Journal of Computational Physics 375 (2018) 797–822

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Higher-order temporal integration for the incompressible Navier–Stokes equations in bounded domains

M.L. Minion [a], R.I. Saye [b,*]

[a] Applied Mathematics Department, Lawrence Berkeley National Laboratory, Berkeley, CA, 94720, United States of America
[b] Mathematics Group, Lawrence Berkeley National Laboratory, Berkeley, CA, 94720, United States of America

ARTICLE INFO

Article history:
Received 9 March 2018
Received in revised form 29 August 2018
Accepted 29 August 2018
Available online 5 September 2018

Keywords:
Projection methods
Gauge methods
Auxiliary variable formulation
Spectral deferred correction

ABSTRACT

This paper compares and contrasts higher-order, semi-implicit temporal integration strategies for the incompressible Navier–Stokes methods based on spectral deferred corrections applied to certain gauge or auxiliary variable formulations of the equations. Particular focus is placed on the imposition of boundary conditions in the semi-implicit formulation, the accurate treatment of the pressure term, and the smoothness of the numerical solution for different formulations. The main result presented here is the formulation and numerical validation of a single-step, semi-implicit method called *spectral deferred pressure corrections*, which can in theory obtain arbitrary formal order of accuracy in time. Numerical results demonstrate up to eighth-order accuracy, scenarios where optimal temporal accuracy is attained, and scenarios where order reduction is observed due to time-dependent boundary conditions.

1. Introduction

The motivation of this study is to develop higher-order accurate temporal integration methods for the incompressible Navier–Stokes equations in bounded domains and with possibly time-dependent boundary conditions. In [60,61], the second author presents a framework for applying discontinuous Galerkin (DG) methods to problems of interfacial dynamics in incompressible flows. Although the spatial discretization developed in these papers is high-order accurate, only second-order predictor corrector time stepping schemes are considered therein. The original motivation of this work was to investigate the construction of temporal methods that can match the spatial order of accuracy of these DG methods. The present work focuses on the issues needed to construct such methods for the case of single-phase, incompressible flow in non-trivial physical domains and with time-dependent boundary conditions.

Our approach is based on the method of lines in which the equations are discretized first in space and then an appropriate method is applied to the resulting system of ordinary differential equations (ODEs). As discussed below, there are numerous technical issues that must be addressed to achieve higher-order temporal accuracy for such problems, including the form of the equations to discretize, the choice of time stepping strategy, the imposition of numerical boundary conditions, and the problem of order reduction (possibly due to more than one source). The approach used here is based on a novel combination of a deferred correction method for ODEs and a gauge or auxiliary variable formulation of the equations

* Corresponding author.
E-mail address: rsaye@lbl.gov (R.I. Saye).

https://doi.org/10.1016/j.jcp.2018.08.054
0021-9991/

of motion using projection operators. In these schemes, the gauge variable is used to iteratively update an approximation to the pressure variable via a deferred correction procedure.

In the present context, "higher-order in time" is meant to denote at least fourth-order accuracy. Since the target applications are for incompressible flows with non-negligible viscosity, the stability constraints associated with an explicit treatment of viscous terms must be avoided. On the other hand, it is desirable to also avoid the computational cost associated with fully implicit temporal methods, and hence this paper considers methods that allow an implicit treatment of the (linear) diffusive terms and explicit treatment of the (nonlinear) convective terms (i.e., semi-implicit or implicit-explicit (IMEX) methods). Various IMEX-based linear multistep methods for ODEs have been proposed [7,29,69], however the stability of these approaches degrades significantly for higher-order methods. In particular, the popular choice of using a BDF type method for the implicit terms with an Adams–Bashforth treatment of the explicit terms is unstable for order greater than six. In addition, these methods are not self starting and are cumbersome to use with variable time steps. Many IMEX Runge–Kutta methods have also been proposed (see, e.g., [6,51,42,11,10,12,1]), but methods of order higher than five are not available to our knowledge and are difficult to construct directly due to the enormous number of matching conditions needed (see [51]). There are however two iterative strategies for constructing higher-order ODE methods that have been extended to IMEX formulations. In [20], IMEX-based extrapolation methods are investigated with order up to 12. These methods build higher-order schemes by repeated solution over a time step using lower-order methods (e.g., forward/backward Euler schemes). Similarly, IMEX schemes based on spectral deferred corrections (SDC) were first proposed in [47] (and referred to there as semi-implicit or SISDC methods). These methods apply a forward/backward Euler substepping strategy to the correction equation to iteratively improve the order of accuracy. Here, a variant of the SISDC approach is used, which is explained in section 4.4.

Applying an IMEX method-of-lines approach to the incompressible Navier–Stokes equation is unfortunately not straightforward due to the pressure term and the divergence constraint in the governing equations. Our approach is related to well known projection methods first introduced by Chorin [17] and has similarities to gauge methods [27,59] and to auxiliary variable methods [49,41,57]. Projection and gauge methods are discussed in section 3.

Another issue which arises in the construction of higher-order methods for PDEs with time dependent boundary conditions is that of order reduction. In [16] it is shown that when the classical fourth-order explicit Runge–Kutta method is applied by the method-of-lines to hyperbolic PDEs with time-dependent boundary conditions, the order of accuracy drops from four to two using the most natural choice of numerical boundary conditions. The cause of this type of order reduction is a mismatch in the error between the domain boundary and its interior in the stage values computed during the Runge–Kutta procedure. Various approaches for mitigating this type of order reduction have been proposed for explicit Runge–Kutta schemes [52,15,4,3], but no general remedy for non-linear PDEs has been developed. SISDC methods are also susceptible to order reduction in the presence of time-dependent boundary conditions because they also use lower-order stage values in the solution process [46]. The subject of order reduction in the context of IMEX schemes for the incompressible Navier–Stokes is explored in more detail in section 4.5.

