p$ is the discrete scalar Laplacian that satisfies $\boldsymbol{L}_p = \boldsymbol{D}\boldsymbol{G}$. We can see that (14b) is indeed of the same form as (12) with $k = 0$, initial guess $p_0^{n+1} = p^n$ and preconditioner $\boldsymbol{P}^{-1} = (a/\Delta t)\boldsymbol{L}_p^{-1}$ (or $\boldsymbol{P}^{-1} = (a/\Delta t)\boldsymbol{L}_p^{-1} - \nu\boldsymbol{I}$ in rotational form). This means that the incremental pressure-correction method is a preconditioned Richardson iterative method for the pressure-Schur complement equation. The preconditioner is the Laplacian and the first guess is the value at the previous time step. Only one iteration is performed and a bound is known for the error of the solution obtained after that one iteration. A more detailed explanation of this can be found in Sec. 2.2 of [60]. Interestingly, according to [60], the use of this preconditioner was first introduced in [23], even before the rotational

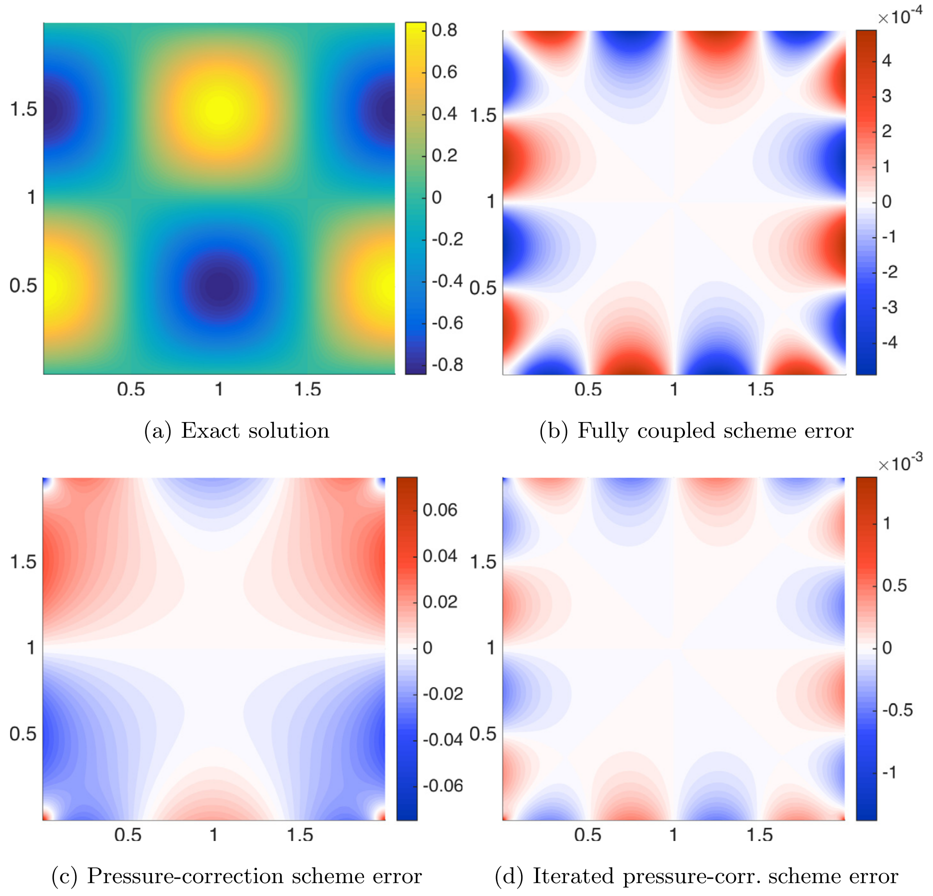


Fig. 1. (a) Analytical solution of the Stokes equations along with the errors obtained using (b) a fully-coupled scheme, (c) a regular pressure-correction scheme, and (d) an iterated pressure-correction scheme after 12 iterations (color bar limits differ among panels). The time-integration method is a BDF2 scheme with $\Delta t = .1$. The space resolution ($N = 200$, 4th order discretization in space) is chosen to be high enough such that spatial errors are negligible compared to temporal and splitting errors. This means that the error of the fully-coupled scheme is essentially that of the time integration scheme. As more iterations are performed, the error obtained with the iterated scheme converges to that of the fully-coupled scheme. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

correction was used in [57]. Moreover, as [60] pointed out, the popular Uzawa-like algorithms can also be formulated as such an iteration, but with a scalar matrix (or the mass matrix in pressure space for finite element) being the preconditioner.

Note that $-\boldsymbol{P}$ is semi-SPD (symmetric positive semi-definite) without the rotational correction, because $-\boldsymbol{L}_p = -\boldsymbol{D}\boldsymbol{G} = \boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}$ is semi-SPD. With the rotational correction, $-\boldsymbol{P}$ is SPD (symmetric positive definite) because $\nu > 0$. Since $\boldsymbol{L}_p$ is singular with a one-dimensional null space (due to the constant up to which a pressure field is determined), $\boldsymbol{L}_p^{-1}$ should be understood here as taking a regularized solution that satisfies a specified extra constraint for pressure.

To showcase the differences between a fully-coupled scheme, a regular pressure-correction, and an iterated pressure-correction (the latter two in rotational form), a classic manufactured solution for the Stoke equations [25] is utilized. This test case will be revisited in more details in Sec. 5.1. Fig. 1 illustrates the analytic solution and the errors of the three schemes. We find that the regular pressure-correction scheme (bottom-left) yields an error two orders of magnitude larger than the fully-coupled scheme (top-right) does, while the error of the iterated pressure-correction (bottom-right) becomes similar to the fully-coupled error. Without iteration, the splitting errors induced by the pressure-velocity decoupling dominates the time integration error. The iterated projection methods proposed in the next section attempts to address this issue of regular projection methods. We aim at schemes that avoid solving a coupled saddle-point problem and have splitting errors smaller than time integration errors.

3.5. Connection between pressure correction and SIMPLE-based methods

Another family of velocity–pressure decoupling schemes are based on the SIMPLE (Semi-Implicit Method for Pressure-Linked Equations; [12]) algorithm. Popular variants of SIMPLE include SIMPLEC (“C” for “Consistent”; [64]), SIMPLER (“R” for “Revised”; Sec. 6.8 of [47]) and PISO (Pressure Implicit with Splitting of Operator; [29]). The family of pressure correction methods and the family of SIMPLE-based methods were rarely compared and connected in literature. It is mostly in [44] that SIMPLE, SIMPLEC, and pressure-correction schemes are connected and compared, using a unified local-stencil framework.

In Appendix C, we first present this connection from an operator-splitting perspective that may be more natural and provides new insights. We show that both families of techniques utilize a prediction-correction approach. The difference lies in how the matrix operator $\boldsymbol{A} = (a/\Delta t)\boldsymbol{I} - \nu \boldsymbol{L_u}$ is split in the pressure and velocity correction step. The SIMPLE splits it into the diagonal and the off-diagonal part, which makes the resultant discrete correction equations have no continuous counterpart. In contrast, the pressure-correction projection method splits the operator into the time-derivative and the diffusion part, which readily leads to a continuous interpretation of the correction equations. SIMPLEC is a modification of SIMPLE that coincidentally coincide with the pressure-correction projection method in non-rotational form, "coincidentally" because this connection was not noticed when SIMPLEC was proposed in [64]. This connection helps explain why SIMPLEC improves SIMPLE.

Beyond SIMPLE and SIMPLEC, we also show that although SIMPLER and PISO are two modifications of SIMPLE, the key ideas therein do not depend on how the operator $\boldsymbol{A}$ is split and therefore, those ideas can be applied to the pressure-correction projection method as well. Indeed, we find that SIMPLER is simply a first iteration of an alternating projection scheme based on the Helmholtz–Hodge decomposition. PISO is an iterative scheme that treats diffusion explicitly for the exact velocity and pressure correction equations. These insights enable us to better evaluate how well they work and even further motivate the new iterated pressure-correction methods proposed here. Results of Appendix C may thus contribute to clarifying the big picture of different existing velocity–pressure decoupling techniques.

4. Iterated pressure-correction schemes

In Sec. 3, we formulated the regular incremental pressure-correction scheme in the form of a preconditioned fixed-point iteration of the pressure-Schur complement method. This idea naturally leads to the iterated projection methods. Since the regular pressure-correction scheme is equivalent to performing the first iteration, if we keep iterating and the iteration solution sequence converges, it should converge to the solution to the fully-coupled scheme.

This interpretation also explains why the pressure-correction scheme yields much larger errors than the fully-coupled scheme, because it is simply analogous to using an iterative scheme to solve a linear system and taking the result of the first iteration as an approximate solution. However, the fact that even this first iteration is decent, in the sense that we can already use it for time integration and obtain reasonably accurate results that will converge upon mesh and time-step refinement, indicates that the regular pressure-correction scheme serves as a very good preconditioner $\boldsymbol{P}^{-1}$ to the fixed-point iteration (12).

Moreover, the pressure-correction scheme also hints at a convenient way to carry out the iteration (12). If we rewrite (14) as

$$\boldsymbol{u}_*^{n+1} = \boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1} - \boldsymbol{G}p^n) \tag{15a}$$

$$p^{n+1} = p^n + \left(\frac{a}{\Delta t}\boldsymbol{L}_p^{-1} \underbrace{-\nu \boldsymbol{I}}_{\text{Rot}}\right)\left(\boldsymbol{D}\boldsymbol{u}_*^{n+1} + \text{bc}_{\text{con}}^{n+1}\right) \tag{15b}$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}_*^{n+1} - \boldsymbol{G}\boldsymbol{L}_p^{-1}(\boldsymbol{D}\boldsymbol{u}_*^{n+1} + \text{bc}_{\text{con}}^{n+1}), \tag{15c}$$

we can see that we can indeed use (15a) and (15b) to iterate on velocity and pressure alternately as

$$\boldsymbol{u}_{k+1}^{n+1} = \boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1} - \boldsymbol{G}p_k^{n+1}) \tag{16a}$$

$$p_{k+1}^{n+1} = p_k^n + \left(\frac{a}{\Delta t}\boldsymbol{L}_p^{-1} \underbrace{-\nu \boldsymbol{I}}_{\text{Rot}}\right)\left(\boldsymbol{D}\boldsymbol{u}_{k+1}^{n+1} + \text{bc}_{\text{con}}^{n+1}\right) \tag{16b}$$

and only in the end, after k_{n+1} iterations, apply $\boldsymbol{u}^{n+1} = \boldsymbol{u}_{k_{n+1}}^{n+1} - \boldsymbol{G}\boldsymbol{L}_p^{-1}(\boldsymbol{D}\boldsymbol{u}_{k_{n+1}}^{n+1} + \text{bc}_{\text{con}}^{n+1})$. Note that the last step is not necessary if we iterate until convergence, because the final predicted velocity $\boldsymbol{u}_{k_{n+1}}^{n+1}$ and the final corrected velocity $\boldsymbol{u}^{n+1}$ should coincide.

The fact that the predicted and corrected velocities coincide when the iteration converges provides a second motivation for iterated projection methods that does not originate from the pressure-Schur complement iteration. It is well known that in projection methods, neither of the predicted and the corrected velocity is superior to the other in terms of accuracy [24]. The predicted velocity is computed with both normal and tangential boundary conditions imposed, but is not divergence-free, while the corrected velocity is divergence-free, but is computed with only the normal boundary condition imposed. This motivates the idea of re-inserting the new pressure back into the prediction step as a better guess of p^{n+1} and performing the projection method again. Hopefully, if this is repeated enough times and convergence is achieved, the two velocities will coincide and the dilemma will go away.

Since both motivations point to iterated projection methods, we will investigate iterated pressure-correction schemes analytically and numerically. As a first intuition, Fig. 1d shows the result of 12 iterations of the pressure-correction scheme (in rotational form unless otherwise mentioned). We see that the error is indeed driven to that of the fully-coupled scheme, in both magnitude and spatial pattern.

This observation leads to a key question relating to the trade-off between accuracy and computational costs: To achieve the same accuracy, is it less costly to perform more iterations of the pressure-correction scheme within a fixed time step Δt than to reduce the time step with the regular scheme, i.e. with only one iteration in each time step?

Next, we show that the iterated pressure-correction schemes converge in both rotational and non-rotational forms, and that the former should be preferred because the convergence rate of the latter could get arbitrarily slow due to a boundary inconsistency in pressure. We also show how iteration schemes can efficiently reduce the splitting error and improve the global temporal order of convergence. We compare the resulting computational costs to that of simply decreasing the time step and show why iterated pressure-correction schemes are more efficient than regular ones. Moreover, we discuss the stopping criteria for the iteration and how Aitken acceleration speeds up the iteration.

4.1. General scheme

4.1.1. Scheme

The generic iterated pressure-correction scheme considerer here is

$$\boldsymbol{u}_{k+1}^{n+1} = \boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1} - \boldsymbol{G}p_k^{n+1}) \tag{17a}$$

$$p_{k+1}^{n+1} = p_k^n + \left(\frac{a}{\Delta t}\boldsymbol{L}_p^{-1} \underbrace{-\nu\boldsymbol{I}}_{\text{Rot}}\right)\left(\boldsymbol{D}\boldsymbol{u}_{k+1}^{n+1} + \text{bc}_{\text{con}}^{n+1}\right) \tag{17b}$$

where $\boldsymbol{A} = (a/\Delta t)\boldsymbol{I} - \nu\boldsymbol{L_u}$ and a is a positive parameter depending on the time-integration scheme used (e.g. $a = 1$ for BDF1-IMEX scheme).[4] Here $\boldsymbol{L_u}$ is the discrete vector Laplacian corresponding to the diffusion term in the momentum equations. This iteration is performed in each time step for a multi-step time-integration scheme, or in each stage within each time step for a multi-stage scheme such as Runge–Kutta.

Remark. Such iterated projection methods differ from using a projection method to precondition the original fully-coupled system, as has been studied extensively (e.g. [10]). In the latter, some iterative solver (e.g. GMRES) is applied to the fully-coupled system and the projection method serves only as a preconditioner. In contrast, an iterated projection method itself is not just a preconditioner, but a complete iterative scheme to solve the fully-coupled system.

4.1.2. Error types

When dealing with an incompressible flow solver using a projection method, one is confronted with three types of errors:

- Spatial error, which scales as h^p, where h is a parameter representative of the mesh resolution and p is the order of the spatial discretization scheme.
- Time-integration error, which scales as Δt^s with s being the order of the time-integration scheme. Typically s is the number of steps in a multi-step scheme (e.g. BDF) or of stages in a multi-stage scheme (e.g. Runge–Kutta).
- Splitting error, which scales with $\Delta t^{\tilde{s}}$, caused by applying a projection method on top of the time-integration scheme to decouple the velocity and pressure in computation

The total error of the numerical solution can thus be conceptually decomposed into three parts:

$$e_{\text{total}} = e_{\text{space}} + e_{\text{splitting}} + e_{\text{time}} = \mathrm{O}(h^p) + \mathrm{O}(\Delta t^{\tilde{s}}) + \mathrm{O}(\Delta t^s). \tag{18}$$

Note that technically such an expression is not precise, because errors from different sources may not be separable (e.g. mixed terms such as $\mathrm{O}(h^p\Delta t^{\tilde{s}})$ occur).

Since a focus here is on how to control $e_{\text{splitting}}$ relative to e_{time} for the iterated pressure-correction projection method, we assume h is small enough and e_{space} is negligible compared to the other two, and we ensure that this assumption holds in the numerical tests (e.g. see Sec. 6.1). Also since in general pressure converges more slowly than velocity due to the special role it plays in the equations, we will base our analysis on the errors in pressure.

When the time step is reduced, both the time-integration and splitting errors decrease. Hence, if one of the errors dominates in magnitude, it affects the rate at which total errors decrease. If we measure this apparent temporal order of convergence by successive reduction of Δt, we will get something between $\tilde{s}$ and s depending on which error dominates and by how much.

[4] We will identify what a is for particular time-marching schemes in Sec. 4. Roughly speaking, a is the ratio of the coefficient in front of $\boldsymbol{u}^{n+1}$ to the coefficient in front of the implicitly treated terms evaluated at t_{n+1} (e.g. $\nu\nabla^2\boldsymbol{u}^{n+1}$). See Appendix B for a derivation.

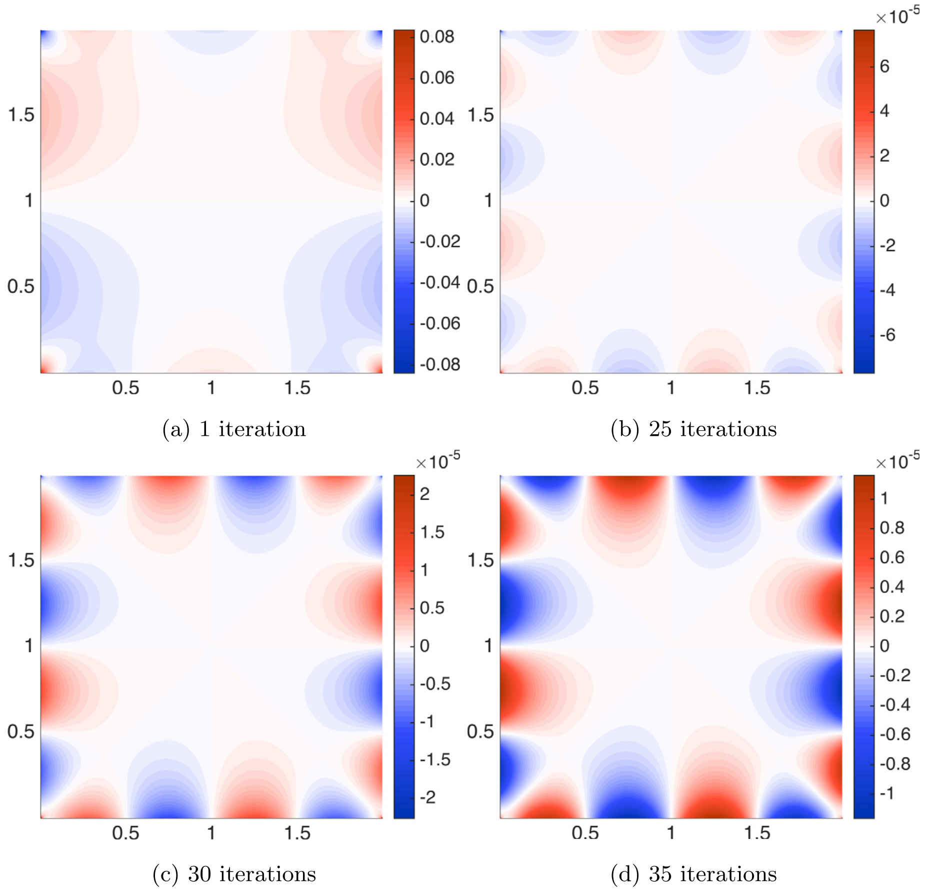


Fig. 2. Pressure errors after different numbers of pressure-correction iterations (color bar limits differ among panels). Same manufactured solution test case for Stokes equations as in Fig. 1, except that a BDF3 time-integration scheme is used. After 25 iterations, the shape of the time-integration error is already distinguishable, while after 35 iterations, the splitting error is negligible compared to the time-integration error and the shape of the total error is virtually identical to that obtained with a fully-coupled scheme. The space resolution ($N = 400$, 4th order discretization in space) is chosen to be high enough so that the spatial errors are negligible compared to the time-integration errors.

Typically we have $s > \tilde{s}$ for pressure, because with an incremental pressure-correction scheme, $\tilde{s} = 1.5$ in rotational form and $\tilde{s} = 1$ in non-rotational form, irrespective of what time-integration scheme is used [24]. Therefore, asymptotically the splitting error should dominate the time-integration error.

This can be verified numerically. In Fig. 1, the error of the fully-coupled scheme (Fig. 1b) is purely due to the time-integration error, while the error of a regular pressure-correction scheme (Fig. 1c) consists of both the time-integration and splitting errors. The latter is two orders of magnitude larger than the former, and the splitting error thus dominates. This is further verified by how pressure-correction iterations drive the latter to the former, as illustrated in Fig. 1d and in more details, in Fig. 2, which is from the same test case with a different time-integration scheme. After a number of iterations, the pressure field becomes closer and closer to that of the fully-coupled scheme.