The main results presented here are the formulation and numerical validation of a single-step, semi-implicit or IMEX method for the incompressible Navier–Stokes equations that is similar in style to projection methods yet can in theory obtain arbitrary formal order of accuracy in time. The temporal method is based on SISDC and the key idea is to use an auxiliary variable method in such a way that the pressure is improved each SISDC iteration, hence the method is called *spectral deferred pressure corrections*, or SDPC for short. The updating of the pressure in the iterations drives the auxiliary variable to match the physical velocity so that the boundary conditions for the velocity can be applied to the auxiliary variables directly. Up to eighth-order accuracy in time is demonstrated with numerical examples for the SDPC scheme as well as examples where order-reduction is observed.

The remainder of this paper is organized as follows. In section 2, the incompressible Navier–Stokes equations and several equivalent variations are presented. In section 3, a discussion of how the different formulations presented in section 2 can be discretized in time is presented in a simplified first-order setting. The details of higher-order temporal methods based on collocation schemes and spectral deferred corrections is presented in section 4. The spatial discretization is briefly discussed in section 5. Numerical results are presented in section 6, followed by a discussion of results and outlook for future research in section 7.

2. The Navier–Stokes equations and equivalent formulations

This paper considers flows governed by the incompressible Navier–Stokes equations with constant density and viscosity in a simply connected domain Ω. The velocity $\mathbf{u}$ then satisfies

$$\mathbf{u}_t + \nabla p = -\mathbf{u}\cdot\nabla\mathbf{u} + \nu\nabla^2\mathbf{u} + \mathbf{f}, \tag{2.1}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{2.2}$$

where p is the pressure, ν is the kinematic viscosity, and $\mathbf{f}$ is an explicitly defined forcing term that may depend on time and space but not on the solution. To close these equations, initial and boundary conditions must also be applied. This work examines conditions given on the velocity as

$$\mathbf{u}\cdot n = \mathbf{u}_b\cdot n \ \text{ on } \ \partial\Omega, \tag{2.3}$$

$$\mathbf{u}\cdot \tau = \mathbf{u}_b\cdot \tau \ \text{ on } \ \partial\Omega, \tag{2.4}$$

where n is the outward pointing unit normal vector, and τ represents tangential components orthogonal to n. Alternative boundary conditions such as traction or outflow conditions will be considered in future work. By incompressibility, the boundary conditions must satisfy the compatibility condition

$$\int_{\partial\Omega} \mathbf{u}_b\cdot n \, ds = 0. \tag{2.5}$$

The initial conditions $\mathbf{u}(x, t = 0) = \mathbf{u}_0(x)$ are assumed to be divergence free and satisfy the given boundary conditions. There are several equivalent formulations of these equations that are detailed next. In sections 3 and 4, numerical methods inspired by the different formulations will be discussed.

2.1. Hodge decomposition and projection operators

Central to the numerical methods investigated here are discrete approximations to the Helmholtz–Hodge decomposition of a vector field. Given a smooth vector field $\mathbf{v}$ on Ω, one can write

$$\mathbf{v} = \mathbf{u} + \nabla\phi, \tag{2.6}$$

where $\mathbf{u}$ is divergence free and satisfies a no-flow $\mathbf{u}_b\cdot n = 0$ boundary condition on $\partial\Omega$. The decomposition is unique up to an arbitrary scalar in ϕ and can be constructed by taking the divergence of both sides and applying the normal boundary condition to give the Poisson equation

$$\nabla^2\phi = \nabla\cdot\mathbf{v} \ \text{ in } \ \Omega, \tag{2.7}$$

$$\nabla\phi\cdot n = \mathbf{v}\cdot n \ \text{ on } \ \partial\Omega. \tag{2.8}$$

One can then define the operator $\mathbb{P}(\mathbf{v}) = \mathbf{v} - \nabla\phi$; $\mathbb{P}$ is idempotent and linear, and hence is a projection operator. Specifically, $\mathbb{P}$ is the projection of the vector field $\mathbf{v}$ onto the space of divergence-free flows with no-flow boundary conditions.

One can alter the definition of the projection operator to enforce non-zero normal-component boundary conditions on $\mathbf{u}$: in this case Eqs. (2.7)–(2.8) become

$$\nabla^2\phi_b = \nabla\cdot\mathbf{v} \ \text{ in } \ \Omega, \tag{2.9}$$

$$\nabla\phi_b\cdot n = (\mathbf{v} - \mathbf{u}_b)\cdot n \ \text{ on } \ \partial\Omega. \tag{2.10}$$

One can define $\mathbb{P}_b(\mathbf{v}) = \mathbf{v} - \nabla\phi_b$, so that the resulting velocity is divergence free and has normal boundary condition equal to $\mathbf{u}_b\cdot n$. The operator $\mathbb{P}_b$ is idempotent, but does not satisfy the traditional definition of a projection operator since it is affine instead of linear. Nevertheless, in a slight abuse of terminology both $\mathbb{P}_b$ and $\mathbb{P}$ will be referred to as projections. An important point to make about these projection operators is that the normal component of the resultant divergence-free velocity field is imposed by the definition of the projection, hence the value of the velocity tangential to the boundary cannot be specified *a priori* as well.

An equivalent way of deriving $\mathbb{P}_b$ is through the potential flow corresponding to the prescribed in-flow boundary conditions defined by

$$\nabla^2\eta_b = 0 \ \text{ in } \ \Omega, \tag{2.11}$$

$$\nabla\eta_b\cdot n = \mathbf{u}_b\cdot n \ \text{ on } \ \partial\Omega. \tag{2.12}$$

By construction, $\mathbb{P}_b(\mathbf{v}) = \mathbb{P}(\mathbf{v}) + \nabla\eta_b$ (note $\nabla\eta_b = 0$ when $\mathbf{u}_b\cdot n = 0$). Also, for any gradient ∇q, $\mathbb{P}(\nabla q) = 0$, and, hence $\mathbb{P}_b(\nabla q) = \nabla\eta_b$. Also, for any vector field $\mathbf{v}$ and gradient ∇q

$$\mathbb{P}_b(\mathbf{v} + \nabla q) = \mathbb{P}(\mathbf{v} + \nabla q) + \nabla\eta_b = \mathbb{P}(\mathbf{v}) + \mathbb{P}(\nabla q) + \nabla\eta_b = \mathbb{P}(\mathbf{v}) + \nabla\eta_b = \mathbb{P}_b(\mathbf{v}). \tag{2.13}$$

Finally, it is useful to also define the operator $\mathbb{Q} = \mathbb{I} - \mathbb{P}$ (and the counterpart $\mathbb{Q}_b = \mathbb{I} - \mathbb{P}_b$), where $\mathbb{I}$ is the identity operator. For an arbitrary gradient ∇q,

$$\mathbb{Q}(\nabla q) = \nabla q - \mathbb{P}(\nabla q) = \nabla q. \tag{2.14}$$

Also, it is easy to show that $\mathbb{Q}_b(\nabla q) = \nabla q - \nabla\eta_b$.