For clarity of exposition, next, we utilize the following terms:

- *Iteration*: One pressure-correction iteration from k to $k + 1$ by (17).
- *Temporal convergence*: Apparent order of convergence of the aggregate scheme of time integration and a projection method, as observed when the time step is reduced.
- *Time-integration scheme convergence*: Intrinsic order of convergence of the time-integration scheme.

4.2. Convergence properties of iterated pressure-correction schemes

Here we first prove the convergence and derive the convergence rate of the iterated pressure-correction schemes. We utilize a theorem from [10], which used a non-incremental pressure-correction scheme in rotational form to precondition the fully-coupled system coming from discretizing the Stokes equation with the same grid and spatial schemes as we use for the numerical tests here, but with only a backward Euler scheme. That work analyzed the spectrum of the preconditioned system, and we use an intermediate result from there to bound the convergence rate of our new iterated pressure-correction schemes, first in rotational form. We then show that the convergence of the schemes in non-rotational form could be much slower and how this can be related to the artificial boundary conditions imposed on pressure due to the lack of the rotational correction. Our analysis applies to cases of any multi-step or multi-stage time-marching schemes.

Second, we show how the number of iterations can be controlled to make the splitting error decay at a desired order to raise the apparent order of convergence to that of the time-integration schemes. Moreover, we propose practical stopping criteria for implementation. Finally, analogous to how [6] used Aitken relaxation to accelerate an iterated velocity-correction scheme, we show how the same technique can accelerate an iterated pressure-correction scheme.

4.2.1. *Convergence guarantee of the iteration*

First, it is straightforward to verify that if convergence is achieved in (17) (with or without rotational correction), the converging point will satisfy (10) and (11) obtained by the pressure-Schur complement reduction and thus also satisfy the original fully-coupled system (5).

To analyze the convergence rate, writing the iteration (17) in the form of (12) in terms of pressure only is more convenient:

$$p_{k+1}^{n+1} = p_k^{n+1} - \boldsymbol{P}^{-1}\left(\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}p_k^{n+1} - \boldsymbol{b}^{n+1}\right), \tag{19}$$

where we recall that $-\boldsymbol{P}^{-1} = (-a/\Delta t)\boldsymbol{L}_p^{-1}$ (or $-\boldsymbol{P}^{-1} = (-a/\Delta t)\boldsymbol{L}_p^{-1} + \nu\boldsymbol{I}$ with the rotational correction) and $\boldsymbol{A} = (a/\Delta t)\boldsymbol{I} - \nu\boldsymbol{L}_{\boldsymbol{u}}$.

The iteration matrix of scheme (19) is $\boldsymbol{M} = \boldsymbol{I} - (-\boldsymbol{P}^{-1})(-\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G})$. For convenience to analyze the spectrum of $\boldsymbol{M}$, we can rearrange the scalar coefficient $a/\Delta t$ between $\boldsymbol{P}^{-1}$ and $\boldsymbol{A}$ without changing $\boldsymbol{M}$ to make $-\boldsymbol{P}^{-1} = -\boldsymbol{L}_p^{-1}$ (or $-\boldsymbol{P}^{-1} = -\boldsymbol{L}_p^{-1} + (\nu\Delta t/a)\boldsymbol{I}$ with the rotational correction) and $\boldsymbol{A} = \boldsymbol{I} - (\nu\Delta t/a)\boldsymbol{L}_{\boldsymbol{u}}$. For stationary iterative methods such as the classic Richardson iteration as well as the Jacobi and Gauss–Seidel iterations, the necessary and sufficient condition for convergence is $\rho(\boldsymbol{M}) < 1$, where $\rho(\boldsymbol{M}) = \max|\lambda(\boldsymbol{M})|$ is the spectral radius of $\boldsymbol{M}$ [66].

Since the values corresponding to the boundary conditions, the variables at the previous time level, and the forcing terms are all contained in the vector $\boldsymbol{b}^{n+1}$, they do not affect $\rho(\boldsymbol{M})$ and therefore have no effect on the convergence of the algorithm. Only the choice of space and time discretization does.

If we denote the (negative) pressure-Schur complement by $\boldsymbol{S} = -\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}$ and an eigenvalue of $\boldsymbol{M}$ by $\lambda(\boldsymbol{M})$, we have $\lambda(\boldsymbol{M}) = 1 - \lambda(-\boldsymbol{P}^{-1}\boldsymbol{S})$ and $\rho(\boldsymbol{M}) = \max\left|1 - \lambda(-\boldsymbol{P}^{-1}\boldsymbol{S})\right|$. We know pressure is only determined up to a uniform constant, which spans $\mathrm{Ker}(\boldsymbol{G})$, so here we restrict ourselves to $\mathrm{Ker}(\boldsymbol{G})^{\perp}$ to avoid dealing with the singularity of $\boldsymbol{S}$ and $\boldsymbol{L}_p$, which is immaterial. The regular eigen-problem $-\boldsymbol{P}^{-1}\boldsymbol{S}\boldsymbol{p} = \lambda\boldsymbol{p}$ can be equivalently transformed to the generalized eigen-problem $\boldsymbol{S}\boldsymbol{p} = \lambda(-\boldsymbol{P})\boldsymbol{p}$. Since the eigenvalues of the latter are real and positive because both $\boldsymbol{S}$ and $(-\boldsymbol{P})$ are SPD, although $-\boldsymbol{P}^{-1}\boldsymbol{S}$ may not even be symmetric, we know $\lambda(-\boldsymbol{P}^{-1}\boldsymbol{S}) \in \mathbb{R}_+$.

In the following, we will prove the convergence of the pressure correction iteration in both rotational and non-rotational form by showing that $\lambda(-\boldsymbol{P}^{-1}\boldsymbol{S}) \in [C, 1]$ with some $C > 0$. We denote the $-\boldsymbol{P}^{-1}$ in the case with and without the rotational correction by $-\boldsymbol{P}_{\mathrm{rot}}^{-1} = -\boldsymbol{L}_p^{-1} + (\nu\Delta t/a)\boldsymbol{I}$ and $-\boldsymbol{P}_{\mathrm{non}}^{-1} = -\boldsymbol{L}_p^{-1}$, respectively, and then by $\boldsymbol{M}_{\mathrm{rot}}$ / $\boldsymbol{M}_{\mathrm{non}}$ the $\boldsymbol{M}$ corresponding to $\boldsymbol{P}_{\mathrm{rot}}$ / $\boldsymbol{P}_{\mathrm{non}}$.

Pressure correction in rotational form. When the rotational form is used, we can directly use Theorem 2 in [10] to prove that $\beta^2 \le \lambda(-\boldsymbol{P}_{\mathrm{rot}}^{-1}\boldsymbol{S}) \le 1$, where β is the constant in the inf-sup condition (a.k.a. the Ladyzhenskaya–Babuška–Brezzi (LBB) condition), which is positive for a stable $\boldsymbol{u}$-p discretization and depends only on the domain geometry, but not on the resolution of the mesh. For clarity, we extract the relevant part of that theorem and restate it here using the notation in this paper.

Theorem 1 *(Thm. 2 of [10]). Let* $\boldsymbol{A} = \boldsymbol{I} - \varepsilon^2\boldsymbol{L}_{\boldsymbol{u}}$ *and* $-\boldsymbol{P}^{-1} = -\boldsymbol{L}_p^{-1} + \varepsilon^2\boldsymbol{I}$ *with* $\varepsilon^2 \ge 0$, *then we have* $\lambda(-\boldsymbol{P}^{-1}\boldsymbol{S}) \in [\beta^2, 1]$, *where* $\boldsymbol{S} = -\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}$.

In our case, we simply plug in $\varepsilon^2 = \nu\Delta t/a$ and thus conclude that $\rho(\boldsymbol{M}_{\mathrm{rot}}) \le 1 - \beta^2$. Estimating the spectrum of a non-symmetric matrix is in general hard, because we don't have the variational characterization of the eigenvalues in terms of a Rayleigh quotient, which is much easier to estimate than an eigenvalue. The key step of the proof of this theorem is relating the (nonzero) eigenvalues of $-\boldsymbol{P}^{-1}\boldsymbol{S}$ to the generalized Rayleigh quotient[5]

$$\frac{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}(-\boldsymbol{P}^{-1})\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{u}}, \quad \boldsymbol{u} \in \mathrm{Ker}(\boldsymbol{G}^{\mathrm{T}})^{\perp}\backslash\{0\}. \tag{20}$$

With this connection, by bounding (20), we obtain an upper and a lower bound for $\lambda(-\boldsymbol{P}^{-1}\boldsymbol{S})$. Next, we sketch the derivation in a slightly different form than in [10]. This modification later facilitates adapting the proof to the case without the rotational correction.

[5] In [10], the domain of $\boldsymbol{u}$ for (20) and some other equations is consistently written as $\boldsymbol{u} \notin \mathrm{Ker}(\boldsymbol{G}^{\mathrm{T}})$, which we believe is a typo.

To obtain both sides of the bound, we need the following identity:

$$\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}(-\boldsymbol{L}_{\mathrm{p}}^{-1})\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}=\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}, \quad \boldsymbol{u}\in \mathrm{Ker}(\boldsymbol{G}^{\mathrm{T}})^{\perp}. \tag{21}$$

This is true because since $\mathrm{Im}(\boldsymbol{G})=\mathrm{Ker}(\boldsymbol{G}^{\mathrm{T}})^{\perp}$, there is some p such that $\boldsymbol{u}=\boldsymbol{G}p$. Therefore,

$$\begin{aligned}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}(-\boldsymbol{L}_{\mathrm{p}}^{-1})\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}&=p^{\mathrm{T}}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G})^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}p\\&=p^{\mathrm{T}}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}p\\&=\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}.\end{aligned}$$

Another key result we need is the inequality

$$1\le\frac{\boldsymbol{u}^{\mathrm{T}}(-\boldsymbol{L}_{\boldsymbol{u}})\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}\le\frac{1}{\beta^2}, \quad \boldsymbol{u}\in\mathrm{Ker}(\boldsymbol{G}^{\mathrm{T}})^{\perp}\backslash\{0\}, \tag{22}$$

which is also proved in [10] and summarized in Remark 1 therein.

To bound (20), we use the particular structure of $\boldsymbol{P}_{\mathrm{rot}}$ and manipulate it as

$$\begin{aligned}\frac{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}(-\boldsymbol{P}_{\mathrm{rot}}^{-1})\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{u}}&=\frac{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}(-\boldsymbol{L}_{p}^{-1})\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}+\varepsilon^2\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}+\varepsilon^2\boldsymbol{u}^{\mathrm{T}}(-\boldsymbol{L}_{\boldsymbol{u}})\boldsymbol{u}}\\&=\frac{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}+\varepsilon^2\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}+\varepsilon^2\boldsymbol{u}^{\mathrm{T}}(-\boldsymbol{L}_{\boldsymbol{u}})\boldsymbol{u}}\\&=\frac{1+\varepsilon^2(\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u})/(\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u})}{1+\varepsilon^2(\boldsymbol{u}^{\mathrm{T}}(-\boldsymbol{L}_{\boldsymbol{u}})\boldsymbol{u})/(\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u})}.\end{aligned}$$

Because of (22), the last expression is a decreasing function of ε^2 over $[0,+\infty)$. Therefore, by taking $\varepsilon^2=0$ and $\varepsilon^2\to+\infty$, we have

$$\beta^2\le\frac{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}(-\boldsymbol{L}_{\boldsymbol{u}})\boldsymbol{u}}\le\frac{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}(-\boldsymbol{P}_{\mathrm{rot}}^{-1})\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{u}}\le 1, \tag{23}$$

and the desired bound $[\beta^2,1]$ follows.

Pressure correction in non-rotational form. When the non-rotational form is used, (20) becomes

$$\frac{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}(-\boldsymbol{P}_{\mathrm{non}}^{-1})\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{u}}=\frac{1}{1+\varepsilon^2(\boldsymbol{u}^{\mathrm{T}}(-\boldsymbol{L}_{\boldsymbol{u}})\boldsymbol{u})/(\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u})}.$$

The only difference with the rotational case is that the second term in the numerator is now missing. This is still a decreasing function of ε^2, but if we take $\varepsilon^2=0$ and $\varepsilon^2\to+\infty$, we only have the bound $[0,1]$, whose lower part is useless because we need a positive lower bound to guarantee convergence. It turns out that in this case we simply do not have a lower bound that is mesh- and viscosity-independent. For a particular ε^2, we have

$$\inf_{\boldsymbol{u}\in\mathrm{Ker}(\boldsymbol{G}^{\mathrm{T}})^{\perp}}\frac{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{G}(-\boldsymbol{P}_{\mathrm{non}}^{-1})\boldsymbol{G}^{\mathrm{T}}\boldsymbol{u}}{\boldsymbol{u}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{u}}\ge\frac{1}{1+\varepsilon^2\lambda_{\max}(-\boldsymbol{L}_{\boldsymbol{u}})}>0. \tag{24}$$

Hence, convergence is still guaranteed. For a given geometry and a particular spatial discretization, $\lambda_{\max}(-\boldsymbol{L}_{\boldsymbol{u}})$ typically scales as $1/h^2$, where h is characteristic of the mesh resolution. By comparing the generalized Rayleigh quotient in the two cases, we already know for any given ε^2, the lower bound is always smaller in the case without the rotational correction than otherwise, which implies slower convergence. However, what is worse and what is actually devastating is the mesh- and viscosity-dependence of the lower bound. Since $\varepsilon^2\lambda_{\max}(-\boldsymbol{L}_{\boldsymbol{u}})\propto\nu\Delta t/h^2$, when $\nu\Delta t/h^2$ is large, $\rho(\boldsymbol{M}_{\mathrm{non}})$ can become arbitrarily close to 1 and thus the convergence of the non-rotational pressure-correction iteration can become arbitrarily slow, much slower than its rotational counterpart. We will show this dramatic difference in the numerical results of Sec. 6. Conversely, to accelerate the convergence, one needs to make sure $\nu\Delta t/h^2$ is moderate, which imposes similar prohibitively stringent constraint on Δt as in the case of explicit diffusion treatment.

This phenomenon is likely due to the inconsistency of the pressure normal gradient at the boundary caused by the non-rotational scheme. When the boundary condition $\partial q/\partial\boldsymbol{n}=0$ is used for (13b) to impose the normal boundary condition on the corrected velocity through (13d), a side effect is that $\partial p/\partial\boldsymbol{n}$ is locked because of (13c) without a rotational correction, which is not physical. This inconsistency produces an artificial boundary layer in the solution when the non-rotational scheme is used [24]. However, we have just shown that the iterated pressure-correction scheme in non-rotational form nonetheless converges to the solution of the fully-coupled system. How can we reconcile this with the fact that analytically $\partial p_k^{n+1}/\partial\boldsymbol{n}$ is locked along iterations? A key observation is that when discretizing PDEs here (and in most cases in general), we only impose the boundary values “softly” through the forcing terms in the linear system. Based on this, a plausible explanation to the dilemma is that when the non-rotational scheme is iterated, the artificial boundary layer is further and further confined towards the boundary. Since all the cells next to the boundary have finite sizes, eventually this boundary layer effect becomes negligible to the discrete solution. The fact that the convergence slows down when $\nu\Delta t/h^2$ increases also supports this explanation. The higher the viscosity, the deeper into the interior will the boundary conditions be felt. The larger the time step, the larger will the normal pressure gradient inconsistency be, because $\left|\partial p^{n+1}/\partial \boldsymbol{n} - \partial p^n/\partial \boldsymbol{n}\right|$ increases with Δt. Finally, the finer the mesh, the smaller and closer to the boundary will the cells right next to the boundary be.

In summary, since the extra cost of the rotational correction is simply a vector addition, but the gain is a higher order scheme (larger $\tilde{s}$) and mesh-independent convergence behaviors if the scheme is iterated, one should always use the rotational form in practice.

Lastly, also note that when $\nu = 0$ and thus $\varepsilon^2 = 0$, the above analysis indicates that both $\rho(\boldsymbol{M}_{\text{rot}})$ and $\rho(\boldsymbol{M}_{\text{non}})$ are zero, which implies that one iteration, i.e. the classical projection method, yields the solution of the fully-coupled system. This makes sense because the rotational correction is proportional to ν, so if $\nu = 0$, this correction is null and should make no difference. Moreover, since $\nu = 0$ reduces the Navier–Stokes equations to the Euler equations (still incompressible), which admit only the normal velocity boundary condition, the inconsistency in the tangential boundary condition for the projection methods disappears. This explains why a classical pressure-correction scheme is totally equivalent to the fully-coupled system in this inviscid case.

4.2.2. Number of iterations needed to achieve a certain temporal convergence order

With the guaranteed convergence of the iterated projection methods, in principle we can keep iterating and drive the splitting error arbitrarily close to zero (e.g. machine epsilon in practice). However, since this convergence is only linear and each iteration is an application of the whole projection scheme, which requires solving several large linear systems, we still want to be judicious about the number of iterations and do not want to iterate more than needed.

The core issue here is that the lower order of convergence $\tilde{s} < s$ of the splitting error is the bottleneck that prevents the temporal convergence from achieving the order s of the time-integration scheme. To resolve this, we don’t need to eliminate the splitting error completely, but only need to make sure the remaining splitting error converges in time at an order of at least s. Since the remaining splitting error can be controlled by the number of iterations k, we discuss next how this can be achieved by setting the number of iterations to be Δt-dependent and by estimating the ideal k_* for the desired order of accuracy.

If only one iteration is performed, as for the regular pressure-correction method, the splitting error $\|e_1\|$ on the pressure is such that[6]

$$\|e_1\| = \Theta(\Delta t^{\tilde{s}}). \tag{25}$$

Now, at each time step, we perform a sufficient number k_* of iterations such that the temporal order of convergence of the splitting error $\|e_{k_*}\|$ is $\tilde{s}_* > \tilde{s}$:

$$\|e_{k_*}\| = \Theta\left(\Delta t^{\tilde{s}_*}\right). \tag{26}$$

If $R < 1$ is the convergence rate of the iteration under some norm $\|\cdot\|$, we have $\|e_{k_*}\| \le \|e_1\| R^{k_*}$. Hence, we want to ensure

$$\|e_1\| R^{k_*} = \Theta\left(\Delta t^{\tilde{s}_*}\right). \tag{27}$$

Using (25), for the needed k_* iterations, we obtain

$$R^{k_*} = \Theta\left(\Delta t^{\tilde{s}_* - \tilde{s}}\right). \tag{28}$$

Therefore, asymptotically, we can set the number of iterations k_* to

$$k_* = (\tilde{s}_* - \tilde{s}) \log_R(\Delta t) = \Theta(\ln(\Delta t)). \tag{29}$$

As Δt tends to 0, k_* tends to infinity, which means that as the time step gets smaller, more iterations are needed to achieve the desired order of convergence for the splitting error. Next we show that iterated projection methods with k_* set by (29) are computationally more efficient than their regular counterparts.

[6] Note that $f(\Delta t) = \Theta(g(\Delta t))$ if for Δt small enough, we have $C_1|g(\Delta t)| \le |f(\Delta t)| \le C_2|g(\Delta t)|$ for some positive C_1 and C_2. Intuitively $f(\Delta t) = \Theta(g(\Delta t))$ means $f(\Delta t)$ and $g(\Delta t)$ change at the same speed with Δt asymptotically.

4.2.3. Cost comparison between regular and iterated projection methods

The splitting error e_1 for the regular pressure-correction scheme converges in order $\tilde{s}$ asymptotically (25), i.e.

$$e_1 = \Theta(\Delta t^{\tilde{s}}), \tag{30}$$

and the temporal error (both time-integration and splitting) for an iterated scheme with enough iterations can converge at most as fast as the time-integration error, namely at order s. Therefore, the most economic way is to set k_* in (26)–(29) such that $\tilde{s}_* = s$, i.e.

$$e_{k_*} = \Theta(\Delta t^{s}). \tag{31}$$

Let us now assume that if the time step used for the first scheme is Δt_1, the time step needed for the second scheme to have an error e_{k_*} asymptotically equivalent to e_1 is Δt_2. Neglecting higher order terms, $e_1 = \Theta(e_2)$ implies:

$$\Delta t_1 = \Theta\left(\Delta t_2^{s/\tilde{s}}\right). \tag{32}$$

Using this relation (32) and the number of iterations (29), the computational costs c_1 and c_2 for each of the two schemes can then be expressed in terms of the number of projections, respectively, as

$$c_1 = \frac{T}{\Delta t_1} = \Theta\left(\Delta t_2^{-s/\tilde{s}}\right) \tag{33}$$

and

$$c_2 = k_* \frac{T}{\Delta t_2} = \Theta(\ln(\Delta t_2)) \cdot \Theta\left(\Delta t_2^{-1}\right) = \Theta\left(\frac{\ln(\Delta t_2)}{\Delta t_2}\right). \tag{34}$$

The ratio of the costs is thus,

$$\frac{c_2}{c_1} = \Theta\left(\frac{\ln(\Delta t_2)}{\Delta t_2^{1-s/\tilde{s}}}\right), \tag{35}$$

which converges to 0 as Δt_2 tends to 0, because $s > \tilde{s}$ and thus $1 - s/\tilde{s} < 0$. Therefore, asymptotically, the iterated scheme costs less to achieve the same accuracy than the regular scheme does, and is thus computationally more efficient. More importantly, this cost ratio is not just some fixed small fraction, but will decrease to 0 as Δt is reduced, so in practice, as long as Δt falls below the threshold where asymptotic results on errors hold reasonably well, the iterated scheme can be expected to outperform its regular counterpart.

4.3. Stopping criteria: convergence order and magnitude of splitting errors

As with any iterative method, one difficulty is to decide when to stop iterating, and indeed choosing a stopping criterion is an art in itself. Stopping criteria often involve the value of the residual and, for the best ones, knowledge of the iteration matrix [8], neither of which are often readily available. In any case, it is necessary to choose a tolerance for the error, and if possible, this choice should not be arbitrary.

An advantage of projection methods is that we know a bound of the error in terms of the time step after one iteration. We will now show that using this information, it is possible to devise a stopping criterion that depends only on the parameters at hand, and also allows some level of control on the overall temporal convergence rate of the scheme.

Let us assume that for a certain projection scheme, the splitting error is of order $\tilde{s}$ and let p_{ex}^{n+1} be the exact solution,

$$\|p_1^{n+1} - p_{\text{ex}}^{n+1}\| \leq C_1 \Delta t^{\tilde{s}}. \tag{36}$$

Since p_{ex}^{n+1} is unknown, this bound is of little practical value. However, we do know the value of the second pressure correction $\|p_1^{n+1} - p_2^{n+1}\|$ for which we can derive a bound using (36):

$$\|p_1^{n+1} - p_2^{n+1}\| = \|p_1^{n+1} - p_{ex}^{n+1} + p_{ex}^{n+1} - p_2^{n+1}\| \leq \|p_1^{n+1} - p_{ex}^{n+1}\| + \|p_2^{n+1} - p_{ex}^{n+1}\|. \tag{37}$$

Recall that $R < 1$ is the convergence rate of the iteration under some norm $\|\cdot\|$. Therefore,

$$\|p_2^{n+1} - p_{ex}^{n+1}\| \leq R\|p_1^{n+1} - p_{ex}^{n+1}\| \leq RC_1 \Delta t^{\tilde{s}}, \tag{38}$$

so

$$\|p_1^{n+1} - p_2^{n+1}\| \leq (1+R) C_1 \Delta t^{\tilde{s}}. \tag{39}$$

We now utilize this simple result to devise a stopping criterion. As mentioned previously, we want a scheme that converges at order $\tilde{s}$, in which case we would stop at the iteration k when, if for the first time,

$$\|p_k^{n+1} - p_{ex}^{n+1}\| \le C\Delta t^s \tag{40}$$

for some constant C. We now show that if we choose the following stopping criterion,

$$\|p_{k-1}^{n+1} - p_k^{n+1}\| \le C_2 \|p_1^{n+1} - p_2^{n+1}\| \Delta t^{s-\tilde{s}}\,, \tag{41}$$

with an arbitrary constant C_2, (40) will be satisfied.

First, at each iteration, we have

$$\|p_k^{n+1} - p_{ex}^{n+1}\| \le R\|p_{k-1}^{n+1} - p_{ex}^{n+1}\|\,. \tag{42}$$

With this, the error at iteration k can be bounded by a multiple of the pressure correction at iteration k, i.e. $\|p_{k-1}^{n+1} - p_k^{n+1}\|$, as follows:

$$\begin{aligned} \|p_{k-1}^{n+1} - p_{ex}^{n+1}\| &\le \|p_{k-1}^{n+1} - p_k^{n+1}\| + \|p_k^{n+1} - p_{ex}^{n+1}\| \\ &\le \|p_{k-1}^{n+1} - p_k^{n+1}\| + R\|p_{k-1}^{n+1} - p_{ex}^{n+1}\|\,, \end{aligned} \tag{43}$$

and thus, since $1 - R > 0$,

$$\|p_k^{n+1} - p_{ex}^{n+1}\| \le R\|p_{k-1}^{n+1} - p_{ex}^{n+1}\| \le \frac{R}{1-R}\|p_{k-1}^{n+1} - p_k^{n+1}\|\,. \tag{44}$$

If we now choose (41) as our stopping criterion, using (44), we see that,

$$\|p_k^{n+1} - p_{ex}^{n+1}\| \le \frac{C_2 R}{1-R}\|p_1^{n+1} - p_2^{n+1}\|\Delta t^{s-\tilde{s}}\,. \tag{45}$$

Hence, using (45) and (39), we indeed obtain (40):

$$\|p_k^{n+1} - p_{ex}^{n+1}\| \le \frac{C_1 C_2 R(1+R)}{1-R}\Delta t^{\tilde{s}}\Delta t^{s-\tilde{s}} = C\Delta t^s\,. \tag{46}$$

This result is valid for any classic norm. In this study, we implement the criterion with the L_∞-norm and $C_2 = 1$.

Of course, (41) is only guaranteed to be effective asymptotically. The criterion (41) only ensures that when we successively reduce Δt, we achieve temporal convergence order s for the aggregate scheme. In practice, we are not only concerned about the convergence order; we also want the splitting error to be dominated by the time-integration error to avoid contaminating the accuracy of the time-integration scheme by the artificial boundary layer effects caused by the projection methods. However, (41) does not guarantee this for any finite Δt. Also there is an unspecified constant C_2 in (41), which does not affect the asymptotic behavior but surely affects the outcome of the iteration for a particular finite Δt. Indeed, for Δt given, without knowing R, C_1 and especially the time-integration error, one cannot tell a priori how small C_2 should be.

A first way to deal with this is to use (41) jointly with a common stopping criterion for iterations (i.e. exit only when both criteria are met),

$$\frac{\left\|p_{k-1}^{n+1} - p_k^{n+1}\right\|}{\left\|p_k^{n+1}\right\|} \le \varepsilon, \tag{47}$$

which directly detects how well the convergence has been achieved. If we have a rough estimate of the order of magnitude of the time-integration relative error, we can simply let ε be one or two orders of magnitude smaller than that to make sure the iteration does not stop until the splitting error magnitude goes below the time-integration error magnitude.

The time-integration error can be estimated by three simulations (e.g. using Δt, $\Delta t/2$ and $\Delta t/4$) for a short period of time, but only if the splitting error is eliminated. In order to completely eliminate the splitting error, we can use stopping criterion (47) with ε set to machine epsilon. If the spatial error is also dominated by the time-integration error, we can then estimate the time-integration error using (e.g. see [34])

$$s \approx \log_2 \frac{\left\|p_{\Delta t} - p_{\Delta t/2}\right\|}{\left\|p_{\Delta t/2} - p_{\Delta t/4}\right\|}, \qquad \|p_{\Delta t} - p_{\text{ex}}\| \approx \frac{\left\|p_{\Delta t} - p_{\Delta t/2}\right\|}{1 - 1/2^s}, \tag{48}$$

where $p_{\Delta t}$ represents the numerical solution obtained with time step size Δt.

If an estimate of $\|p_{\Delta t} - p_{\text{ex}}\|$ is available, one can also use the convergence rate R of the pressure-correction iteration to obtain a second stopping criterion. An estimate for R is provided by

$$R \approx \frac{\left\|p_k - p_{k+1}\right\|}{\left\|p_{k-1} - p_k\right\|} \tag{49}$$

in a sample iteration sequence. With R and $\|p_{\Delta t} - p_{\text{ex}}\|$, together with

$$\|p_k - p_{\Delta t}\| \leq \frac{R}{1-R}\|p_{k-1} - p_k\|, \tag{50}$$

which can be derived similarly as (44), we can use the stopping criterion

$$\left\|p_{k-1} - p_k\right\| \leq \frac{1-R}{R}\,\|p_{\Delta t} - p_{\mathrm{ex}}\| \tag{51}$$

to ensure $\|p_k - p_{\Delta t}\| \leq \|p_{\Delta t} - p_{\mathrm{exact}}\|$, i.e. the splitting error dominated by the time-integration error.

A third criterion is from the residual of the continuity equation, i.e.

$$\left\|\boldsymbol{D}\boldsymbol{u}_{k+1}^{n+1} + \mathrm{bc}_{\mathrm{con}}\right\| \leq \varepsilon, \tag{52}$$

because if the pressure is close to the converging point, the predicted velocity using this pressure as a guess should be close to being divergence-free.

Note that since $\tilde{s} < s$, if a classic projection method without iteration is used, asymptotically the splitting error dominates the time-integration error. Therefore, if we use an iterated projection method with $\tilde{s}_* > s$ but without requiring the remaining splitting error to be always dominated by the time-integration error, we may observe a super-convergence phenomenon at a range of Δt where the apparent time convergence order becomes $\tilde{s}_*$, which is even faster than what the time-integration scheme promises. Such super-convergence implied by our theory is verified and investigated numerically in Sec. 6.2.

4.4. Convergence acceleration

Stationary iterative methods converge only linearly with an arbitrary initial guess when the spectral radius of the iteration matrix is smaller than 1. A good candidate to accelerate such iterations is the Aitken relaxation method. There are many variants of this method in the case of a vector sequence, which differ in the scheme and in the choice of initial parameters [39]. Here we selected the scheme devised by [2], which was also used by [6] for an iterated velocity-correction scheme. Given an iteration scheme

$$\boldsymbol{x}_{k+1} = F(\boldsymbol{x}_k) \tag{53}$$

where $\boldsymbol{x}_k$ is a vector, the Aitken relaxation scheme is defined as

$$\boldsymbol{x}_{k+1} = F(\boldsymbol{x}_k) + \omega_k\big(\boldsymbol{x}_k - F(\boldsymbol{x}_k)\big)\,, \tag{54}$$

where ω_k is a dynamic relaxation parameter, computed as follows,

$$\omega_k = \omega_{k-1} + (\omega_{k-1} - 1)\frac{\boldsymbol{d}_k^{\mathrm{T}}(\boldsymbol{d}_k - \boldsymbol{d}_{k-1})}{\|\boldsymbol{d}_k - \boldsymbol{d}_{k-1}\|_2^2}\,, \tag{55}$$

and $\boldsymbol{d}_k$ is defined by $\boldsymbol{d}_k = \boldsymbol{x}_k - F(\boldsymbol{x}_k)$.

Here, we apply the Aitken relaxation to pressure at the end of each iteration, so $\boldsymbol{x}_k = p_k^{n+1}$ and $F(\cdot)$ is the operation (19) that computes p_{k+1}^{n+1} from p_k^{n+1}. For implementation, we employ the version of [7], which is a slight modification of [2] that includes a stabilization parameter. The initial ω_0 can be chosen to be a large number [7] or the value at the previous time step [6]. The stopping criterion (41) for the unaccelerated iteration translates to this case as,

$$\|\boldsymbol{d}_{k-1}^{n+1}\| \leq C_2\|\boldsymbol{d}_1^{n+1}\|\Delta t^{\tilde{s}-s}\,. \tag{56}$$

The extra cost of the Aitken relaxation (55) is mainly computing two vector inner products and one vector subtraction, which is negligible compared to the cost of one normal projection iteration. But, as we will see in the results of Sec. 5, this tiny effort significantly reduces the number of iterations needed.

5. Iterated pressure correction with common time-integrations

In the previous section, using the discrete version of the differential operators represented as matrices, we showed how pressure correction schemes can be iterated to control the splitting errors and we analyzed properties of this iteration. Now, we provide detailed algorithmic descriptions of how the iterated pressure correction is applied to several common types of time-integration schemes.

5.1. General IMEX linear multi-step schemes

A regular pressure-correction scheme applied to (3) contains four steps:

1. Compute the predicted velocity with (3) except that p^{n+1} in the $j=-1$ implicit term $\boldsymbol{F}^{\text{im}}_{n+1}$ is replaced by a guess p^n.
2. Solve the Poisson equation for an auxiliary scalar field q.
3. Correct the pressure from p^n to p^{n+1}.
4. Correct the predicted velocity to recover the intended scheme (3).

To obtain the iterated version, we simply iterate through the first three steps by inserting the newly-obtained corrected pressure from step 3 back to step 1 as a new guess of p^{n+1}.

Algorithm 1 *(Iterated pressure correction for IMEX linear multi-step schemes).*
Input: $\boldsymbol{u}^n$ and p^n.

```
1. $p_0^{n+1} = p^n$
2. Iterate through $k = 0, \dots, (k_{n+1} - 1)$:
       $\frac{a_{-1}\boldsymbol{u}^{n+1}_{k+1} + \sum_{j=0}^{s-1} a_j \boldsymbol{u}^{n-j}}{\Delta t} = c_{-1}\left(\nu\nabla^2\boldsymbol{u}^{n+1}_{k+1} - \nabla p^{n+1}_k + \boldsymbol{f}^{n+1}\right) + \sum_{j=0}^{s-1} c_j \boldsymbol{F}^{\text{im}}_{n-j} + \sum_{j=0}^{s-1} b_j \boldsymbol{F}^{\text{ex}}_{n-j}$
       $\nabla^2 q_{k+1} = \frac{a_{-1}}{c_{-1}\Delta t}\nabla\cdot\boldsymbol{u}^{n+1}_{k+1}$
       $p^{n+1}_{k+1} = p^{n+1}_k + q_{k+1} - \nu\nabla\cdot\boldsymbol{u}^{n+1}_{k+1}$
3. $\boldsymbol{u}^{n+1} = \boldsymbol{u}^{n+1}_{k_{n+1}} - (c_{-1}\Delta t/a_{-1})\nabla q_{k_{n+1}}, \quad p^{n+1} = p^{n+1}_{k_{n+1}}$
```

Output: $\boldsymbol{u}^{n+1}$ and p^{n+1}.

Note that this scheme formulated as (17) will have $a = a_{-1}/c_{-1}$. Here k_{n+1} is the number of iterations performed at time step $(n+1)$, which can either be prescribed or be determined implicitly by a stopping criterion proposed in Sec. 3.3. Note that although upon convergence (i.e. after infinite iterations), the correction of $\boldsymbol{u}^{n+1}_{k_{n+1}}$ in the final step is null and thus redundant, we may still want to carry it out in practice (with only finite iterations), to ensure $\boldsymbol{u}^{n+1}$ satisfies the discrete divergence-free constraint up to only round-off errors.

A popular choice of the IMEX multi-step schemes is the family of s-step BDFs-IMEX schemes, which have $c_j = 0$ except for $c_{-1} = 1$:

$$\frac{\sum_{j=-1}^{s-1} a_j \boldsymbol{u}^{n-j}}{\Delta t} = \boldsymbol{F}^{\text{im}}_{n+1} + \sum_{j=0}^{s-1} b_j \boldsymbol{F}^{\text{ex}}_{n-j}. \tag{57}$$

The corresponding iterated version is as follows.

Algorithm 2 *(Iterated pressure correction for BDFs-IMEX schemes).*
Input: $\boldsymbol{u}^n$ and p^n.

```
1. $p_0^{n+1} = p^n$
2. Iterate through $k = 0, \dots, (k_{n+1} - 1)$:
       $\frac{a_{-1}\boldsymbol{u}^{n+1}_{k+1} + \sum_{j=0}^{s-1} a_j \boldsymbol{u}^{n-j}}{\Delta t} = \nu\nabla^2\boldsymbol{u}^{n+1}_{k+1} - \nabla p^{n+1}_k + \boldsymbol{f}^{n+1} + \sum_{j=0}^{s-1} b_j \boldsymbol{F}^{\text{ex}}_{n-j}$
       $\nabla^2 q_{k+1} = \frac{a_{-1}}{\Delta t}\nabla\cdot\boldsymbol{u}^{n+1}_{k+1}$
       $p^{n+1}_{k+1} = p^{n+1}_k + q_{k+1} - \nu\nabla\cdot\boldsymbol{u}^{n+1}_{k+1}$
3. $\boldsymbol{u}^{n+1} = \boldsymbol{u}^{n+1}_{k_{n+1}} - (\Delta t/a_{-1})\nabla q_{k_{n+1}}, \quad p^{n+1} = p^{n+1}_{k_{n+1}}$
```

Output: $\boldsymbol{u}^{n+1}$ and p^{n+1}.

5.2. General IMEX Runge–Kutta schemes

IMEX-RK (Runge–Kutta) schemes were first developed in [4]. We follow the procedure outlined in [62] except that the forcing term is treated implicitly. An s-stage IMEX-RK scheme can be represented by the following Butcher tableau:

$$\begin{array}{c|cccccc} 0 & 0 & 0 & 0 & \cdots & 0 \\ c_1 & a^{\text{ex}}_{1,0} & 0 & 0 & \cdots & 0 \\ c_2 & a^{\text{ex}}_{2,0} & a^{\text{ex}}_{2,1} & 0 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ c_{s-1} & a^{\text{ex}}_{s-1,0} & a^{\text{ex}}_{s-1,1} & \cdots & a^{\text{ex}}_{s-1,s-2} & 0 \\ \hline & b^{\text{ex}}_0 & b^{\text{ex}}_1 & \cdots & \cdots & b^{\text{ex}}_{s-1} \end{array} \tag{58}$$

$$\begin{array}{c|cccccc} 0 & 0 & 0 & 0 & \cdots & 0 \\ c_1 & a^{\text{im}}_{1,0} & a^{\text{im}}_{1,1} & 0 & \cdots & 0 \\ c_2 & a^{\text{im}}_{2,0} & a^{\text{im}}_{2,1} & a^{\text{im}}_{2,2} & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ c_{s-1} & a^{\text{im}}_{s-1,0} & a^{\text{im}}_{s-1,1} & \cdots & a^{\text{im}}_{s-1,s-2} & a^{\text{im}}_{s-1,s-1} \\ \hline & b^{\text{im}}_0 & b^{\text{im}}_1 & \cdots & \cdots & b^{\text{im}}_{s-1} \end{array} \tag{59}$$

The IMEX-RK time marching scheme applied to (2) is carried out as below:

$$\frac{\boldsymbol{u}^j - \boldsymbol{u}^{[n]}}{\Delta t} = \sum_{i=0}^{j} a^{\text{im}}_{j,i} \boldsymbol{F}^{\text{im}}_i + \sum_{i=0}^{j-1} a^{\text{ex}}_{j,i} \boldsymbol{F}^{\text{ex}}_i, \qquad j = 1, \dots, s-1, \tag{60a}$$

$$\frac{\boldsymbol{u}^{[n+1]} - \boldsymbol{u}^{[n]}}{\Delta t} = \sum_{j=0}^{s-1} b^{\text{im}}_j \boldsymbol{F}^{\text{im}}_j + \sum_{j=0}^{s-1} b^{\text{ex}}_j \boldsymbol{F}^{\text{ex}}_j, \tag{60b}$$

where $\boldsymbol{u}^{[n]}$ denotes the approximation of $\boldsymbol{u}(t_n)$ at the end of each time step and $\boldsymbol{u}^j$ denotes the approximation of $\boldsymbol{u}(t_n + c_j \Delta t)$ at each stage within a time step. $\boldsymbol{F}^{\text{im}}_j = \boldsymbol{F}^{\text{im}}(\boldsymbol{u}^j)$ and $\boldsymbol{F}^{\text{ex}}_j = \boldsymbol{F}^{\text{ex}}(\boldsymbol{u}^j)$ are the right hand side terms evaluated at different stages. Note also that $\boldsymbol{u}^0 = \boldsymbol{u}^{[n]}$.