2.2. Projection formulation of Navier–Stokes

The left-hand side of Eq. (2.1) appears in the form of the Hodge decomposition. Assume for the moment that the normal boundary condition Eq. (2.3) depends on space but is the same for all time. Applying $\mathbb{P}$ to both sides of Eq. (2.1) gives

$$\mathbf{u}_t = \mathbb{P}(-\mathbf{u}\cdot\nabla\mathbf{u} + \nu\nabla^2\mathbf{u} + \mathbf{f}) \text{ in } \Omega, \tag{2.15}$$

$$\mathbf{u} = \mathbf{u}_b \text{ on } \partial\Omega. \tag{2.16}$$

Any solution of these equations will also satisfy the divergence constraint Eq. (2.2) and the inflow boundary condition. Furthermore

$$\nabla p = \mathbb{Q}(-\mathbf{u}\cdot\nabla\mathbf{u} + \nu\nabla^2\mathbf{u} + \mathbf{f}). \tag{2.17}$$

This projection formulation motivates the class of numerical methods called projection methods described in section 3.1.

By Eq. (2.13), adding a gradient to the quantity acted on by $\mathbb{P}$ in Eq. (2.15) will not change the equations. Therefore

$$\mathbf{u}_t = \mathbb{P}(-\mathbf{u}\cdot\nabla\mathbf{u} - \nabla q + \nu\nabla^2\mathbf{u} + \mathbf{f}) \text{ in } \Omega, \tag{2.18}$$

$$\mathbf{u} = \mathbf{u}_b \text{ on } \partial\Omega, \tag{2.19}$$

is an equivalent formulation of the Navier–Stokes equations for an arbitrary gradient ∇q. In numerical schemes, ∇q may represent an approximation to the pressure gradient. Eq. (2.18) can be derived by applying a projection operator to both sides of the equation

$$\mathbf{u}_t + \nabla p - \nabla q = -\mathbf{u}\cdot\nabla\mathbf{u} - \nabla q + \nu\nabla^2\mathbf{u} + \mathbf{f}. \tag{2.20}$$

Clearly, it is also true that

$$\nabla p - \nabla q = \mathbb{Q}(-\mathbf{u}\cdot\nabla\mathbf{u} - \nabla q + \nu\nabla^2\mathbf{u} + \mathbf{f}). \tag{2.21}$$

In the case of time-dependent boundary conditions, the operators $\mathbb{P}_b$ and $\mathbb{Q}_b$ can be used to derive similar projection formulations. The issue of time-dependent boundary conditions for projection methods will be discussed in more detail in section 3.

2.3. Gauge formulations

Additional formulations of the Navier–Stokes equations can be derived by introducing a variable that differs from the velocity by the gradient of a scalar. Such formulations have appeared many times and with differing names dating back to the 1970s [59]; the reader is referred to the work of Russo and Smereka [58], which provides a general framework for describing different gauge formulations (albeit in the case of the incompressible Euler equations).

Following the terminology in [27], the *gauge variable* formulation for the auxiliary variable $\mathbf{m} = \mathbf{u} + \nabla\chi$ is introduced, where χ is the gauge variable[1] and $\mathbf{u}$ is a solution to Eqs. (2.1)–(2.2). Specifically,

$$\mathbf{m}_t = -\mathbf{u}\cdot\nabla\mathbf{u} + \nu\nabla^2\mathbf{m} + \mathbf{f}, \tag{2.22}$$

$$\mathbf{u} = \mathbb{P}(\mathbf{m}) = \mathbf{m} - \nabla\chi. \tag{2.23}$$

The divergence constraint (2.2) is here replaced by defining $\mathbf{u}$ as the projection of $\mathbf{m}$. For clarity, consider for the moment no-flow and no-slip boundary conditions. By the definition of the projection operator $\mathbb{P}$ and Eq. (2.23), the boundary condition on the normal component of $\mathbf{m}$ is equal to $\nabla\chi\cdot n$, and this boundary condition can be prescribed. Additionally, the tangential components of $\mathbf{m}$ and $\nabla\chi$ must match at the boundary, but this is not solely enforced by the projection. Hence, the slip boundary condition for $\mathbf{u}$ in Eq. (2.4) is replaced here by a coupling of $\mathbf{m}$ and $\nabla\chi$ at the boundary. This boundary condition will play an important role in the discussion of numerical methods below. In summary, boundary conditions which are consistent with no-flow and no-slip for the velocity take the form

$$\mathbf{m}\cdot n = 0 \text{ on } \partial\Omega, \tag{2.24}$$

$$(\mathbf{m} - \nabla\chi)\cdot\tau = 0 \text{ on } \partial\Omega. \tag{2.25}$$

These boundary conditions must be modified in the case that the velocity satisfies inhomogeneous boundary conditions. Again, there is some freedom in terms of the normal boundary condition on $\mathbf{m}$ so long as the projection operator is defined

[1] Note that this definition of *gauge* differs from that in [58].

correctly. Here, the normal component of $\mathbf{m}$ is assigned the boundary condition of the velocity $\mathbf{u}_b \cdot n$, which implies a homogeneous Neumann condition $\nabla\chi \cdot n = 0$. The tangential boundary condition for $\mathbf{m}$ must take into account the tangential component of the velocity boundary condition, giving

$$\mathbf{m}\cdot n = \mathbf{u}_b \cdot n \ \text{ on } \ \partial\Omega, \tag{2.26}$$

$$(\mathbf{m} - \nabla\chi)\cdot\tau = \mathbf{u}_b \cdot \tau \ \text{ on } \ \partial\Omega. \tag{2.27}$$