In each stage of (60a), the scheme is analogous to a linear multi-step scheme (3), so the pressure correction can be similarly applied and iterated. If the scheme at stage j is formulated as (17), we will have $a = 1/a^{\text{im}}_{j,j}$. All the previous analysis on iteration based on a linear multi-step scheme still applies.

Algorithm 3 *(Iterated pressure correction for IMEX Runge–Kutta schemes).*
Input: $\boldsymbol{u}^{[n]}$ and $p^{[n]}$.

1. $\boldsymbol{u}^0 = \boldsymbol{u}^{[n]}$ and $p^0 = p^{[n]}$
2. Loop over RK stages: For $j = 1, \dots, (s-1)$,
   (a) $p^j_0 = p^{j-1}$
   (b) Iterate through $k = 0, \dots, (k_j - 1)$:

$$\frac{\boldsymbol{u}^j_{k+1} - \boldsymbol{u}^{[n]}}{\Delta t} = a^{\text{im}}_{j,j}\left(\nu \nabla^2 \boldsymbol{u}^j_{k+1} - \nabla p^j_k + \boldsymbol{f}^j\right) + \sum_{i=0}^{j-1} a^{\text{im}}_{j,i} \boldsymbol{F}^{\text{im}}_i + \sum_{i=0}^{j-1} a^{\text{ex}}_{j,i} \boldsymbol{F}^{\text{ex}}_i$$

$$\nabla^2 q_{k+1} = \frac{1}{a^{\text{im}}_{j,j} \Delta t} \nabla \cdot \boldsymbol{u}^j_{k+1}$$

$$p^j_{k+1} = p^j_k + q_{k+1} - \nu \nabla \cdot \boldsymbol{u}^j_{k+1}$$

   (c) $\boldsymbol{u}^j = \boldsymbol{u}^j_{k_j} - (a^{\text{im}}_{j,j} \Delta t) \nabla q_{k_j}, \quad p^j = p^j_{k_j}$

3. Perform the recombination step:

$$\frac{\hat{\boldsymbol{u}}^{[n+1]}-\boldsymbol{u}^{[n]}}{\Delta t}=\sum_{j=0}^{s-1}b_j^{\text{im}}\boldsymbol{F}_j^{\text{im}}+\sum_{j=0}^{s-1}b_j^{\text{ex}}\boldsymbol{F}_j^{\text{ex}}$$

$$\nabla^2 q=\frac{1}{b_{s-1}^{\text{im}}\Delta t}\nabla\cdot\hat{\boldsymbol{u}}^{[n+1]}$$

$$p^{[n+1]}=p^{s-1}+q$$

$$\boldsymbol{u}^{[n+1]}=\hat{\boldsymbol{u}}^{[n+1]}-(b_{s-1}^{\text{im}}\Delta t)\nabla q.$$

Output: $\boldsymbol{u}^{[n+1]}$ and $p^{[n+1]}$.

Note that as the notation implies, the number of iterations within each RK stage needs not be the same, which is the case when a tolerance instead of a prescribed number of iterations is set.

5.3. General fully-implicit schemes

Although in this paper we focus on the iterated pressure correction with IMEX schemes, which treat the nonlinear advection term explicitly and thus enable a rigorous analysis of the convergence and the cost, the idea of iterating a pressure-correction scheme also applies to fully-implicit schemes. This is similar to how SIMPLE is applied to fully-implicit schemes for the Navier–Stokes equations. Indeed, we show next that, in the fully-implicit case, when compared to the single-step pressure-correction schemes and to the fully-coupled schemes, it is very advantageous to iterate the pressure correction.

First, the aggregate scheme obtained by regular single-step pressure correction, even with the rotational correction, will not recover the fully-implicit scheme exactly, but the iterated version removes this inconsistency. The fully-implicit counterpart of (13) will have (13a) replaced by

$$\left(\frac{a}{\Delta t}-\nu\nabla^2+\boldsymbol{u}_*^{n+1}\cdot\nabla\right)\boldsymbol{u}_*^{n+1}+\nabla p^n=\boldsymbol{f}_{\text{rhs}}^{n+1,n},\tag{61}$$

where $\boldsymbol{f}_{\text{rhs}}^{n+1,n}$ is the fully-implicit counterpart. The resultant aggregate scheme with the rotational correction thus becomes

$$\left(\frac{a}{\Delta t}-\nu\nabla^2\right)\boldsymbol{u}^{n+1}+\boldsymbol{u}_*^{n+1}\cdot\nabla\boldsymbol{u}_*^{n+1}+\nabla p^{n+1}=\boldsymbol{f}_{\text{rhs}}^{n+1,n},\tag{62}$$

which is almost the same as (6a), except that the advection term is $\boldsymbol{u}_*^{n+1}\cdot\nabla\boldsymbol{u}_*^{n+1}$ instead of $\boldsymbol{u}^{n+1}\cdot\nabla\boldsymbol{u}^{n+1}$. Although $\boldsymbol{u}_*^{n+1}$ and $\boldsymbol{u}^{n+1}$ are equivalent in terms of accuracy, this scheme discrepancy is not ideal conceptually. If the pressure correction is iterated, the algorithm becomes Algorithm 1 with the prediction step replaced by

$$\left(\frac{a}{\Delta t}-\nu\nabla^2+\boldsymbol{u}_{k+1}^{n+1}\cdot\nabla\right)\boldsymbol{u}_{k+1}^{n+1}+\nabla p_k^{n+1}=\boldsymbol{f}_{\text{rhs}}^{n+1,n}.\tag{63}$$

Upon convergence, the predicted velocity coincides with the corrected velocity, which removes the inconsistency in the aggregate scheme and recovers the fully-implicit scheme exactly.

However, to solve the nonlinear equation (63), we need some iteration to deal with the advection term. As was mentioned at the end of Sec. 2, the iteration for the nonlinear term can be efficiently combined with the pressure-correction iteration, i.e. with (63) replaced by either of the following three options:

$$\left(\frac{a}{\Delta t}-\nu\nabla^2+\boldsymbol{u}_k^{n+1}\cdot\nabla+(\nabla\boldsymbol{u}_k^{n+1})^{\mathrm{T}}\cdot\right)\boldsymbol{u}_{k+1}^{n+1}+\nabla p_k^{n+1}=\boldsymbol{f}_{\text{rhs,NR}}^{n+1},\tag{64a}$$

$$\left(\frac{a}{\Delta t}-\nu\nabla^2+\boldsymbol{u}_k^{n+1}\cdot\nabla\right)\boldsymbol{u}_{k+1}^{n+1}+\nabla p_k^{n+1}=\boldsymbol{f}_{\text{rhs}}^{n+1,n},\tag{64b}$$

$$\left(\frac{a}{\Delta t}-\nu\nabla^2\right)\boldsymbol{u}_{k+1}^{n+1}+\boldsymbol{u}_k^{n+1}\cdot\nabla\boldsymbol{u}_k^{n+1}+\nabla p_k^{n+1}=\boldsymbol{f}_{\text{rhs}}^{n+1,n},\tag{64c}$$

which are analogous to (7), (8) and (9), respectively. The initial guess can be $\boldsymbol{u}_0^{n+1}=\boldsymbol{u}^n$. The above novel iterated modifications (64) will not change the converging point but they allow the use of new fully-implicit iterated schemes at almost the same cost per iteration as that incurred by their IMEX counterpart.

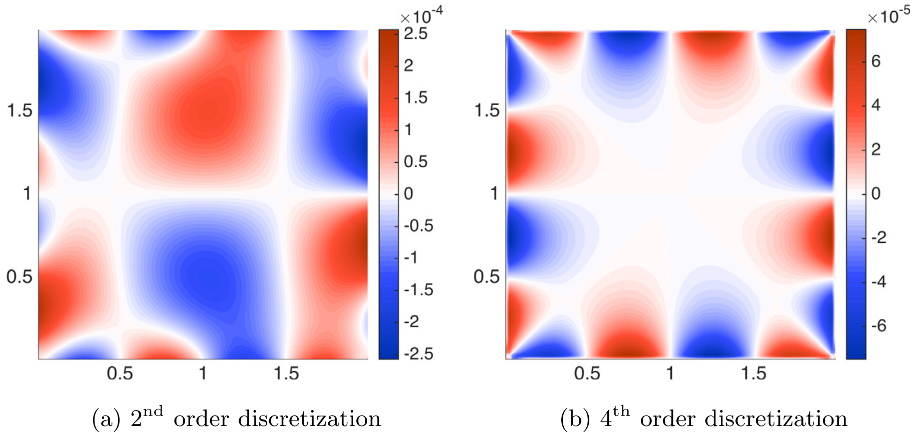


Fig. 3. Spatial error shapes obtained with discretizations of different orders.

6. Numerical results

We now present numerical examples to support the theoretical results and illustrate the concepts and algorithms obtained. We use a finite volume method on a staggered Arakawa C-grid (a.k.a. marker-and-cell grid) that is uniform and Cartesian for the spatial discretization of the Stokes and Navier–Stokes equations. We use the 4th order central difference scheme for the diffusion term and the QUICK scheme [67] for the advection term. Further details on the numerical schemes and their implementation are provided in [61].

Next, unless otherwise noted, we always use the rotational form of the pressure correction schemes. For the Stokes equations, we always use fully-implicit time-marching schemes, while for Navier–Stokes, we use IMEX time-marching schemes that treat the advection term explicitly and the others implicitly.

6.1. Guermond and Shen test case

We start with a widely-used analytic test case originally proposed by [25] for Stokes and later extended to Navier–Stokes [71,13,62,16]. Figs. 1 and 2 shown previously were based on this test case.

Denote the two-dimensional velocity by $\boldsymbol{u} = (u, v)$. The analytical expressions for this manufactured solution are

$$\begin{aligned} u &= \pi \sin(t)\sin(2\pi y)\sin^2(\pi x), \\ v &= -\pi \sin(t)\sin(2\pi x)\sin^2(\pi y), \\ p &= \sin(t)\cos(\pi x)\sin(\pi y), \end{aligned} \tag{65}$$

which has periods of 2 in space and of 2π in time. The viscosity is set to $\nu = 1$, which corresponds to a Reynolds number of order 1 to 10. We specify a no-slip boundary condition $u = v = 0$ at the four boundaries of the solution domain $[0, 2] \times [0, 2]$. The corresponding boundary condition for the auxiliary variable $\tilde{s}$ in pressure correction is thus $\partial\tilde{s}/\partial\boldsymbol{n} = 0$. We set the initial time as $t_0 = 0$ and run all the simulations up to $t = 1$ (away from a null solution), when we compare the numerical solution to the analytic one and compute the errors.

To investigate the temporal convergence order, we ensure that the spatial resolution is fine enough to have the spatial errors dominated by temporal errors due to both the time-marching scheme and pressure correction scheme. Here we use a mesh resolution of $\Delta x = \Delta y = 0.01$ (i.e. a 200×200 mesh). Note that in some cases, the spatial error field can look deceivingly similar to the temporal one. For example, the spatial error due to a 4th order spatial discretization plotted in Fig. 3b has a similar pattern as the temporal error shown in Fig. 1b, while Fig. 3a illustrates that the pattern of the spatial error due to a 2nd order spatial discretization is closer to that of the solution shown in Fig. 1a.

6.1.1. Reducing time step size vs. increasing number of iterations

An objective of this study is to illustrate whether it is more efficient to perform more iterations or to merely reduce the time step without iterating the pressure correction scheme. Therefore, we compare and contrast the use of different time step size Δt and different number k of pressure-correction iterations per time step. All examples integrate the Stokes equations with the BDF3 scheme. For testing purpose, we fix k for each time step in a simulation instead of using a stopping criterion introduced in Sec. 3.3. We let k vary across 1, 2, 4, 8 and 16 and for each k, we plot in Fig. 4 how the errors for pressure and velocity in L^2 and L^∞ norms vary as Δt is reduced. Note that we use the total number of pressure correction iterations $k \cdot T/\Delta t$ as abscissas, so that points on the same vertical line correspond to the same computational cost. This makes it straightforward to compare efficiency because for two points on the same vertical line, the lower point achieves a

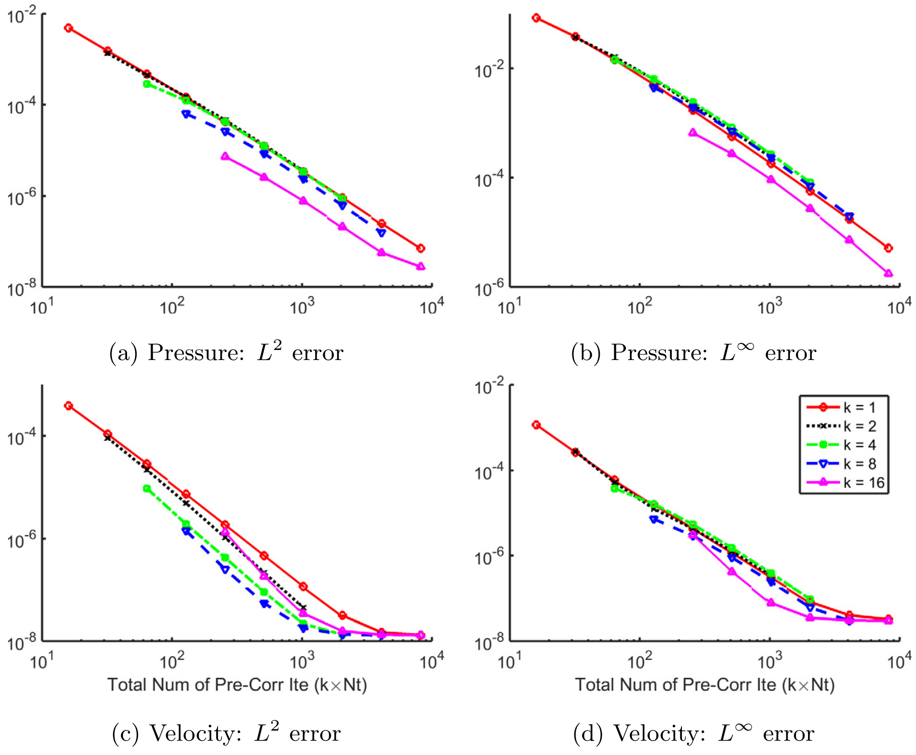


Fig. 4. Errors for pressure and velocity are plotted against the total number of pressure correction iterations $k \cdot T/\Delta t$ for different values of k and Δt. For each curve, the number of pressure correction iterations n within one time step is prescribed and fixed, while the time step size Δt varies. The first point of each curve corresponds to $\Delta t = .0625$, which is why curves for different k starts at different abscissas. These results are from integrating the Stokes equations in time with the BDF3 scheme.

smaller error with the same computational cost and thus has a higher efficiency. The first point of each curve corresponds to the same $\Delta t = .0625$ and that is why curves for different k start at different abscissas.

As we can see in Fig. 4, more than half of the curves of $k > 1$, especially those of larger k, lie below the curve of $k = 1$ (red curves) corresponding to the classical pressure-correction scheme. This implies that when enough iterations are performed (e.g. $k = 16$), the iterated pressure-correction scheme offers better accuracy for the same cost, although the threshold of k above which iterating becomes more efficient is found to differ across variables and norms. For example, for the velocity errors in L^2 norm (Fig. 4c), doing iterations always improves efficiency, whereas for the pressure errors in L^∞ norm (Fig. 4b), only the $k = 16$ curve indicates a better efficiency.

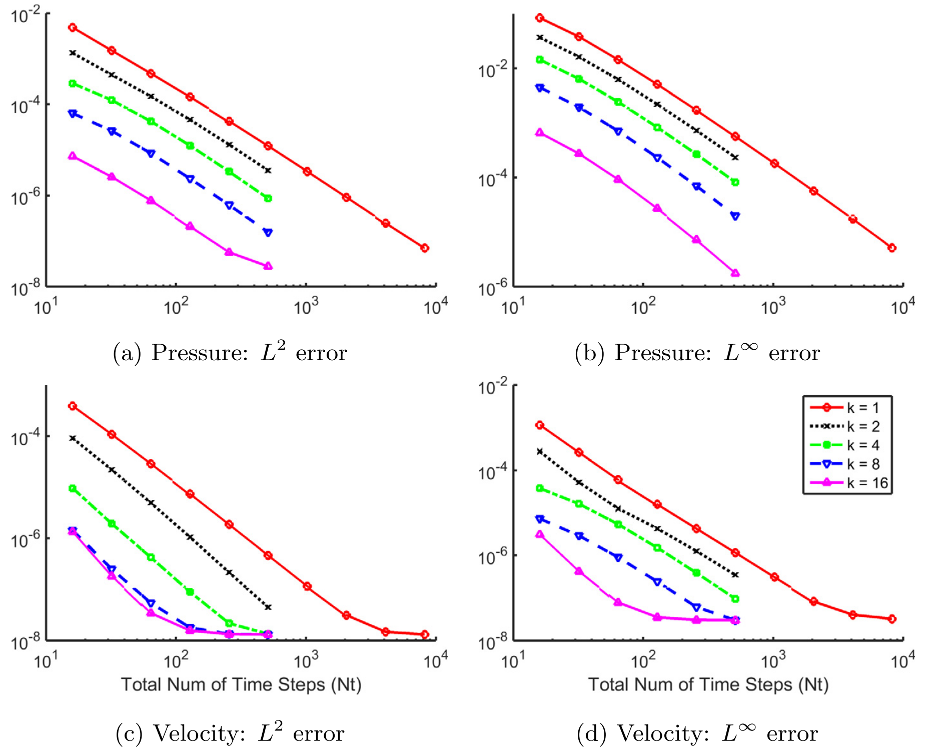
We note that the "$k > 1$ curves" lying above the red curves should not be misinterpreted as "iterating the pressure-correction scheme can increase the errors". For this type of comparison, we should compare points corresponding to the same Δt. If we scale the abscissas of the green curves by $1/k$ and shift them to the left as shown in Fig. 5, we will find all the "$k > 1$ curves" lying below the red ones. Therefore, iterating can only decrease the errors, as predicted by our convergence analysis in Sec. 3.2. We also find that typically the velocity first converges to the solution of the fully-coupled system (see where curves with different k clutter) and when Δt is reduced, the velocity first hits the spatial errors and saturates. Both of these findings result from a lower order of convergence of the splitting error (i.e. $\tilde{s}$) for pressure than for velocity.

6.1.2. Recovery of temporal convergence order of time-marching schemes

If enough iterations are performed so that the splitting error becomes negligible compared to the time-integration error, then only the latter is observed and we would expect the apparent temporal convergence order would match the order of the time-integration scheme. Fig. 6 and 7 show that this is indeed the case. We can see the apparent temporal convergence order changing from $\tilde{s}$ to s when enough number of iterations are performed in the Stokes and Navier–Stokes case, respectively.

6.1.3. Aitken acceleration

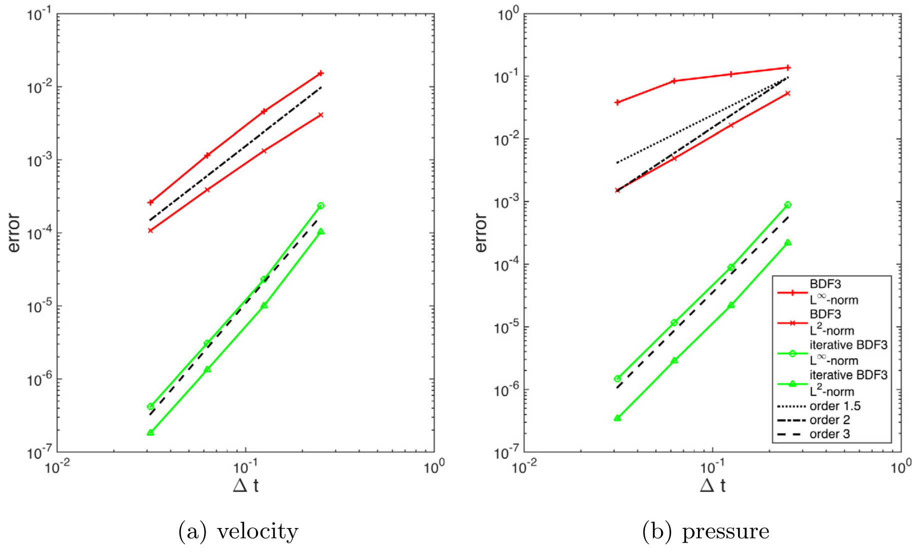
The Aitken relaxation outlined in Sec. 3.4 makes the splitting error converge faster to zero in the pressure-correction iteration. To compute the splitting errors only, we need to use the solution of the fully-coupled scheme rather than the analytic solution as the reference. As shown in Fig. 8, in this test, compared to the iteration without the Aitken relaxation, the iteration with the relaxation reaches an error of 10^{-10} in L^∞ norm and an error of 10^{-12} in L^2 norm, with approximately only 1/3 of the total number of the iterations.



(a) Pressure: L^2 error (b) Pressure: L^∞ error

(c) Velocity: L^2 error (d) Velocity: L^∞ error

Fig. 5. Errors for pressure and velocity are plotted against the total number of time steps $N_t = T/\Delta t$ for different values of k and Δt. Except for the abscissa, everything is as in Fig. 4.



(a) velocity (b) pressure

Fig. 6. Stokes problem. The order of convergence of the time-integration scheme is recovered when enough iterations are performed to eliminate the splitting error.

6.1.4. Effects of the rotational correction

As our analysis in Sec. 4.2 predicts, Fig. 9 shows that both the pressure-correction iterations with and without the rotational correction enjoy linear convergence to the solution of the fully-coupled system, but the convergence of the former is much faster. We can actually theoretically quantify the convergence rates in this case and compare them to the empirically measured values.

As [17] points out, the constant β in the inf-sup condition for a square domain, however elementary it may seem, is surprisingly still unknown. The numerical evidence supports the conjecture that $\beta^2 = 1/2 - 1/\pi \approx 0.18$, so (23) predicts,

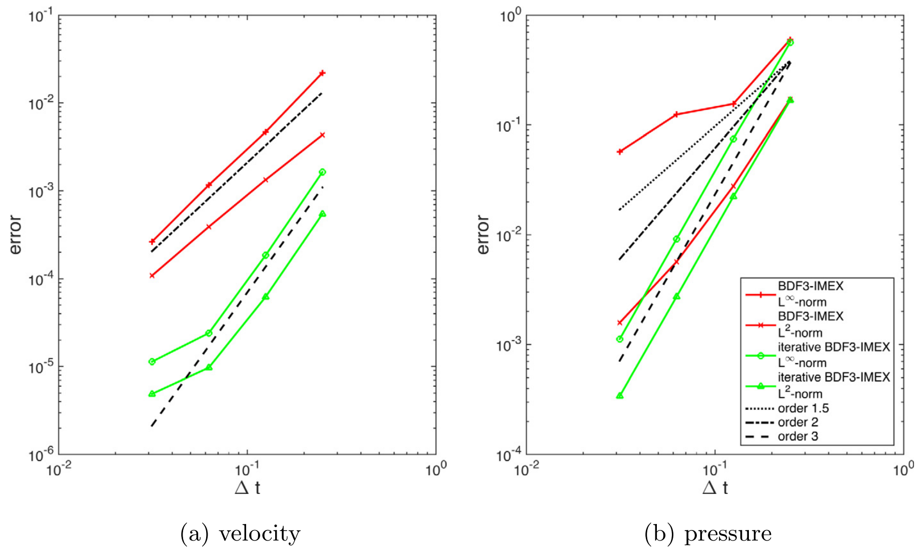


Fig. 7. Navier–Stokes problem. The order of convergence of the time-integration scheme s is recovered when enough iterations are performed to eliminate the splitting error.

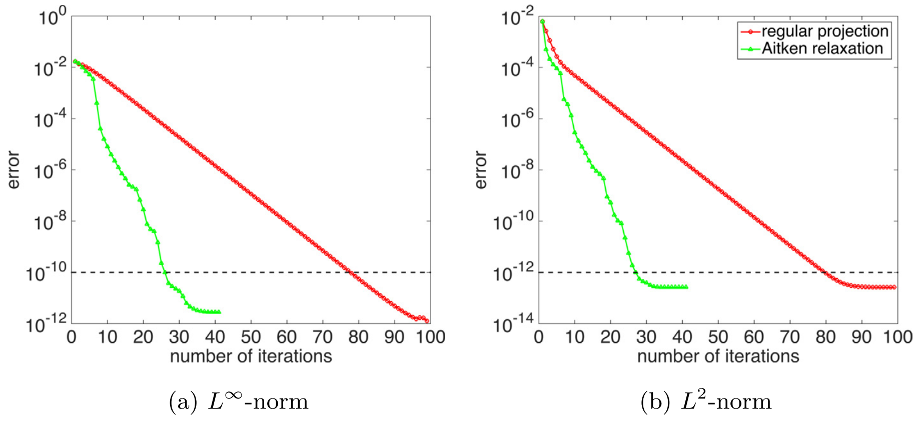


Fig. 8. Convergence of the iterated pressure-correction scheme with and without Aitken acceleration. A BDF2 with time step size $\Delta t = 0.1$ is used to integrate the Stokes equations. The errors are evaluated at time $T = 0.1$ and are computed against the solution to the fully-coupled scheme, so all the errors here are splitting errors.

$\rho(\boldsymbol{M}_{\text{rot}}) \approx 1 - \beta^2 \approx 0.82$. What we measure from Fig. 9 is a convergence rate of $R \approx 0.78$ (both norms give similar numbers), which agrees well with our theory. For the non-rotational case, since $\nu = 1$, $\Delta t = 0.1$, and $h = 0.01$, we have $\nu\Delta t/h^2 = 10^3$ and (24) thus predicts, $\rho(\boldsymbol{M}_{\text{non}}) \approx 0.999$. The measured value is $R \approx 0.99$, which again agrees well with our theory. This verifies that when $\nu\Delta t/h^2$ is large, as in most practical cases, the rotational correction makes a big difference in accuracy at a negligible cost.

6.2. Modified Colomés and Badia test case

As was shown in Fig. 4c and 4d, when the time step size is small enough, the temporal errors in velocity are dominated by the spatial errors and hence the curve levels off when the time step size is further reduced. Therefore, if spatial errors are large, the range of Δt in which we can observe the temporal convergence order is small, such as in Fig. 7a. This issue is aggravated for Navier–Stokes with a moderate to large Péclet number $\text{Pe} = u\Delta x/\nu$ (a.k.a. grid Reynolds number) and the advection term is treated explicitly, because a stability constraint of the form $\Delta t \leq C\Delta x/U_0$ will apply, where U_0 is a characteristic velocity of the flow and usually $C \leq 1$ for an IMEX scheme. Hence, only Δt smaller than the stability threshold and larger than the threshold for spatial error dominance can be used to test the temporal convergence order. Refining the mesh does us not help in this case because although it reduces the lower threshold of Δt for spatial error dominance, it also reduces the upper threshold for stability. Moreover, a spatial resolution that is not high enough could also cause a false higher convergence order in time to be observed [24].

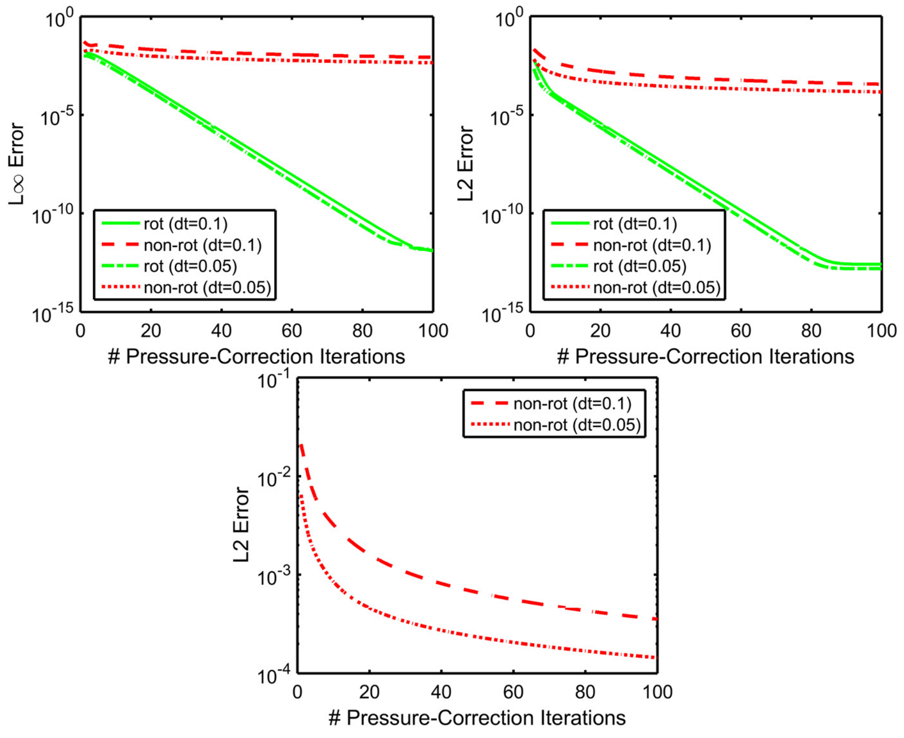


Fig. 9. Convergence of the pressure-correction iteration with and without rotational correction. A BDF2 scheme is used with $\Delta t = 0.1$ and 0.05 to integrate the Stokes equations. Errors are evaluated at time $T = 0.1$ and computed against the solution to the fully-coupled system. The top left panel shows the L_∞ errors and the top right shows the L_2 errors. The bottom is a zoom-in to the two red curves for the non-rotational scheme in the top right panel.

To make the spatial error negligible, one can use a manufactured solution that varies in space as a low order polynomial. In fact, this can even eliminate all the spatial errors because finite difference/volume and finite element (with a polynomial basis) approximations can recover the derivative exactly when the true solution is a polynomial of low enough order. For example, [16] proposed a manufactured solution that is linear in space. However, their pressure did not depend on time at all, which eliminates the splitting error when a projection method is used because the pressure guess is always exact. We thus modify the solution for the pressure to include some temporal dependence as below:

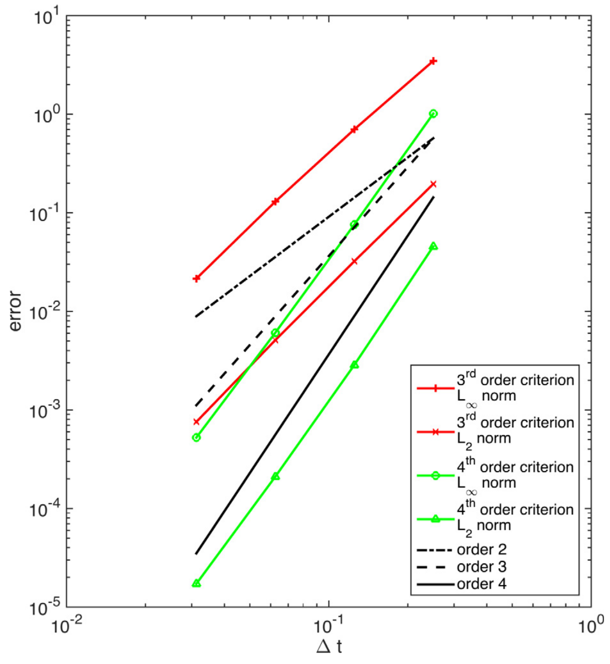
$$\begin{aligned} u &= x \sin\left(\frac{\pi t}{10}\right) \exp\left(\frac{t}{25}\right), \\ v &= -y \sin\left(\frac{\pi t}{10}\right) \exp\left(\frac{t}{25}\right), \\ p &= 1000(x+y) \exp\left(\frac{t}{25}\right). \end{aligned} \tag{66}$$

The solution domain is the unit square $[0, 1] \times [0, 1]$ and we simulate the flow from $t = 0$ to $t = 0.1$. The viscosity is $\nu = 1$ (Reynolds number of order 1). A downside of using a polynomial solution is that the solution is not periodic and some of the boundary condition values usually vary in space and time. Here we specify Dirichlet boundary conditions for both velocity components at all four boundaries with the boundary values agreeing with the analytic solution.

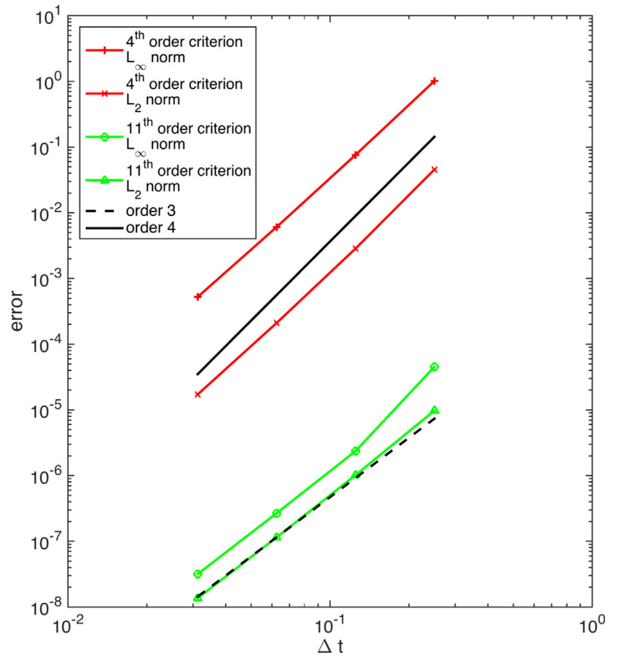
6.2.1. Stopping criterion

First, we test numerically whether the stopping criterion (41) proposed in Sec. 3.3 controls the remaining splitting error as intended. We focus on the splitting error in pressure, which is of order $O(\Delta t^{1.5})$ when only one pressure-correction iteration is applied, i.e. $\tilde{s} = 1.5$. We use the stopping criterion with $C_2 = 1$ and different $\tilde{s}_*$, and refer to it as the $\tilde{s}_*$th order criterion.

The results for the Stokes equations with a BDF3 time-marching scheme and stopping criteria of order $\tilde{s}_* = 3$, 4 and 11 are shown in Fig. 10. With the range of Δt used here, the splitting error of the classic pressure-correction scheme dominates the time-marching scheme error, so if the stopping criterion is loose enough, the remaining splitting error could remain dominant and the apparent temporal convergence order should then be the convergence order of the remaining splitting error, which is $\tilde{s}_*$ as the stopping criterion (41) implies. This is indeed the case. In Fig. 10a, when a 4th order criterion is used, we observe an apparent convergence order $\tilde{s}_* = 4$, which is even higher than $s = 3$, the order of the time-marching scheme. However, if the stopping criterion is stringent enough to make the remaining splitting error dominated by the



(a) controlled convergence



(b) splitting error elimination

Fig. 10. Temporal convergence of pressure for stopping criteria (41) with different $\tilde{s}_*$. The BDF3 scheme is used to integrate the Stokes equations in time.

time-marching scheme error, the apparent temporal convergence order should recover to the order of the time-marching scheme s and should thus be independent of the $\tilde{s}_*$ used. This is verified in Fig. 10b, where we can see that when $\tilde{s}_* = 11$ is used, we observe an apparent temporal convergence order 3 and the errors are much smaller than those obtained with $\tilde{s}_* = 4$.

The above results remind us that when we use a stopping criterion of order equal to the order of the time-marching scheme, it only guarantees us to recover the temporal convergence order of the time-marching scheme. It does not guarantee that the remaining splitting error is dominated by the time-marching scheme error, because this has to do with the absolute error magnitude, but $\tilde{s}_*$ only governs the convergence order in time. With the same $\tilde{s}_*$, we can use a smaller C_2 to drive down the magnitude of the remaining splitting error.

Fig. 11 shows how the number of iterations required to meet a stopping criterion of a particular order $\tilde{s}_*$ within one time step grows as the time step size Δt is reduced. The horizontal axis is in logarithmic scale, so the initial straight lines for large Δt in both plots verify the logarithmic growth indicated by (29). As expected, a higher order criterion requires more iterations when $\Delta t < 1$. However, for small Δt, the number of iteration begins to level off and even decrease. This indicates that the estimate of (29) is actually pessimistic in this case. This behavior is actually desirable in practice because when Δt becomes small, the spatial error will be increasingly dominant and at some point, driving the remaining errors down to the spatial errors is already sufficient. In the extreme case where the spatial errors dominate the initial splitting error, there is no point iterating the pressure correction at all.

6.2.2. Runge–Kutta schemes

This section shows results of the Runge–Kutta schemes applied to both Stokes and Navier–Stokes. For the Stokes equations, the BDF3 scheme and the A-stable third order DIRK (diagonally implicity Runge–Kutta) scheme [1] are compared in Fig. 12. For the Navier–Stokes equations, the BDF3-IMEX scheme and the third order four-stage L-stable IMEX-RK scheme [31] are compared in Fig. 13. In both cases, we find that iterating the pressure correction schemes boosts the apparent temporal convergence order to the order of the time-marching schemes.

7. Summary and conclusions

In this work, we approached the incremental pressure-correction as a Richardson iterative scheme for the pressure–Schur complement equation, where the preconditioner is closely related to the discrete scalar Laplacian. For classical projection schemes, only one iteration is performed. For a given spatial discretization, the time step size can be reduced to increase accuracy. Alternatively, the time step size can be fixed and more iterations of the projection scheme performed. We developed,

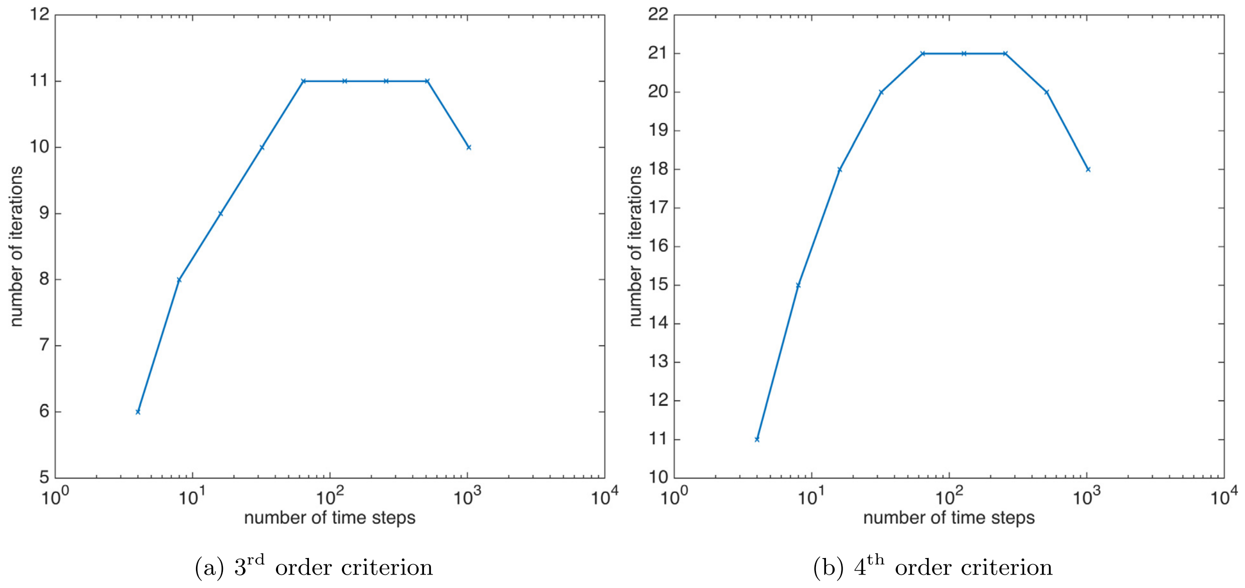


Fig. 11. Number of iterations required to meet a stopping criterion (41) of a particular order $\tilde{s}_*$ within one time step for different time step sizes.