The pressure term does not appear in Eq. (2.22), but is related to χ by

$$p = \chi_t - \nu\nabla^2\chi. \tag{2.28}$$

As emphasized in [58], one can add an arbitrary gradient term to Eq. (2.22) without changing the equivalence of the gauge equations to the Navier–Stokes equations. Hence a more general gauge formulation for a given prescribed function q is

$$\mathbf{u}_t^* = -\mathbf{u}\cdot\nabla\mathbf{u} - \nabla q + \nu\nabla^2\mathbf{u}^* + \mathbf{f}, \tag{2.29}$$

$$\mathbf{u} = \mathbb{P}(\mathbf{u}^*) = \mathbf{u}^* - \nabla\psi. \tag{2.30}$$

The change in notation in the presence of q consisting of replacing $\mathbf{m}$ by $\mathbf{u}^*$ and χ by ψ is to avoid confusion in comparing numerical methods based on these formulations in the next section. Following the terminology in [41], the methods based on these equations are referred to as *auxiliary variable* methods.

If $q \equiv 0$ in Eq. (2.29), then $\mathbf{u}^* = \mathbf{m}$ and $\nabla\chi = \nabla\psi$. However when $q \neq 0$, $\mathbb{P}(\mathbf{u}^*) = \mathbb{P}(\mathbf{m}) = \mathbf{u}$ satisfies the Navier–Stokes equations, but $\mathbf{u}^* \neq \mathbf{m}$ and $\nabla\chi \neq \nabla\psi$. Also, the relationship between ψ and the pressure is now

$$p = q + \psi_t - \nu\nabla^2\psi. \tag{2.31}$$

As mentioned above, in a numerical scheme, q may be chosen to be an approximation to the pressure. Clearly, if q could be given the value of the pressure (for all time), then $\psi \equiv 0$ and $\mathbf{u}^* = \mathbf{u}$. In general, the closer ∇q is to ∇p, the smaller $\nabla\psi$ becomes. This fact impacts the imposition of boundary conditions in the numerical methods since the analog to Eq. (2.27),

$$(\mathbf{u}^* - \nabla\psi)\cdot\tau = \mathbf{u}_b\cdot\tau \ \text{ on } \ \partial\Omega, \tag{2.32}$$

must be enforced at the boundary.

2.4. Other formulations

The formulations of the Navier–Stokes equations described in the previous sections all employ the projection operator to enforce the divergence constraint given by Eq. (2.2), and the study of numerical methods based on these formulations is the subject of this work. For completeness, here we mention three prominent alternative formulations. First, an equivalent system often referred to as the *pressure Poisson formulation* (e.g., [33]) can be derived which replaces the divergence constraint with a Poisson equation for the pressure with suitable boundary conditions. A second classical approach involves deriving an evolution equation for the vorticity or curl of the velocity, and many numerical methods based on both a Lagrangian and Eulerian discretization of vorticity formulations have appeared in the literature (see, e.g., [21]). Finally, formulations based on the evolution of the stream function can be derived, and recent work on direct discretizations of this formulation based on integral equations has appeared [40]. A proper review of these formulations is beyond the scope of this work.

3. First-order temporal integration strategies

In the previous section, different, but analytically equivalent, formulations of the Navier–Stokes equations are presented. Although of some interest in the abstract, our focus here is on the numerical approximation of these equations, and the natural question to ask is whether there is some intrinsic advantage of one formulation over the other in terms of numerical approximation. In this section, we discuss the issues related to the temporal evolution of these equations and how the different formulations affect the numerical boundary conditions. The presentation in this section leaves all spatial derivatives in the continuous form. As discussed in section 1, the focus is on semi-implicit temporal methods that treat the viscous terms in the evolution equations implicitly for stability and the nonlinear advective terms explicitly.

The key difficulties in constructing higher-order semi-implicit projection and gauge methods can be illuminated by considering first-order time stepping methods applied to the simpler case of the Stokes equations, with no-flow and no-slip boundary conditions, given by

$$\mathbf{u}_t + \nabla p = \nabla^2\mathbf{u} + \mathbf{f}(t) \ \text{ in } \ \Omega, \tag{3.1}$$

$$\nabla\cdot\mathbf{u} = 0 \quad \text{in } \ \Omega, \tag{3.2}$$

$$\mathbf{u} = 0 \ \text{ on } \ \partial\Omega. \tag{3.3}$$

The first-order methods discussed below also form the basis of the higher-order spectral deferred correction methods introduced in section 4.4.

For simplicity, assume a uniform time step $\Delta t = t_{n+1} - t_n$, and let the numerical approximation of a given quantity at time t_n be denoted by a subscript. A first-order method for the Stokes equations (3.1)–(3.3) is given by

$$\frac{\mathbf{u}_{n+1} - \mathbf{u}_n}{\Delta t} + \nabla p_{n+1} = \nabla^2 \mathbf{u}_{n+1} + \mathbf{f}_n \text{ in } \Omega, \tag{3.4}$$

$$\nabla \cdot \mathbf{u}_{n+1} = 0 \text{ in } \Omega, \tag{3.5}$$

$$\mathbf{u}_{n+1} = 0 \text{ on } \partial\Omega. \tag{3.6}$$

This implicit equation couples the unknowns $\mathbf{u}_{n+1}$ and ∇p_{n+1}. Since $\mathbf{f}$ is an explicit function in time, it is possible to use $\mathbf{f}_{n+1}$ in Eq. (3.4), but the current form matches the methods discussed later for the Navier–Stokes equations where the advective terms are computed explicitly.

The main drawback of this direct approach is that the full system for $\mathbf{u}_{n+1}$ and ∇p_{n+1} must be solved at once. The methods discussed below are largely motivated by the desire to replace this coupled system with two simpler linear systems. The following sections discuss projection, gauge, and auxiliary variable methods in the context of first-order temporal methods for Eqs. (3.1)–(3.3).

3.1. Projection methods

Projection methods, introduced by Chorin [17], are widely used and studied (see for example the review [35]), and they form the basis of methods for more complex flows than incompressible Navier–Stokes (see, e.g., [9,45,19,22,62,65,50]). A first-order projection method for the Stokes equations replaces the coupled system in Eqs. (3.4) and (3.5) with two sequential implicit equations for which efficient numerical methods are well established. Roughly speaking, the first step approximates Eq. (3.4) without regard to the divergence constraint Eq. (3.5), and the second involves a discrete projection operator to enforce Eq. (3.5). Hence projection methods have traditionally also been referred to as fractional step methods, although this correspondence is less obvious for projection methods with higher-order temporal accuracy.

The left-hand side of Eq. (3.4) is in the form of a Hodge decomposition, hence applying the projection operator to both sides gives

$$\frac{\mathbf{u}_{n+1} - \mathbf{u}_n}{\Delta t} = \mathbb{P}(\nabla^2 \mathbf{u}_{n+1} + \mathbf{f}_n) \tag{3.7}$$

and likewise

$$\nabla p_{n+1} = \mathbb{Q}(\nabla^2 \mathbf{u}_{n+1} + \mathbf{f}_n). \tag{3.8}$$

Since only the normal boundary condition is enforced by the projection operator, these two equations are not sufficient to give an equivalent form of Eqs. (3.4)–(3.6) without including the additional boundary condition on the tangential component of Eq. (3.3). Assuming $\nabla \cdot \mathbf{u}_n = 0$ and using the linearity of $\mathbb{P}$, an equivalent form is given by

$$\mathbf{u}_{n+1} = \mathbb{P}(\mathbf{u}_n + \Delta t(\nabla^2 \mathbf{u}_{n+1} + \mathbf{f}_n)) \text{ in } \Omega, \tag{3.9}$$

$$\Delta t \nabla p_{n+1} = \mathbb{Q}(\mathbf{u}_n + \Delta t(\nabla^2 \mathbf{u}_{n+1} + \mathbf{f}_n)) \text{ in } \Omega, \tag{3.10}$$

$$\mathbf{u}_{n+1} = 0 \text{ on } \partial\Omega. \tag{3.11}$$

The difficulty in solving these equations numerically is that the right-hand side of Eq. (3.9) couples the implicit Poisson equation implied by the projection operator with the implicit diffusion type equation for $\mathbf{u}_{n+1}$ (and the boundary condition Eq. (3.11)). The 1968 paper of Chorin [17] avoids this difficulty by replacing Eq. (3.9) with two equations

$$\mathbf{u}^* = \mathbf{u}_n + \Delta t(\nabla^2 \mathbf{u}^* + \mathbf{f}_n), \tag{3.12}$$

$$\mathbf{u}_{n+1} = \mathbb{P}(\mathbf{u}^*) = \mathbf{u}^* - \nabla\phi, \tag{3.13}$$

where by definition ϕ is the solution to the Poisson equation

$$\nabla^2 \phi = \nabla \cdot \mathbf{u}^* \text{ in } \Omega, \tag{3.14}$$

$$\nabla\phi \cdot n = \mathbf{u}^* \cdot n \text{ on } \partial\Omega. \tag{3.15}$$

Both of these equations can be readily solved using standard methods. The actual implementation in [17] of Eq. (3.12) is done in a dimensional splitting manner and is hence slightly different. Here the intermediate variable $\mathbf{u}^*$ is computed by an implicit solution of a diffusion type equation and then projected to yield the updated velocity $\mathbf{u}_{n+1}$, which requires the solution of a Poisson problem. This approach resembles a fractional step approach where Eq. (2.1) is solved first followed by Eq. (2.2).

Eq. (3.12) requires boundary conditions, and the proper choice of these conditions has been a source of contention in the literature. One argument is that since $\mathbf{u}^*$ is an approximation to $\mathbf{u}_{n+1}$, the boundary conditions for $\mathbf{u}^*$ should be the given boundary conditions on the velocity, namely $\mathbf{u}^* = 0$. Some papers in the literature (e.g., [26] Eq. (2.3), [35] Eq. (3.1)) erroneously assert that these are the boundary conditions used in the original Chorin paper [17] when in fact the boundary conditions in [17] are an approximation of

$$\mathbf{u}^* = \Delta t \nabla p_{n+1} \text{ on } \partial\Omega. \tag{3.16}$$

To understand this boundary condition heuristically, replace $\mathbf{u}^*$ on the right-hand side of Eq. (3.12) with $\mathbf{u}_{n+1} + \nabla\phi$,

$$\mathbf{u}_{n+1} + \nabla\phi = \mathbf{u}_n + \Delta t(\nabla^2 \mathbf{u}^* + \mathbf{f}_n), \tag{3.17}$$

from which it is clear that $\nabla\phi$ is an approximation to $\Delta t \nabla p_{n+1}$ (also implied by Eq. (3.10)). Since $\nabla\phi$ (or equivalently $\Delta t \nabla p_{n+1}$) is not yet known at the boundary when Eq. (3.12) is solved, the most reasonable approximation is to use the previous value $\Delta t \nabla p_n$.

One confusing aspect of the choice of boundary conditions in Chorin's method is that the normal component of the pressure gradient does not change over the time step since

$$\Delta t \nabla p_{n+1} \cdot n = \nabla\phi \cdot n = \mathbf{u}^* \cdot n = \Delta t \nabla p_n \cdot n \text{ on } \partial\Omega. \tag{3.18}$$

This issue will be revisited in the discussion of gauge methods below, but note that substituting $\mathbf{u}^* = \mathbf{u}_{n+1} + \nabla\phi$ into Eq. (3.12) gives

$$\mathbf{u}_{n+1} + \nabla\phi = \mathbf{u}_n + \Delta t(\nabla^2 \mathbf{u}_{n+1} + \nabla^2 \nabla\phi + \mathbf{f}_n) \tag{3.19}$$

or, upon rearranging terms,

$$\mathbf{u}_{n+1} + \nabla(\phi - \Delta t \nabla^2 \phi) = \mathbf{u}_n + \Delta t(\nabla^2 \mathbf{u}_{n+1} + \mathbf{f}_n). \tag{3.20}$$

This suggests an alternative definition of the pressure

$$\nabla p_{n+1} = \nabla(\phi/\Delta t - \nabla^2 \phi). \tag{3.21}$$

This observation dates back at least to Kim and Moin [43] (see page 310 therein). Again, this issue will be revisited below.