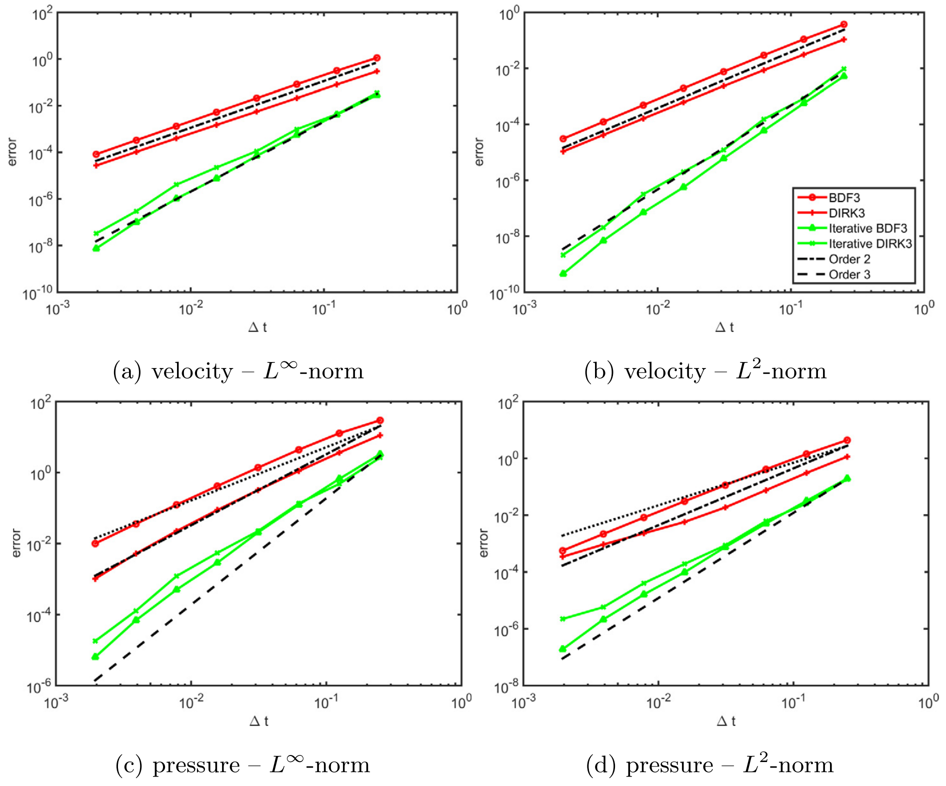
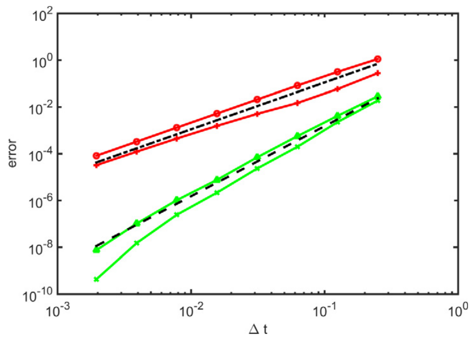


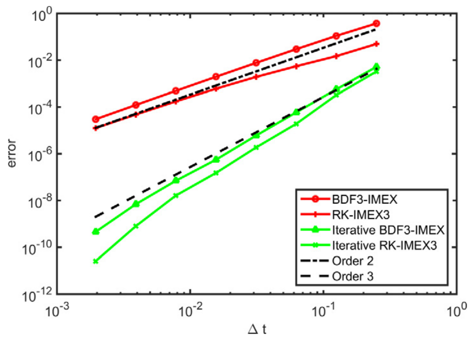
Fig. 12. Stokes problem with 3rd order stopping criterion (41).

analyzed, and exemplified the computational properties and benefits of performing iterations as opposed to merely reducing the time step size. The main result is the iterated pressure-correction projection methods for the unsteady incompressible Stokes and Navier–Stokes equations.

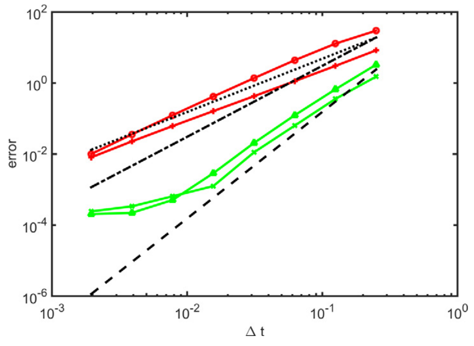
The accuracy and convergence properties of the iterated pressure-correction schemes were obtained and analyzed. We showed that such iterations can be more efficient computationally than merely reducing the time step size. We observed



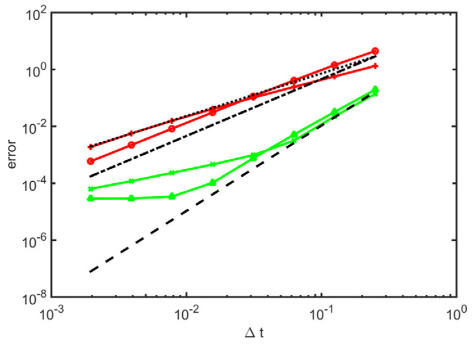
(a) velocity – L^∞-norm



(b) velocity – L^2-norm



(c) pressure – L^∞-norm



(d) pressure – L^2-norm

Fig. 13. Navier–Stokes problem with 3rd order stopping criterion (41).

that they can remove the splitting error due to projection methods: they thus eliminate the artificial boundary layer issue and recover the temporal order of convergence of the time-marching scheme in both velocity and pressure. We confirmed the benefit of the rotational correction, in our case in terms of convergence speed of the iteration. We obtained stopping criteria for controlling the temporal convergence order and the magnitude of the splitting error. We identified a potential super-convergence phenomenon caused by this stopping criterion and analyzed its underlying mechanism. To further reduce computational costs, we showed how the method can be accelerated using an Aitken relaxation scheme. We also explained how the iterated pressure correction schemes can be conveniently embedded into popular IMEX time-marching schemes, including linear multi-step schemes and multi-stage schemes such as Runge–Kutta. We also discussed how the application of such iterations to fully-implicit schemes would be beneficial.

Numerical results were provided to support the theoretical advances, including manufactured test cases for the Stokes and Navier–Stokes equations. We showed that as more iterations are performed, the spurious numerical boundary layer that characterizes projection methods vanishes. We illustrated that if enough iterations are performed, the error becomes virtually identical to that of a fully-coupled scheme, which solves for velocity and pressure simultaneously in one linear system. It is then possible to observe the intended temporal convergence order for both IMEX linear multi-step and IMEX Runge–Kutta time-integration schemes. We also showed cases where spatial errors can be mistaken for temporal errors: a high enough spatial resolution must be used in order to obtain sensible results. Finally, we illustrated the use of the time-step-size-dependent stopping criterion and Aitken acceleration scheme. When compared to the solutions of classical projection methods, computational costs were reduced by one or more orders of magnitude for the same level of accuracy.

Presently, we evaluated iterated pressure-correction schemes for a finite volume spatial discretization. Future work include testing the schemes for other spatial discretizations. For example, good candidates are the hybrid discontinuous Galerkin (HDG) schemes for ocean modeling with projection methods and IMEX Runge–Kutta schemes [62]. Moreover, the iterated pressure-correction schemes can potentially accelerate stochastic flow simulation as well. Examples include uncertainty quantification for fluid flows with polynomial chaos expansions [33] and stochastic Navier–Stokes solvers using dynamically orthogonal equations [63]. Accuracy of the base Navier–Stokes solver for stochastic flow simulation is essential because numerical errors can become spurious uncertainty modes and contaminate the statistics of interest. More accurate flow simulations will also benefit disciplines downstream of computational fluid dynamics, i.e. disciplines that take simulated flow data as input, such as path planning [38,55,54,35], Lagrangian transport [20], data assimilation [53], adaptive control of fluid flows [22], and aerospace applications [68].

Acknowledgements

We thank the MSEAS group members for useful discussions. We thank the Office of Naval Research (ONR) for research support under grants N00014-11-1-0701 (MURI-IODA), N00014-12-1-0944 (ONR6.2), and N00014-14-1-0476 (Science of Autonomy – LEARNS), and the National Oceanographic Partnership Program (NOPP) for research support under grant N00014-15-1-2597 (Seamless Multiscale Forecasting), all to MIT. We are also grateful to the Technology and Contracts Office at MIT Lincoln Laboratory.

Appendix A. The rotational correction

As mentioned previously [57,25,42,62], the rotational correction serves two important purposes. We reiterate them here in an intuitive fashion.

First, it avoids a nonphysical locking of the normal derivative of pressure at the boundary. On a boundary where the normal velocity is specified, this boundary condition is imposed on $\boldsymbol{u}_*^{n+1}$ in (13a). To ensure that $\boldsymbol{u}^{n+1}$ obtained in (13d) satisfies this same boundary condition, one obtains $\partial q/\partial \boldsymbol{n} = 0$, which is fed to (13b) as the boundary condition for q. However, without the rotational correction, this would imply through (13c) that $\partial p^{n+1}/\partial \boldsymbol{n} = \partial p^n/\partial \boldsymbol{n}$ and hence, $\partial p^n/\partial \boldsymbol{n} \equiv \partial p^0/\partial \boldsymbol{n}$ for all n, which is not physical and yields a numerical artifact. Although the rotational correction unlocks the pressure normal derivative, this itself does not ensure the superiority of the scheme with the correction, because changing a quantity by a wrong amount may not be better than keeping it fixed.

The second purpose of the rotational correction is to address this and render the aggregate scheme (13) consistent with the fully-coupled scheme (4), up to the tangential velocity boundary conditions. Here the aggregate scheme refers to the equations obtained by manipulating (13) to eliminate the intermediate variables $\boldsymbol{u}_*^{n+1}$ and q, and thus aggregate the pressure correction effects. Ideally, the aggregate scheme should coincide with the fully-coupled scheme (4) because the pressure correction is simply an algorithm for computing (4) more efficiently. Since (13b) and (13d) together guarantee $\nabla \cdot \boldsymbol{u}^{n+1} = 0$, we just need to check whether the aggregate scheme for the momentum equation agrees with (4a), which is an exact application of an IMEX scheme to momentum.

Substituting (13c) and (13d) into $(a/\Delta t)\boldsymbol{u}_*^{n+1}$ and eliminating q, we have

$$\begin{aligned}\frac{a}{\Delta t}\boldsymbol{u}_*^{n+1} &= \frac{a}{\Delta t}\left(\boldsymbol{u}^{n+1} + \frac{\Delta t}{a}\nabla\left(p^{n+1} - p^n + \nu\nabla\cdot\boldsymbol{u}_*^{n+1}\right)\right)\\ &= \frac{a}{\Delta t}\boldsymbol{u}^{n+1} + \nabla\left(p^{n+1} - p^n\right)\underbrace{+\,\nu\nabla(\nabla\cdot\boldsymbol{u}_*^{n+1})}_{\text{Rotational corr}}.\end{aligned}$$

Inserting this back to (13a), with some terms canceling each other, we obtain the aggregate scheme

$$\frac{a}{\Delta t}\boldsymbol{u}^{n+1} - \nu\nabla^2\boldsymbol{u}_*^{n+1}\underbrace{+\,\nu\nabla(\nabla\cdot\boldsymbol{u}_*^{n+1})}_{\text{Rotational corr}} + \nabla p^{n+1} = \boldsymbol{f}_{\text{rhs}}^{n+1,n}. \tag{A.1}$$

Without the rotational correction, (A.1) is almost (4a), except that the diffusion term is evaluated with the intermediate result $\boldsymbol{u}_*^{n+1}$ rather than the end result $\boldsymbol{u}^{n+1}$. However, with the rotational correction in (A.1), (4a) is recovered. To show this, we express the inconsistency as,

$$\begin{aligned}\nu\nabla^2\boldsymbol{u}^{n+1} - \nu\nabla^2\boldsymbol{u}_*^{n+1} &= -\nabla\times\nabla\times(\boldsymbol{u}^{n+1} - \boldsymbol{u}_*^{n+1}) - \nu\nabla(\nabla\cdot\boldsymbol{u}_*^{n+1})\\ &= -\nu\nabla(\nabla\cdot\boldsymbol{u}_*^{n+1}),\end{aligned}$$

where $\nabla \cdot \boldsymbol{u}^{n+1} = 0$ and the vector identity $\nabla^2\boldsymbol{u} = \nabla(\nabla\cdot\boldsymbol{u}) - \nabla\times\nabla\times\boldsymbol{u}$ were used. The final equality is due to (13d), which renders the velocity difference a gradient field and thus irrotational. This shows that the rotational correction exactly makes up for the inconsistency and hence enables (4a) to be recovered.

Appendix B. Connection between pressure correction projection methods and the pressure-Schur complement

In Sec. 3, we show how a pressure correction scheme is equivalent to a Richardson iteration of the pressure-Schur complement linear system, but we omit the derivation of its matrix form. Here we provide this derivation for the case of a general IMEX linear multi-step scheme and show the equivalence in general. Since each stage of a IMEX-RK scheme is analogous to an IMEX linear multi-step scheme, our derivation also applies to general IMEX-RK schemes.

Let us start by writing down the fully-coupled scheme for

$$\begin{cases}\dfrac{\partial\boldsymbol{u}}{\partial t} = \nu\nabla^2\boldsymbol{u} - \nabla p + \boldsymbol{f} - \boldsymbol{u}\cdot\nabla\boldsymbol{u} & \text{(a)}\\ -\nabla\cdot\boldsymbol{u} = 0 & \text{(b)}\end{cases} \tag{B.1}$$

when a general IMEX linear multi-step scheme is applied:

$$\begin{cases} \dfrac{a_{-1}\boldsymbol{u}^{n+1}+\sum_{j=0}^{s-1}a_j\boldsymbol{u}^{n-j}}{\Delta t} = c_{-1}\left(\nu\nabla^2\boldsymbol{u}^{n+1}-\nabla p^{n+1}+\boldsymbol{f}^{n+1}\right)+\sum_{j=0}^{s-1}c_j\boldsymbol{F}^{\mathrm{im}}_{n-j}+\sum_{j=0}^{s-1}b_j\boldsymbol{F}^{\mathrm{ex}}_{n-j}, & \text{(a)} \\ \nabla\cdot\boldsymbol{u}^{n+1}=0, & \text{(b)} \end{cases} \tag{B.2}$$

where

$$\boldsymbol{F}^{\mathrm{im}}_{n-j}=\nu\nabla^2\boldsymbol{u}^{n-j}-\nabla p^{n-j}+\boldsymbol{f}^{n-j} \quad \text{and} \quad \boldsymbol{F}^{\mathrm{ex}}_{n-j}=-\boldsymbol{u}^{n-j}\cdot\nabla\boldsymbol{u}^{n-j}.$$

After spatial discretization (and (B.2a) being divided by c_{-1}), we obtain the fully-coupled linear system

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{G} \\ -\boldsymbol{D} & \boldsymbol{0} \end{bmatrix}\begin{bmatrix} \boldsymbol{u}^{n+1} \\ p^{n+1} \end{bmatrix}=\begin{bmatrix} \boldsymbol{f}^{n+1,n}_{\mathrm{rhs}}+\mathrm{bc}^{n+1}_{\mathrm{mom}} \\ \mathrm{bc}^{n+1}_{\mathrm{con}} \end{bmatrix}, \tag{B.3}$$

where $\boldsymbol{A}=(a_{-1}/(c_{-1}\Delta t))\boldsymbol{I}-\nu\boldsymbol{L}_{\boldsymbol{u}}$. Note that we use the same notation $\boldsymbol{u}$ and p for both the continuum (a spatial field) and the discrete (a finite-length vector) state variables. The precise meaning should be clear from the context. Here $\boldsymbol{D}$ is the discrete divergence, $\boldsymbol{G}=-\boldsymbol{D}^{\mathrm{T}}$ the discrete gradient, and $\boldsymbol{L}_{\boldsymbol{u}}$ the discrete vector Laplacian. More precisely,

$$\nabla\cdot\boldsymbol{u}^{n-j}\approx\boldsymbol{D}\boldsymbol{u}^{n-j}+\mathrm{bc}^{n-j}_{\mathrm{con}}, \quad \nabla p^{n-j}\approx\boldsymbol{G}p^{n-j}, \quad \nabla^2\boldsymbol{u}^{n-j}\approx\boldsymbol{L}_{\boldsymbol{u}}\boldsymbol{u}^{n-j}+\mathrm{bc}^{n-j}_{\mathrm{mom}}.$$

Lastly, $\boldsymbol{f}^{n+1,n}_{\mathrm{rhs}}$ accounts for all the terms in (B.2a) that do not depend on $\boldsymbol{u}^{n+1}$ or p^{n+1}:

$$\boldsymbol{f}^{n+1,n}_{\mathrm{rhs}}=\frac{1}{c_{-1}}\left(-\frac{\sum_{j=0}^{s-1}a_j\boldsymbol{u}^{n-j}}{\Delta t}+c_{-1}\boldsymbol{f}^{n+1}+\sum_{j=0}^{s-1}c_j\boldsymbol{F}^{\mathrm{im}}_{n-j}+\sum_{j=0}^{s-1}b_j\boldsymbol{F}^{\mathrm{ex}}_{n-j}\right),$$

where the $\boldsymbol{F}$'s are discretized accordingly.

Now, let us turn to the counterpart of (B.2) with the pressure-correction scheme in rotational form applied:

$$\frac{a_{-1}\boldsymbol{u}^{n+1}_*+\sum_{j=0}^{s-1}a_j\boldsymbol{u}^{n-j}}{\Delta t}=c_{-1}\left(\nu\nabla^2\boldsymbol{u}^{n+1}_*-\nabla p^n+\boldsymbol{f}^{n+1}\right)+\sum_{j=0}^{s-1}c_j\boldsymbol{F}^{\mathrm{im}}_{n-j}+\sum_{j=0}^{s-1}b_j\boldsymbol{F}^{\mathrm{ex}}_{n-j} \tag{B.4a}$$

$$\nabla^2 q=\frac{a_{-1}}{c_{-1}\Delta t}\nabla\cdot\boldsymbol{u}^{n+1}_* \tag{B.4b}$$

$$p^{n+1}=p^n+q-\nu\nabla\cdot\boldsymbol{u}^{n+1}_* \tag{B.4c}$$

$$\boldsymbol{u}^{n+1}=\boldsymbol{u}^{n+1}_*-\frac{c_{-1}\Delta t}{a_{-1}}\nabla q. \tag{B.4d}$$

With the same spatial discretization as that applied to the fully-coupled scheme, the above can be written in matrix notation:

$$\boldsymbol{A}\boldsymbol{u}^{n+1}_*+\boldsymbol{G}p^n=\boldsymbol{f}^{n+1,n}_{\mathrm{rhs}}+\mathrm{bc}^{n+1}_{\mathrm{mom}} \tag{B.5a}$$

$$\boldsymbol{L}_p q=\frac{a}{\Delta t}(\boldsymbol{D}\boldsymbol{u}^{n+1}_*+\mathrm{bc}^{n+1}_{\mathrm{con}}) \tag{B.5b}$$

$$p^{n+1}=p^n+q-\nu(\boldsymbol{D}\boldsymbol{u}^{n+1}_*+\mathrm{bc}^{n+1}_{\mathrm{con}}) \tag{B.5c}$$

$$\boldsymbol{u}^{n+1}=\boldsymbol{u}^{n+1}_*-\frac{\Delta t}{a}\boldsymbol{G}q, \tag{B.5d}$$

where $a=a_{-1}/c_{-1}$. Here $\boldsymbol{L}_p$ is the discrete scalar Laplacian such that

$$\nabla^2 q\approx\boldsymbol{L}_p q, \qquad \boldsymbol{L}_p=\boldsymbol{D}\boldsymbol{G}=-\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}.$$

Now we can use (B.5b) to obtain $q=(a/\Delta t)\boldsymbol{L}_p^{-1}(\boldsymbol{D}\boldsymbol{u}^{n+1}_*+\mathrm{bc}^{n+1}_{\mathrm{con}})$ and eliminate q in (B.5c) and (B.5d):

$$\boldsymbol{A}\boldsymbol{u}^{n+1}_*+\boldsymbol{G}p^n=\boldsymbol{f}^{n+1,n}_{\mathrm{rhs}}+\mathrm{bc}^{n+1}_{\mathrm{mom}} \tag{B.6a}$$

$$p^{n+1}=p^n+\left(\frac{a}{\Delta t}\boldsymbol{L}_p^{-1}-\nu\boldsymbol{I}\right)(\boldsymbol{D}\boldsymbol{u}^{n+1}_*+\mathrm{bc}^{n+1}_{\mathrm{con}}) \tag{B.6b}$$

$$\boldsymbol{u}^{n+1}=(\boldsymbol{I}-\boldsymbol{G}\boldsymbol{L}_p^{-1}\boldsymbol{D})\boldsymbol{u}^{n+1}_*-\boldsymbol{G}\boldsymbol{L}_p^{-1}\mathrm{bc}^{n+1}_{\mathrm{con}}, \tag{B.6c}$$

which gives us the generalized version of (15). We can further use (B.6a) to obtain $\boldsymbol{u}^{n+1}_*=\boldsymbol{A}^{-1}(-\boldsymbol{G}p^n+\boldsymbol{f}^{n+1,n}_{\mathrm{rhs}}+\mathrm{bc}^{n+1}_{\mathrm{mom}})$ and eliminate $\boldsymbol{u}^{n+1}_*$ in (B.6b):

$$\begin{aligned} p^{n+1} &= p^n + \left(\frac{a}{\Delta t}\boldsymbol{L}_p^{-1} - \nu \boldsymbol{I}\right)(\boldsymbol{D}\boldsymbol{A}^{-1}(-\boldsymbol{G}p^n + \boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1}) + \text{bc}_{\text{con}}^{n+1}) \\ &= p^n - \left(\frac{a}{\Delta t}\boldsymbol{L}_p^{-1} - \nu \boldsymbol{I}\right)(\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}p^n - \boldsymbol{b}^{n+1}), \end{aligned} \tag{B.7}$$

where $\boldsymbol{b}^{n+1} = \boldsymbol{D}\boldsymbol{A}^{-1}(\boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1}) + \text{bc}_{\text{con}}^{n+1}$. This is the generalized version of (14b).