The original method of Chorin has been modified and extended many times. One defining characteristic of projection methods in general is whether the equation for $\mathbf{u}^*$ contains a pressure gradient approximation. In continuous variables, this difference is manifest in the difference between Eq. (2.15) and Eq. (2.18). Since $\mathbb{P}$ annihilates gradients, an equivalent form of Eq. (3.9) is

$$\mathbf{u}_{n+1} = \mathbb{P}(\mathbf{u}_n + \Delta t(-\nabla p_n + \nabla^2 \mathbf{u}_{n+1} + \mathbf{f}_n)). \tag{3.22}$$

Following the logic above for Chorin's original method suggests the method

$$\mathbf{v}^* = \mathbf{u}_n + \Delta t(-\nabla p_n + \nabla^2 \mathbf{v}^* + \mathbf{f}_n), \tag{3.23}$$

$$\mathbf{u}_{n+1} = \mathbb{P}(\mathbf{v}^*) = \mathbf{v}^* - \nabla\phi_c, \tag{3.24}$$

where the new variables $\mathbf{v}^*$ and $\nabla\phi_c$ are introduced to differentiate the solutions from $\mathbf{u}^*$ and $\nabla\phi$. As far as the authors are aware, this formulation was first put forth by Goda [32]. The inclusion of the pressure term has two ramifications, namely the form of the boundary condition for $\mathbf{v}^*$ and the need for a procedure for computing ∇p_{n+1}. Again following the logic of Chorin's method, reorganize Eq. (3.23) using the Hodge decomposition to replace $\mathbf{v}^*$ on the left-hand to yield

$$\mathbf{u}_{n+1} + \nabla\phi_c + \Delta t \nabla p_n = \mathbf{u}_n + \Delta t(\nabla^2 \mathbf{v}^* + \mathbf{f}_n). \tag{3.25}$$

This suggests the pressure update or increment

$$\nabla p_{n+1} = \nabla p_n + \nabla\phi_c/\Delta t. \tag{3.26}$$

It is important to remember that the term $\nabla\phi_c$ here is different in the pressure increment formulation than in the original Chorin method, and in fact now $\nabla\phi_c = O(\Delta t^2)$ while $\nabla\phi = O(\Delta t)$. Hence the boundary condition for $\mathbf{v}^* = 0$ is consistent with Chorin's method. Following the argument used to derive Eq. (3.21), the pressure increment form can be modified by adding the lower-order $\nabla^2\phi_c$ term

$$\nabla p_{n+1} = \nabla p_n + \nabla(\phi_c/\Delta t - \nabla^2 \phi_c). \tag{3.27}$$

Regardless of which of these first-order formulations are used, the updated velocity $\mathbf{u}_{n+1} = \mathbf{u}^* - \nabla\phi$ or $\mathbf{u}_{n+1} = \mathbf{v}^* - \nabla\phi_c$ will not be the same as that resulting from solving the Stokes formulation given by Eqs. (3.4)–(3.6), which is evident because there is no guarantee that the tangential boundary condition $\mathbf{u}_{n+1}\cdot\tau = 0$ is enforced. This obviously cannot be enforced numerically since $\nabla\phi$ is not known when $\mathbf{u}^*$ is computed.

However, consider the following iteration, where k denotes the iteration number:

$$\mathbf{u}^{*,k} = \mathbf{u}_n + \Delta t(\nabla^2\mathbf{u}^{*,k} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.28}$$

$$\mathbf{u}^{*,k} = \nabla\phi^{k-1} \text{ on } \partial\Omega. \tag{3.29}$$

Here $\nabla\phi^k$ is computed during the projection step $\mathbf{u}^k_{n+1} = \mathbb{P}(\mathbf{u}^{*,k})$ by

$$\nabla^2\phi^k = \nabla\cdot\mathbf{u}^{*,k} \text{ in } \Omega, \tag{3.30}$$

$$\nabla\phi^k\cdot n = 0 \text{ on } \partial\Omega. \tag{3.31}$$

If this method converges to $\mathbf{u}^*$ and $\nabla\phi$, then $\mathbf{u}_{n+1} = \mathbf{u}^* - \nabla\phi$ and $\nabla p_{n+1} = \nabla\phi/\Delta t - \nabla^2\nabla\phi$ would also satisfy Eqs. (3.4)–(3.6), including the correct boundary condition $\mathbf{u}_{n+1} = 0$.

Similarly, an iterative form of the pressure correction projection method is given by

$$\mathbf{v}^{*,k} = \mathbf{u}_n + \Delta t(-\nabla p^{k-1}_{n+1} + \nabla^2\mathbf{v}^{*,k} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.32}$$

$$\mathbf{v}^{*,k} = 0 \text{ on } \partial\Omega, \tag{3.33}$$

with

$$\nabla^2\phi^k_c = \nabla\cdot\mathbf{v}^{*,k} \text{ in } \Omega, \tag{3.34}$$

$$\nabla\phi^k_c\cdot n = 0 \text{ on } \partial\Omega \tag{3.35}$$

and

$$\nabla p^k_{n+1} = \nabla p^{k-1}_{n+1} + \nabla\phi^k_c/\Delta t - \nabla^2\nabla\phi^k_c. \tag{3.36}$$

The value for the initial pressure p^0_{n+1} will be discussed below. If this iteration converges to $\mathbf{v}^*$ and ∇p_{n+1}, then $\nabla\phi^k_c \to 0$ as $k\to\infty$, so that $\mathbb{P}(\mathbf{v}^*) = \mathbf{v}^*$. Hence $\mathbf{u}_{n+1} = \mathbf{v}^*$ and ∇p_{n+1} satisfy Eqs. (3.4)–(3.6).

Furthermore, consider the variable defined by $\mathbf{v}^k = \mathbf{u}^{*,k} - \nabla\phi^{k-1}$. Substituting into Eqs. (3.28)–(3.29) gives

$$\mathbf{v}^k = \mathbf{u}_n + \Delta t(-\nabla\phi^{k-1}/\Delta t + \nabla^2\nabla\phi^{k-1} + \nabla^2\mathbf{v}^k + \mathbf{f}_n) \text{ in } \Omega, \tag{3.37}$$

$$\mathbf{v}^k = 0 \text{ on } \partial\Omega, \tag{3.38}$$

which is equivalent to the iteration Eqs. (3.32)–(3.33) with $\nabla p^{k-1}_{n+1} = \nabla\phi^{k-1}/\Delta t - \nabla^2\nabla\phi^{k-1}$. Hence if $\nabla p^0_{n+1} = \nabla\phi^0/\Delta t - \nabla^2\nabla\phi^0$, then for all $k > 1$, $\mathbb{P}(\mathbf{u}^{*,k}) = \mathbb{P}(\mathbf{v}^{*,k})$, and the two iterative formulations are equivalent analytically.

Viewing a given projection method as an iterative step in solving the Stokes problem has appeared in the literature before (see, e.g., [8,14,34,31]). Obviously, performing additional iterations of the projection procedure requires more work than just a single iteration, and one iteration is sufficient for first-order accuracy of the velocity. The main reason for introducing this iteration here is that the spectral deferred correction methods introduced in section 4.4 can be considered as higher-order variants of this type of iteration.

3.2. Gauge methods

To our knowledge, E and Liu were the first to propose numerical methods for the specific form of the gauge equations given in section 2.3. In [24], a finite difference method based on these equations using explicit Runge–Kutta time stepping is introduced. Second-order semi-implicit projection and gauge methods are compared in [13], and E and Liu present gauge methods based on IMEX linear-multistep temporal methods in [25,27].

A first-order gauge method for the Stokes equations takes the form

$$\mathbf{m}_{n+1} = \mathbf{m}_n + \Delta t(\nabla^2\mathbf{m}_{n+1} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.39}$$

$$\mathbf{m}_{n+1} - \nabla\chi_{n+1} = 0 \text{ on } \partial\Omega, \tag{3.40}$$

where $\nabla\chi_{n+1}$ is again defined through the Poisson problem

$$\nabla^2\chi_{n+1} = \nabla\cdot\mathbf{m}_{n+1} \text{ in } \Omega, \tag{3.41}$$

$$\nabla\chi_{n+1}\cdot n = 0 \text{ on } \partial\Omega. \tag{3.42}$$

The velocity is defined through the projection operator $\mathbf{u}_{n+1} = \mathbb{P}(\mathbf{m}_{n+1}) = \mathbf{m}_{n+1} - \nabla\chi_{n+1}$. The key difference between the gauge method and projection methods is that $\mathbf{m}$ is allowed to evolve distinctly from $\mathbf{u}$ over multiple time steps.

Note that the boundary condition in Eq. (3.40) couples with the Poisson equation (3.41)–(3.42) since the tangential component of $\nabla\chi_{n+1}$ cannot be specified. As in projection methods, this can be mitigated by approximating the tangential component of the boundary condition yielding the scheme

$$\mathbf{m}_{n+1} = \mathbf{m}_n + \Delta t(\nabla^2\mathbf{m}_{n+1} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.43}$$

$$\mathbf{m}_{n+1} - \nabla\tilde{\chi}_{n+1} = 0 \text{ on } \partial\Omega, \tag{3.44}$$

with

$$\nabla^2\chi_{n+1} = \nabla\cdot\mathbf{m}_{n+1} \text{ in } \Omega, \tag{3.45}$$

$$\nabla\chi_{n+1}\cdot n = 0 \text{ on } \partial\Omega. \tag{3.46}$$

In [27], the approximate boundary condition given by $\nabla\tilde{\chi}_{n+1}$ is computed using extrapolation in time. The simplest approximation is $\nabla\tilde{\chi}_{n+1} = \nabla\chi_n$.

As was done for projection methods, the unknown boundary condition $\nabla\tilde{\chi}_{n+1}$ could be improved by iteration. Iterative schemes like this have been used before to implement Stokes solvers using gauge variables [31]. Recalling the operator $\mathbb{Q} = \mathbb{I} - \mathbb{P}$, an equivalent expression is $\nabla\chi_{n+1} = \mathbb{Q}(\mathbf{m}_{n+1})$, so the boundary condition in Eq. (3.40) can be written

$$\mathbf{m}_{n+1} - \mathbb{Q}(\mathbf{m}_{n+1}) = 0 \text{ on } \partial\Omega. \tag{3.47}$$

Then, if the iteration

$$\mathbf{m}^k_{n+1} = \mathbf{m}_n + \Delta t(\nabla^2\mathbf{m}^k_{n+1} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.48}$$

$$\mathbf{m}^k_{n+1} = \mathbb{Q}(\mathbf{m}^{k-1}_{n+1}) \text{ on } \partial\Omega \tag{3.49}$$

converges to $\mathbf{m}_{n+1}$, the solution is equivalent to Eqs. (3.39)–(3.40).

The pressure term does not appear in the temporal method, but the pressure could be computed by discretizing Eq. (2.28). For example,

$$p_{n+1} = (\chi_{n+1} - \chi_n)/\Delta t - \nabla^2\chi_{n+1}. \tag{3.50}$$

In fact, using $\mathbf{m}^k_{n+1} = \mathbf{u}^k_{n+1} + \nabla\chi^k_{n+1}$, one can rewrite Eqs. (3.48)–(3.49) as

$$\mathbf{u}^k_{n+1} = \mathbf{u}_n + \Delta t(-\nabla(\chi^k_{n+1} - \chi_n)/\Delta t + \nabla^2\nabla\chi^k_{n+1} + \nabla^2\mathbf{u}^k_{n+1} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.51}$$

$$\mathbf{u}^k_{n+1} = \nabla\chi^k_{n+1} - \nabla\chi^{k-1}_{n+1} \text{ on } \partial\Omega. \tag{3.52}$$

This shows that the preceding gauge method iteration is implicitly computing both a velocity with consistent boundary conditions and a pressure gradient consistent with that velocity.