The derivation of the pressure-Schur complement linear system from the fully-coupled system (B.3) is exactly the same as (10) and (11). The linear system can be written as

$$\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}p^n = \boldsymbol{b}^{n+1}. \tag{B.8}$$

Therefore, the pressure update (B.7) due to the pressure-correction scheme can be viewed as one step of a fixed-point iteration scheme for solving (B.8) with the initial guess being p^n. Particularly, such iteration is known as the Richardson iteration and $-\left(a/\Delta t \boldsymbol{L}_p^{-1} - \nu \boldsymbol{I}\right)$ plays the role of a preconditioner.

Appendix C. Connections between pressure correction and SIMPLE-based methods

The SIMPLE-based schemes are usually presented with the discrete equations for each degree of freedom[7] (e.g. see Sec. 7.3.4 of [21] and Ch. 6 of [65]). With such formulation, it is hard to discern the connection between SIMPLE-based and pressure-correction projection methods, because the latter are usually presented with time-discrete but space-continuous equations such as (13).

The key for our new intuitive connection is to rewrite both families of schemes in matrix form. [44] attempts this approach to bridge SIMPLE, SIMPLEC, and incremental pressure-correction schemes, but the matrix equations formulated there still mimic the space-discrete equations by identifying the terms due to each neighboring cell explicitly, e.g. using $\boldsymbol{L}_{\boldsymbol{u}}\boldsymbol{u} = \boldsymbol{L}_{\boldsymbol{u},P}\boldsymbol{u} + \sum_M \boldsymbol{L}_{\boldsymbol{u},M}\boldsymbol{u}_M$ for the discrete vector Laplacian, where M loops over all neighboring cells involved in the stencil of the spatial schemes. Here the $\boldsymbol{u}_M$'s are different permuted vectors of $\boldsymbol{u}$ and the entries of the diagonal matrices $\boldsymbol{L}_{\boldsymbol{u},P}$ and $\boldsymbol{L}_{\boldsymbol{u},M}$'s simply come from a partition of the nonzero entries of $\boldsymbol{L}_{\boldsymbol{u}}$. Indeed, the same present-neighboring-cell decomposition can be formulated in a more elegant way by splitting a matrix into its diagonal and off-diagonal part, e.g. $\boldsymbol{L}_{\boldsymbol{u}} = \boldsymbol{L}_{\boldsymbol{u},\text{D}} + \boldsymbol{L}_{\boldsymbol{u},\text{OD}}$, where the diagonal part $\boldsymbol{L}_{\boldsymbol{u},\text{D}}$ is the same as $\boldsymbol{L}_{\boldsymbol{u},P}$ in the previous formulation. As we shall see, a better notation will help provide new deeper insights.

SIMPLE. SIMPLE and incremental pressure-correction projection methods are connected through the philosophy of the prediction-correction or operator splitting strategy, which is shared in their motivation and derivation. Since it is costly to directly solve the fully-coupled system (5), an idea other than the pressure-Schur complement approach is to first solve for velocity using the momentum equation with the unknown pressure approximated by the pressure at the previous time step, i.e.

$$\boldsymbol{A}\boldsymbol{u}_*^{n+1} + \boldsymbol{G}p^n = \boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1}. \tag{C.1}$$

Then, we want to obtain $\boldsymbol{u}^{n+1}$ and $\boldsymbol{G}p^{n+1}$ that satisfy (5) by adding a correction to $\boldsymbol{u}_*^{n+1}$ and to $\boldsymbol{G}p^n$, respectively. Therefore, we subtract the prediction scheme (C.1) from the desired aggregate scheme (i.e. the discrete momentum equation of the fully-coupled scheme (5))

$$\boldsymbol{A}\boldsymbol{u}^{n+1} + \boldsymbol{G}p^{n+1} = \boldsymbol{f}_{\text{rhs}}^{n+1,n} + \text{bc}_{\text{mom}}^{n+1}, \tag{C.2}$$

to obtain a relation between the velocity correction and pressure correction:

$$\boldsymbol{A}\left(\boldsymbol{u}^{n+1} - \boldsymbol{u}_*^{n+1}\right) = -\boldsymbol{G}(p^{n+1} - p^n). \tag{C.3}$$

If we also want the divergence-free constraint $\boldsymbol{D}\boldsymbol{u}^{n+1} + \text{bc}_{\text{con}}^{n+1} = 0$ to be satisfied, the pressure correction has to be the unique solution of

$$\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}(p^{n+1} - p^n) = \boldsymbol{D}\boldsymbol{u}_*^{n+1} + \text{bc}_{\text{con}}^{n+1}. \tag{C.4}$$

However, as we can see, there is no free lunch because as long as we want (5) to be satisfied exactly, we run into the pressure-Schur complement $\boldsymbol{D}\boldsymbol{A}^{-1}\boldsymbol{G}$, which is difficult to solve as already mentioned in Sec. 3.

To alleviate this difficulty, we could find a proxy for $\boldsymbol{A}$ used only in the velocity and pressure correction, but not in the prediction. A simple idea is to approximate $\boldsymbol{A}$ by some diagonal matrix. For such purpose, $\boldsymbol{A}$ can be split into the sum of two parts $\boldsymbol{A}_\text{t} + \boldsymbol{A}_\text{s}$, where $\boldsymbol{A}_\text{t} = (a/\Delta t)\boldsymbol{I}$ is from discretizing the time derivative and is diagonal, while $\boldsymbol{A}_\text{s}$ is from the implicitly-treated spatial-derivative terms, e.g. $\boldsymbol{A}_\text{s} = -\nu \boldsymbol{L}_{\boldsymbol{u}}$ if the diffusion term is treated implicitly and the advection

[7] A degree of freedom is, for example, the value of a variable at an individual grid point for finite differences or at an individual cell for finite volumes.

term explicitly. However, there are different justifiable diagonal proxies for $\boldsymbol{A}$. This is where SIMPLE and pressure-correction projection methods diverge.

The incremental pressure-correction projection methods in non-rotational form are based on the space-continuous counterpart of (C.3), i.e.

$$\frac{a}{\Delta t}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)-\nu\nabla^2\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)=-\nabla\left(p^{n+1}-p^n\right), \tag{C.5}$$

with the diffusion term (and other space-derivative terms if they are also treated implicitly) ignored. This is equivalent to dropping $\boldsymbol{A}_\mathrm{s}$ and approximating $\boldsymbol{A}$ by $\boldsymbol{A}_\mathrm{t}$. Since all off-diagonal entries of $\boldsymbol{A}$ are due to discretizing the space-derivative terms, this leads to a diagonal proxy $\boldsymbol{A}\approx\boldsymbol{A}_\mathrm{t}=(a/\Delta t)\boldsymbol{I}$. Therefore, this gives rise to the correction equations

$$\frac{a}{\Delta t}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)=-\boldsymbol{G}(p^{n+1}-p^n), \tag{C.6a}$$

$$\frac{\Delta t}{a}\boldsymbol{DG}(p^{n+1}-p^n)=\boldsymbol{D}\boldsymbol{u}_*^{n+1}+\mathrm{bc}_\mathrm{con}^{n+1}, \tag{C.6b}$$

which clearly has a space-continuous counterpart

$$\frac{a}{\Delta t}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)=-\nabla(p^{n+1}-p^n), \tag{C.7a}$$

$$\nabla^2(p^{n+1}-p^n)=\frac{a}{\Delta t}\nabla\cdot\boldsymbol{u}_*^{n+1}, \tag{C.7b}$$

as seen in most expositions of this type of schemes.

In contrast, SIMPLE only looks at the matrix form of the equations and ignores where the matrix comes from and how it is assembled, similar to the philosophy underlying the Jacobi and Gauss–Seidel iterations for solving a linear system. The key idea of SIMPLE is to directly drop all the off-diagonal entries of $\boldsymbol{A}$ to obtain a diagonal proxy $\boldsymbol{A}_\mathrm{D}$. Note that $\boldsymbol{A}_\mathrm{D}=\boldsymbol{A}_\mathrm{t}+\boldsymbol{A}_\mathrm{s,D}$ contains both $\boldsymbol{A}_\mathrm{t}=(a/\Delta t)\boldsymbol{I}$ from discretizing the time derivative and the diagonal part of $\boldsymbol{A}_\mathrm{s}$ from discretizing the space-derivative terms. Hence, if $\boldsymbol{A}=\boldsymbol{A}_\mathrm{t}+\boldsymbol{A}_\mathrm{s}=(a/\Delta t)\boldsymbol{I}-\nu\boldsymbol{L}_{\boldsymbol{u}}$, then $\boldsymbol{A}_\mathrm{D}=(a/\Delta t)\boldsymbol{I}-\nu\boldsymbol{L}_{\boldsymbol{u},\mathrm{D}}$, where $\boldsymbol{L}_{\boldsymbol{u},\mathrm{D}}$ is the diagonal part of $\boldsymbol{L}_{\boldsymbol{u}}$. This gives rise to the correction equations

$$\boldsymbol{A}_\mathrm{D}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)=-\boldsymbol{G}(p^{n+1}-p^n), \tag{C.8a}$$

$$\boldsymbol{D}\boldsymbol{A}_\mathrm{D}^{-1}\boldsymbol{G}(p^{n+1}-p^n)=\boldsymbol{D}\boldsymbol{u}_*^{n+1}+\mathrm{bc}_\mathrm{con}^{n+1}, \tag{C.8b}$$

which does not have a space-continuous counterpart due to the lack of a continuous counterpart of the splitting $\boldsymbol{L}_{\boldsymbol{u}}=\boldsymbol{L}_{\boldsymbol{u},\mathrm{D}}+\boldsymbol{L}_{\boldsymbol{u},\mathrm{OD}}$. Note that although $\boldsymbol{A}_\mathrm{D}$ is diagonal, it is not a scalar matrix.

Sometimes the term “projection methods” is used in a more general way (e.g. [21]) to refer to any velocity–pressure decoupling schemes that first predict a new velocity with old pressure and then correct the velocity and pressure such that the corrected new velocity is divergence-free. Only in this loose sense can SIMPLE also be viewed as a pressure-correction projection method. However, as is shown above, SIMPLE does not have a space-continuous interpretation and it is by no means a projection method in the narrow sense of utilizing the Helmholtz decomposition in the correction step.

SIMPLEC. SIMPLEC (“C” for “consistent”) was proposed by [64] as an improvement of SIMPLE. SIMPLE ignores $\boldsymbol{A}_\mathrm{OD}$, the off-diagonal part of the space-derivative terms, but keeps their diagonal part $\boldsymbol{A}_\mathrm{s,D}=\boldsymbol{A}_\mathrm{D}-\boldsymbol{A}_\mathrm{t}$, which is of the same order of magnitude. Therefore, this makes the approximation inconsistent with itself and not well justified. The idea of SIMPLEC is to approximate $\boldsymbol{A}_\mathrm{OD}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)=\sum_M\boldsymbol{A}_\mathrm{M}\left(\boldsymbol{u}_M^{n+1}-\boldsymbol{u}_{*M}^{n+1}\right)$ by $\sum_M\boldsymbol{A}_\mathrm{M}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)$, i.e. substitute the present cell value into the neighboring cell values in the equation at each cell (corresponding to each row of $\boldsymbol{A}_\mathrm{OD}$). Again, we can get rid of the awkward loop-over-neighbor notation by noticing that $\sum_M\boldsymbol{A}_\mathrm{M}$ is the diagonal matrix whose diagonal entries are simply the sum of all the off-diagonal entries of $\boldsymbol{A}_\mathrm{s}$ in each row. Note also that every consistent discretization scheme of a derivative should have all its coefficients sum to zero, therefore, we have $\sum_M\boldsymbol{A}_\mathrm{M}=-\boldsymbol{A}_\mathrm{s,D}$. In summary, SIMPLEC approximates the off-diagonal part by a diagonal term as

$$\boldsymbol{A}_\mathrm{OD}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)\approx-\boldsymbol{A}_\mathrm{s,D}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right).$$

Hence, the diagonal proxy corresponding to this reduces to $\boldsymbol{A}_\mathrm{D}-\boldsymbol{A}_\mathrm{s,D}=\boldsymbol{A}_\mathrm{t}+\boldsymbol{A}_\mathrm{s,D}-\boldsymbol{A}_\mathrm{s,D}=\boldsymbol{A}_\mathrm{t}$. Thus, as is also pointed out in [44], we end up with exactly the incremental pressure-correction scheme in non-rotational form! Each discrete equation of SIMPLEC thus indeed has a space-continuous counterpart, which can be viewed as another sense of “consistency”. We note that [64] did not mention pressure correction projection methods nor this equivalence. Their goal was not a set of discrete equations with a continuous interpretation. Now with this equivalence, their “consistency” argument can be paraphrased as: For any term in a continuous equation, we could typically either ignore it or keep it as a whole but not to split its discrete form in a way without a continuous interpretation.

The improvement achieved by SIMPLEC compared to SIMPLE seems to again verify the rule of thumb in numerical methods for PDEs that taking into account the structure of the original equations usually yields more effective algorithms. This is also a motivation for the iterated pressure-correction projection methods proposed in this paper.

SIMPLER. SIMPLER [47] and PISO [29] are variants of SIMPLE with additional correction steps. Although the original algorithms retain SIMPLE's choice of the diagonal proxy for $\boldsymbol{A}$, this choice is actually irrelevant and the key ideas can be applied to pressure-correction projection methods as well.

The revision ("R" for "revised") of SIMPLE in SIMPLER lies in the procedure for computing the pressure at the new time step. The key idea is that after obtaining the corrected velocity and pressure with SIMPLE, the corrected pressure is tossed away but a new corrected pressure is computed by solving some discrete pressure Poisson equation (PPE) with the corrected velocity. This idea is based on the observation that with SIMPLE, the corrected velocity is usually much more accurate than the corrected pressure. Since in incompressible flows, the PPE will yield an exact pressure given the exact velocity, a natural idea is to use the PPE to compute the pressure with the divergence-free and rather accurate corrected velocity. In particular, to avoid a pressure-Schur complement, we derive the discrete PPE by rewriting the desired aggregate scheme (C.2) as

$$\boldsymbol{A}_{\mathrm{D}}\boldsymbol{u}^{n+1} + \boldsymbol{A}_{\mathrm{OD}}\boldsymbol{u}^{n+1} + \boldsymbol{G}p^{n+1} = \boldsymbol{f}_{\mathrm{rhs}}^{n+1,n} + \mathrm{bc}_{\mathrm{mom}}^{n+1} \tag{C.9}$$

and using $\boldsymbol{D}\boldsymbol{u}^{n+1} + \mathrm{bc}_{\mathrm{con}}^{n+1} = 0$ to obtain

$$\boldsymbol{D}\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{G}p^{n+1} = -\boldsymbol{D}\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{A}_{\mathrm{OD}}\boldsymbol{u}^{n+1} + \boldsymbol{D}\boldsymbol{A}_{\mathrm{D}}^{-1}\left(\boldsymbol{f}_{\mathrm{rhs}}^{n+1,n} + \mathrm{bc}_{\mathrm{mom}}^{n+1}\right) + \mathrm{bc}_{\mathrm{con}}^{n+1}. \tag{C.10}$$

This is similar to how the pressure correction is derived in SIMPLE. It appears to be a free lunch in the sense that (5) seems to be achieved with the pressure-Schur complement bypassed, but this is not the case because with the action of discrete divergence $\boldsymbol{D}$, (C.10) is no longer equivalent to (C.9). Although (C.10) has a unique solution (up to a constant), (C.9) in general does not have a solution for p^{n+1} with a given divergence-free $\boldsymbol{u}^{n+1}$ because $\left(\boldsymbol{f}_{\mathrm{rhs}}^{n+1,n} + \mathrm{bc}_{\mathrm{mom}}^{n+1} - \boldsymbol{A}\boldsymbol{u}^{n+1}\right)$ may not lie in the column space of the rectangular $\boldsymbol{G}$.

If we apply this same idea to incremental pressure-correction schemes, (C.9) becomes

$$\frac{a}{\Delta t}\boldsymbol{u}^{n+1} + \boldsymbol{A}_{\mathrm{s}}\boldsymbol{u}^{n+1} + \boldsymbol{G}p^{n+1} = \boldsymbol{f}_{\mathrm{rhs}}^{n+1,n} + \mathrm{bc}_{\mathrm{mom}}^{n+1} \tag{C.11}$$

and thus the discrete PPE (C.10) becomes

$$\boldsymbol{D}\boldsymbol{G}p^{n+1} = -\boldsymbol{D}\boldsymbol{A}_{\mathrm{s}}\boldsymbol{u}^{n+1} + \boldsymbol{D}\left(\boldsymbol{f}_{\mathrm{rhs}}^{n+1,n} + \mathrm{bc}_{\mathrm{mom}}^{n+1}\right) + \frac{a}{\Delta t}\mathrm{bc}_{\mathrm{con}}^{n+1}, \tag{C.12}$$

which is indeed a Poisson equation now because $\boldsymbol{D}\boldsymbol{G} = \boldsymbol{L}_p$. Note that since $\boldsymbol{D} = -\boldsymbol{G}^{\mathrm{T}}$, (C.12) is equivalent to solving for the least square solution of p^{n+1} to the aggregate scheme (C.2), i.e.

$$\boldsymbol{G}p^{n+1} = \boldsymbol{f}_{\mathrm{rhs}}^{n+1,n} + \mathrm{bc}_{\mathrm{mom}}^{n+1} - \boldsymbol{A}\boldsymbol{u}^{n+1}.$$

However, the counterpart in SIMPLER (C.10) does not have such an interpretation because $\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{G}$ has a different column space than $\boldsymbol{G}$ has. This is yet another evidence that the discrete operator splitting used in SIMPLE is bad. Indeed, we can interpret this least square procedure as a discrete counterpart of performing the Helmholtz decomposition of the sum of all terms other than ∇p^{n+1} in the momentum equation, i.e. we take the orthogonal projection of this sum onto the irrotational subspace as ∇p^{n+1}. Therefore, both SIMPLER and this revised pressure-correction scheme can be viewed as the first iteration of the following alternating projection procedure:

1. Insert the latest estimate of p^{n+1} into the momentum equation to obtain a new predicted velocity $\boldsymbol{u}_*^{n+1}$.
2. Obtain a new corrected velocity $\boldsymbol{u}^{n+1}$ as the orthogonal projection of $\boldsymbol{u}_*^{n+1}$ onto the divergence-free subspace.
3. Obtain a new estimate of p^{n+1} such that ∇p^{n+1} is the orthogonal projection of the sum of other terms in the momentum equation onto the irrotational subspace.

Obviously, nothing prevents us from iterating more than once. We can iterate until $\boldsymbol{u}^{n+1}$ and p^{n+1} converge to the solution to the fully-coupled system. Compared to the above SIMPLER-based iteration, the iterated pressure-correction schemes proposed in this paper solves one fewer scalar Poisson equation in each iteration and with the rotational correction, if the advection term is treated explicitly, each iteration is able to yield a solution that satisfies the fully-coupled system up to round-off errors except at the cells right next to a boundary.

PISO. Although PISO [29] is a modification of SIMPLE, as for SIMPLER, its key idea is not bound to SIMPLE and can also be applied to pressure-correction projection methods. The idea of PISO is to iterate the velocity and pressure correction steps to solve the exact correction equations

$$\begin{cases} \boldsymbol{A}\left(\boldsymbol{u}^{n+1} - \boldsymbol{u}_*^{n+1}\right) = -\boldsymbol{G}(p^{n+1} - p^n), & \text{(a)} \\ -\boldsymbol{D}\boldsymbol{u}^{n+1} = \mathrm{bc}_{\mathrm{con}}^{n+1}, & \text{(b)} \end{cases} \tag{C.13}$$

although the original PISO algorithm stops after iterating only twice. Denoting by $\boldsymbol{u}_k^{n+1}$ and p_k^{n+1} the approximate solution after the k-th iteration and using the same operator splitting as SIMPLE, we rewrite (C.13a) as

$$\boldsymbol{A}_{\mathrm{D}}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)=-\boldsymbol{A}_{\mathrm{OD}}\left(\boldsymbol{u}^{n+1}-\boldsymbol{u}_*^{n+1}\right)-\boldsymbol{G}(p^{n+1}-p^n)$$

and obtain a recursion

$$\begin{cases}\boldsymbol{u}_{k+1}^{n+1}-\boldsymbol{u}_*^{n+1}=-\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{A}_{\mathrm{OD}}\left(\boldsymbol{u}_k^{n+1}-\boldsymbol{u}_*^{n+1}\right)-\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{G}(p_{k+1}^{n+1}-p^n), & \text{(a)}\\ \boldsymbol{D}\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{G}(p_{k+1}^{n+1}-p^n)=-\boldsymbol{D}\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{A}_{\mathrm{OD}}\left(\boldsymbol{u}_k^{n+1}-\boldsymbol{u}_*^{n+1}\right)+\boldsymbol{D}\boldsymbol{u}_*^{n+1}+\mathrm{bc}_{\mathrm{con}}^{n+1} & \text{(b)}\end{cases} \tag{C.14}$$

where p_{k+1}^{n+1} is determined such that $\boldsymbol{u}_{k+1}^{n+1}$ is divergence-free, i.e. $\boldsymbol{D}\boldsymbol{u}_{k+1}^{n+1}+\mathrm{bc}_{\mathrm{con}}^{n+1}=0$. The initial guess is chosen as $\boldsymbol{u}_0^{n+1}=\boldsymbol{u}_*^{n+1}$. For $k\geq 1$, we can equivalently solve for the iteration increment instead:

$$\begin{cases}\boldsymbol{u}_{k+1}^{n+1}-\boldsymbol{u}_k^{n+1}=-\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{A}_{\mathrm{OD}}\left(\boldsymbol{u}_k^{n+1}-\boldsymbol{u}_{k-1}^{n+1}\right)-\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{G}(p_{k+1}^{n+1}-p_k^{n+1}), & \text{(a)}\\ \boldsymbol{D}\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{G}(p_{k+1}^{n+1}-p_k^{n})=-\boldsymbol{D}\boldsymbol{A}_{\mathrm{D}}^{-1}\boldsymbol{A}_{\mathrm{OD}}\left(\boldsymbol{u}_k^{n+1}-\boldsymbol{u}_{k-1}^{n+1}\right), & \text{(b)}\end{cases} \tag{C.15}$$

which is how the original PISO algorithm is presented. Note that with one iteration of (C.14), the algorithm reduces to SIMPLE. With two iterations, it becomes PISO. If we keep iterating, the converging point will be the solution to the fully-coupled system.

If we apply the same idea to incremental pressure-correction projection methods, we obtain the following iteration:

$$\begin{cases}\dfrac{a}{\Delta t}\left(\boldsymbol{u}_{k+1}^{n+1}-\boldsymbol{u}_*^{n+1}\right)=-\boldsymbol{A}_{\mathrm{s}}\left(\boldsymbol{u}_k^{n+1}-\boldsymbol{u}_*^{n+1}\right)-\boldsymbol{G}(p_{k+1}^{n+1}-p^n), & \text{(a)}\\ \boldsymbol{D}\boldsymbol{G}(p_{k+1}^{n+1}-p^n)=-\boldsymbol{D}\boldsymbol{A}_{\mathrm{s}}\left(\boldsymbol{u}_k^{n+1}-\boldsymbol{u}_*^{n+1}\right)+\dfrac{a}{\Delta t}\left(\boldsymbol{D}\boldsymbol{u}_*^{n+1}+\mathrm{bc}_{\mathrm{con}}^{n+1}\right), & \text{(b)}\end{cases} \tag{C.16}$$

which, when iterated only once, reduces to exactly the incremental pressure-correction schemes in non-rotational form.

When we take the iteration perspective on PISO, the single prediction step that precedes the iterations actually appears unnatural. Since the system of velocity and pressure correction (C.13) has exactly the same coefficient matrix as the fully-coupled system (5), if the PISO iteration is efficient in solving this saddle-point problem, we should instead directly apply it to (5), i.e.

$$\begin{cases}\dfrac{a}{\Delta t}\boldsymbol{u}_{k+1}^{n+1}=-\boldsymbol{A}_{\mathrm{s}}\boldsymbol{u}_k^{n+1}-\boldsymbol{G}p_{k+1}^{n+1}+\boldsymbol{f}_{\mathrm{rhs}}^{n+1,n}+\mathrm{bc}_{\mathrm{mom}}^{n+1}, & \text{(a)}\\ \boldsymbol{D}\boldsymbol{G}p_{k+1}^{n+1}=-\boldsymbol{D}\boldsymbol{A}_{\mathrm{s}}\boldsymbol{u}_k^{n+1}+\boldsymbol{D}\left(\boldsymbol{f}_{\mathrm{rhs}}^{n+1,n}+\mathrm{bc}_{\mathrm{mom}}^{n+1}\right)+\dfrac{a}{\Delta t}\mathrm{bc}_{\mathrm{con}}^{n+1}, & \text{(b)}\end{cases} \tag{C.17}$$

with the initial guess $\boldsymbol{u}_0^{n+1}=\boldsymbol{u}^n$, which obviates the prediction step required for (C.16). The reason why this strategy is not adopted is likely that treating the diffusion explicitly in each iteration (though implicitly in the time marching scheme) will make the convergence unacceptably slow due to the global effects of diffusion. However, if this is the case, we should not expect (C.14) or (C.16) to significantly improve the solution obtained by only one iteration.

This is yet another motivation for the iterated pressure-correction schemes proposed in this paper, which iterate both the prediction and the pressure correction step, and are thus different from PISO, which iterates the correction step only. Also, our new iteration can make use of the rotational correction when an incremental pressure-correction projection method is used. As shown in Sec. 4 and 6, the rotational correction is essential in yielding a mesh- and viscosity-independent converging rate for the iteration.

References

[1] R. Alexander, Diagonally implicit Runge–Kutta methods for stiff ode's, SIAM J. Numer. Anal. 14 (6) (1977) 1006–1021.

[2] D.G. Anderson, Iterative procedures for nonlinear integral equations, J. ACM 12 (4) (1965) 547–560.

[3] J.P. Aoussou, An Iterative Pressure-Correction Method for the Unsteady Incompressible Navier–Stokes Equation, Master's thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, 2016.

[4] U.M. Ascher, S.J. Ruuth, R.J. Spiteri, Implicit–explicit Runge–Kutta methods for time-dependent partial differential equations, Appl. Numer. Math. 25 (2–3) (1997) 151–167.

[5] U.M. Ascher, S.J. Ruuth, B.T. Wetton, Implicit–explicit methods for time-dependent partial differential equations, SIAM J. Numer. Anal. 32 (3) (1995) 797–823.

[6] H. Baek, G.E. Karniadakis, Sub-iteration leads to accuracy and stability enhancements of semi-implicit schemes for the Navier–Stokes equations, J. Comput. Phys. 230 (12) (2011) 4384–4402.

[7] M. Balda, Vector Aitken's delta-square accelerator, Mathworks File Exchange (2006).

[8] R. Barrett, M.W. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. Van der Vorst, Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, vol. 43, SIAM, 1994.

[9] M. Benzi, G.H. Golub, J. Liesen, Numerical solution of saddle point problems, Acta Numer. 14 (2005) 1–137.

[10] M. Cai, Analysis of some projection method based preconditioners for models of incompressible flow, Appl. Numer. Math. 90 (2015) 77–90.

[11] C.G. Canuto, M.Y. Hussaini, A.M. Quarteroni, T.A. Zang, Spectral Methods: Evolution to Complex Geometries and Applications to Fluid Dynamics (Scientific Computation), Springer-Verlag, New York, 2007.

[12] L. Caretto, A. Gosman, S. Patankar, D. Spalding, Two calculation procedures for steady, three-dimensional flows with recirculation, in: Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics, Springer, 1973, pp. 60–68.

[13] E. Celledoni, B.K. Kometa, O. Verdier, High order semi-Lagrangian methods for the incompressible Navier–Stokes equations, J. Sci. Comput. (2015) 1–25.
[14] A.J. Chorin, Numerical solution of the Navier–Stokes equations, Math. Comput. 22 (104) (1968) 745–762.
[15] L.O. Chua, P.Y. Lin, Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques, Prentice Hall Professional Technical Reference, 1975.
[16] O. Colomés, S. Badia, Segregated Runge–Kutta methods for the incompressible Navier–Stokes equations, Int. J. Numer. Methods Eng. 105 (5) (2016) 372–400.
[17] M. Costabel, M. Crouzeix, M. Dauge, Y. Lafranche, The inf-sup constant for the divergence on corner domains, Numer. Methods Partial Differ. Equ. 31 (2) (2015) 439–458.
[18] M.A. Cubillos-Moraga, General-Domain Compressible Navier–Stokes Solvers Exhibiting Quasi-Unconditional Stability and High-Order Accuracy in Space and Time, PhD thesis, California Institute of Technology, 2015.
[19] S. Detrembleur, B. Dewals, P. Archambeau, S. Erpicum, M. Pirotton, An explicit projection method for solving incompressible Navier–Stokes equations, in: Hydraulic Structures, 2008, pp. 227–235.
[20] F. Feppon, P.F.J. Lermusiaux, Dynamically orthogonal numerical schemes for efficient stochastic advection and Lagrangian transport, SIAM Rev. 60 (3) (2018), https://doi.org/10.1137/16M1109394.
[21] J.H. Ferziger, M. Peric, Computational Methods for Fluid Dynamics, Springer Science & Business Media, 2012.
[22] O. Ghattas, J.-H. Bark, Optimal control of two-and three-dimensional incompressible Navier–Stokes flows, J. Comput. Phys. 136 (2) (1997) 231–244.
[23] R. Glowinski, Finite element methods for the numerical simulation of incompressible viscous flow. Introduction to the control of the Navier–Stokes equations, in: Lectures in Applied Mathematics, vol. 28, 1991, pp. 219–301.
[24] J. Guermond, P. Minev, J. Shen, An overview of projection methods for incompressible flows, Comput. Methods Appl. Mech. Eng. 195 (44) (2006) 6011–6045.
[25] J. Guermond, J. Shen, On the error estimates for the rotational pressure-correction projection methods, Math. Comput. 73 (248) (2004) 1719–1737.
[26] J.-L. Guermond, L. Quartapelle, On stability and convergence of projection methods based on pressure Poisson equation, Int. J. Numer. Methods Fluids 26 (9) (1998) 1039–1053.
[27] W. Hundsdorfer, S.J. Ruuth, IMEX extensions of linear multistep methods with general monotonicity and boundedness properties, J. Comput. Phys. 225 (2) (2007) 2016–2042.
[28] W. Hundsdorfer, J.G. Verwer, Numerical Solution of Time-Dependent Advection–Diffusion–Reaction Equations, vol. 33, Springer Science & Business Media, 2013.
[29] R.I. Issa, Solution of the implicitly discretised fluid flow equations by operator-splitting, J. Comput. Phys. 62 (1) (1986) 40–65.
[30] M. Kang, R.P. Fedkiw, X.-D. Liu, A boundary condition capturing method for multiphase incompressible flow, J. Sci. Comput. 15 (3) (2000) 323–360.
[31] C.A. Kennedy, M.H. Carpenter, Additive Runge–Kutta schemes for convection–diffusion–reaction equations, Appl. Numer. Math. 44 (1–2) (2003) 139–181.
[32] J. Kim, P. Moin, Application of a fractional-step method to incompressible Navier–Stokes equations, J. Comput. Phys. 59 (2) (1985) 308–323.
[33] O. Le Maître, O.M. Knio, Spectral Methods for Uncertainty Quantification: With Applications to Computational Fluid Dynamics, Springer Science & Business Media, 2010.
[34] P.F.J. Lermusiaux, Numerical Fluid Mechanics, MIT OpenCourseWare, 2015.
[35] P.F.J. Lermusiaux, P.J. Haley Jr., S. Jana, A. Gupta, C.S. Kulkarni, C. Mirabito, W.H. Ali, D.N. Subramani, A. Dutt, J. Lin, A.Y. Shcherbina, C.M. Lee, A. Gangopadhyay, Optimal planning and sampling predictions for autonomous and Lagrangian platforms and sensors in the northern Arabian Sea, Oceanography 30 (2) (2017) 172–185, https://doi.org/10.5670/oceanog.2017.242.
[36] J.-G. Liu, Projection method I: convergence and numerical boundary layers, SIAM J. Numer. Anal. 32 (4) (1995) 1017–1057.
[37] R. Löhner, Multistage explicit advective prediction for projection-type incompressible flow solvers, J. Comput. Phys. 195 (1) (2004) 143–152.
[38] T. Lolla, P.F.J. Lermusiaux, M.P. Ueckermann, P.J. Haley Jr., Time-optimal path planning in dynamic flows using level set equations: theory and schemes, Ocean Dyn. 64 (10) (2014) 1373–1397, https://doi.org/10.1007/s10236-014-0757-y.
[39] A.J. Macleod, Acceleration of vector sequences by multi-dimensional Δ^2 methods, Commun. Appl. Numer. Methods 2 (4) (1986) 385–392.
[40] Y. Maday, D. Meiron, A.T. Patera, E.M. Rønquist, Analysis of iterative methods for the steady and unsteady Stokes problem: application to spectral element discretizations, SIAM J. Sci. Comput. 14 (2) (1993) 310–337.
[41] K.-A. Mardal, H. Langtangen, Mixed finite elements, in: Advanced Topics in Computational Partial Differential Equations, Springer, 2003, pp. 153–197.
[42] P. Minev, P. Gresho, A remark on pressure correction schemes for transient viscous incompressible flow, Commun. Numer. Methods Eng. 14 (4) (1998) 335–346.
[43] N.C. Nguyen, J. Peraire, B. Cockburn, An implicit high-order hybridizable discontinuous Galerkin method for the incompressible Navier–Stokes equations, J. Comput. Phys. 230 (4) (2011) 1147–1170.
[44] M.-J. Ni, M.A. Abdou, A bridge between projection methods and simple type methods for incompressible Navier–Stokes equations, Int. J. Numer. Methods Eng. 72 (12) (2007) 1490–1512.
[45] M.-J. Ni, S. Komori, N. Morley, Projection methods for the calculation of incompressible unsteady flows, Numer. Heat Transf., Part B, Fundam. 44 (6) (2003) 533–551.
[46] M.A. Olshanskii, Y.V. Vassilevski, Pressure Schur complement preconditioners for the discrete Oseen problem, SIAM J. Sci. Comput. 29 (6) (2007) 2686–2704.
[47] S. Patankar, Numerical Heat Transfer and Fluid Flow, CRC Press, 1980.
[48] G. Ren, T. Utnes, A finite element solution of the time-dependent incompressible Navier–Stokes equations using a modified velocity correction method, Int. J. Numer. Methods Fluids 17 (5) (1993) 349–364.
[49] A. Sandu, M. Günther, A generalized-structure approach to additive Runge–Kutta methods, SIAM J. Numer. Anal. 53 (1) (2015) 17–42.
[50] L.F. Shampine, Numerical Solution of Ordinary Differential Equations, vol. 4, CRC Press, 1994.
[51] J. Shen, On error estimates of the projection methods for the Navier–Stokes equations: second-order schemes, Math. Comput. 65 (215) (1996) 1039–1065.
[52] D. Shirokoff, R.R. Rosales, An efficient method for the incompressible Navier–Stokes equations on irregular domains with no-slip boundary conditions, high order up to the boundary, J. Comput. Phys. 230 (23) (2011) 8619–8646.
[53] T. Sondergaard, P.F.J. Lermusiaux, Data assimilation with Gaussian mixture models using the dynamically orthogonal field equations. Part I: theory and scheme, Mon. Weather Rev. 141 (6) (2013) 1737–1760, https://doi.org/10.1175/MWR-D-11-00295.1.
[54] D.N. Subramani, P.J. Haley Jr., P.F.J. Lermusiaux, Energy-optimal path planning in the coastal ocean, J. Geophys. Res., Oceans 122 (2017) 3981–4003, https://doi.org/10.1002/2016JC012231.
[55] D.N. Subramani, P.F.J. Lermusiaux, Energy-optimal path planning by stochastic dynamically orthogonal level-set optimization, Ocean Model. 100 (2016) 57–77, https://doi.org/10.1016/j.ocemod.2016.01.006.
[56] R. Temam, Une méthode d'approximation de la solution des équations de Navier–Stokes, Bull. Soc. Math. Fr. 96 (1968) 115–152.
[57] L. Timmermans, P. Minev, F. Van De Vosse, An approximate projection scheme for incompressible flow using spectral elements, Int. J. Numer. Methods Fluids 22 (7) (1996) 673–688.
[58] S. Turek, A comparative study of time-stepping techniques for the incompressible Navier–Stokes equations: from fully implicit non-linear schemes to semi-implicit projection methods, Int. J. Numer. Methods Fluids 22 (10) (1996) 987–1011.

[59] S. Turek, On discrete projection methods for the incompressible Navier–Stokes equations: an algorithmical approach, Comput. Methods Appl. Mech. Eng. 143 (3) (1997) 271–288.
[60] S. Turek, Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approach, vol. 6, Springer Science & Business Media, 1999.
[61] M.P. Ueckermann, P.F.J. Lermusiaux, 2.29 Finite Volume MATLAB Framework Documentation, MSEAS Report 14, Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, 2012.
[62] M.P. Ueckermann, P.F.J. Lermusiaux, Hybridizable discontinuous Galerkin projection methods for Navier–Stokes and Boussinesq equations, J. Comput. Phys. 306 (2016) 390–421, https://doi.org/10.1016/j.jcp.2015.11.028.
[63] M.P. Ueckermann, P.F.J. Lermusiaux, T.P. Sapsis, Numerical schemes for dynamically orthogonal equations of stochastic fluid and ocean flows, J. Comput. Phys. 233 (2013) 272–294, https://doi.org/10.1016/j.jcp.2012.08.041.
[64] J. van Doormaal, G. Raithby, Enhancements of the simple method for predicting incompressible fluid flows, Numer. Heat Transf. 7 (2) (1984) 147–163.
[65] H.K. Versteeg, W. Malalasekera, An Introduction to Computational Fluid Dynamics: The Finite Volume Method, Pearson education, 2nd edition, 2007.
[66] D.S. Watkins, Fundamentals of Matrix Computations, vol. 64, John Wiley & Sons, 2004.
[67] P. Wesseling, Principles of Computational Fluid Dynamics, vol. 29, Springer Science & Business Media, 2009.
[68] M.J. Zahr, P.-O. Persson, High-order, time-dependent aerodynamic optimization using a discontinuous Galerkin discretization of the Navier–Stokes equations, in: AIAA Science and Technology Forum and Exposition, SciTech 2016, San Diego, California, 2016.
[69] F. Zhang, The Schur Complement and Its Applications, vol. 4, Springer Science & Business Media, 2006.
[70] Q. Zhang, A fourth-order approximate projection method for the incompressible Navier–Stokes equations on locally-refined periodic domains, Appl. Numer. Math. 77 (2014) 16–30.
[71] D. Zhou, High-Order Numerical Methods for Pressure Poisson Equation Reformulations of the Incompressible Navier–Stokes Equations, PhD thesis, Temple University, 2014.