Finally, define $\mathbf{v}^k_{n+1} = \mathbf{m}^k_{n+1} - \nabla\chi^{k-1}_{n+1}$, then substituting into Eqs. (3.48)–(3.49) gives

$$\mathbf{v}^k_{n+1} = \mathbf{v}_n + \Delta t(-\nabla(\chi^{k-1}_{n+1} - \chi_n)/\Delta t + \nabla^2\nabla\chi^{k-1}_{n+1} + \nabla^2\mathbf{v}^k_{n+1} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.53}$$

$$\mathbf{v}^k_{n+1} = 0 \text{ on } \partial\Omega. \tag{3.54}$$

These equations are equivalent to Eqs. (3.37)–(3.38), and hence this iterative formulation of the first-order gauge method gives exactly the same velocity as the projection methods introduced above.

3.3. Auxiliary variable methods

In this section, methods based on auxiliary variables are discussed that are similar in style to those presented in [41,49] and that share similarities to both projection and gauge methods. In the auxiliary variable approach, the gauge formulation given in Eq. (2.29) is used where a pressure approximation is included as the term ∇q. The pressure is then updated after each time step using some approximation to Eq. (2.31). Unlike the gauge method, the auxiliary variable $\mathbf{u}^*$ is reset to the computed velocity after each time step.

A first-order method for the Stokes equations for auxiliary variables is

$$\mathbf{u}^*_{n+1} = \mathbf{u}_n + \Delta t(-\nabla q_n + \nabla^2\mathbf{u}^*_{n+1} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.55}$$

$$\mathbf{u}^*_{n+1} - \nabla\psi_{n+1} = 0 \text{ on } \partial\Omega, \tag{3.56}$$

with

$$\mathbf{u}_{n+1} = \mathbb{P}(\mathbf{u}^*_{n+1}) = \mathbf{u}^*_{n+1} - \nabla\psi_{n+1}. \tag{3.57}$$

As with the gauge method, the tangential part of the boundary condition (3.56) is coupled to $\mathbb{P}(\mathbf{u}^*_{n+1})$ and hence must be approximated. If ∇q_n is chosen to be ∇p_n, then this is equivalent to the pressure correction projection method given by Eq. (3.22), where $\nabla\psi_{n+1}$ in Eq. (3.56) is approximated by zero.

Proceeding as before, an iterative first-order temporal method for the auxiliary variable method is

$$\mathbf{v}^{*,k}_{n+1} = \mathbf{u}_n + \Delta t(-\nabla q^{k-1}_n + \nabla^2\mathbf{v}^{*,k}_{n+1} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.58}$$

$$\mathbf{v}^{*,k}_{n+1} = 0 \text{ on } \partial\Omega, \tag{3.59}$$

where q^k_n is updated by discretizing Eq. (2.31),

$$q^k_n = q^{k-1}_n + \psi^k_{n+1}/\Delta t - \nabla^2\psi^k_{n+1}. \tag{3.60}$$

This is equivalent to the iterative pressure update projection method given in Eqs. (3.32)–(3.36).

Alternatively, one could follow the approach for gauge methods where instead of building an explicit approximation to the pressure gradient, the boundary conditions are changed instead in the iteration

$$\mathbf{v}^{*,k}_{n+1} = \mathbf{u}_n + \Delta t(-\nabla q_n + \nabla^2\mathbf{v}^{*,k}_{n+1} + \mathbf{f}_n) \text{ in } \Omega, \tag{3.61}$$

$$\mathbf{u}^{*,k}_{n+1} = \nabla\psi^{k-1}_{n+1} \text{ on } \partial\Omega. \tag{3.62}$$

Defining $\mathbf{v}^k = \mathbf{v}^{*,k} - \nabla\psi^{k-1}_{n+1}$ and substituting into Eqs. (3.61)–(3.62) gives

$$\mathbf{v}^k = \mathbf{u}_n + \Delta t(-\nabla q_n - \nabla\psi^{k-1}_{n+1}/\Delta t + \nabla^2\nabla\psi^{k-1}_{n+1} + \nabla^2\mathbf{v}^k + \mathbf{f}_n) \text{ in } \Omega, \tag{3.63}$$

$$\mathbf{v}^k = \nabla\psi^{k-1}_{n+1} \text{ on } \partial\Omega, \tag{3.64}$$

which is equivalent to Eqs. (3.53)–(3.54).

3.4. Summary of first-order methods for the stokes equations

In the previous three sections it is shown that first-order projection methods, gauge methods, and auxiliary variable methods can be configured to give mathematically equivalent velocities when spatial discretization is ignored. The three approaches also share the common feature that the tangential velocity condition must be approximated during the discretization of the momentum equation. Two types of iterative procedures for imposing the tangential boundary conditions are also presented, one based on improving the approximation to the tangential boundary condition and a second based on improving the approximation to the pressure term so that a zero tangential boundary condition is consistent. In the first-order, semi-discrete context, both iterations were shown to be equivalent. Hence in this context, there is no benefit in choosing one type of discretization over another. As discussed below, the situation changes when considering higher-order temporal methods and spatial discretization.

4. Extensions to higher-order temporal integration

In this section, higher-order temporal methods based on Gauss collocation schemes and spectral deferred corrections are discussed. In both cases, a general overview of the method is provided followed by details of how the method is applied to the Navier–Stokes equations. A discussion of order reduction follows at the end of the section.

4.1. Gauss collocation methods

Consider the generic ODE

$$y' = f(y) \tag{4.1}$$

on the time interval $[t_n, t_{n+1}]$ with initial condition $y(t_n) = y_n$. To define a collocation method, points in the interval $[t_n, t_{n+1}]$ are chosen, denoted here by t_m for $m = 0, \ldots, M$. The collocation formulation is derived by replacing the integral in the exact expression

$$y(t_m) = y(t_n) + \int_{t_n}^{t_m} f(y(\tau))\, d\tau, \tag{4.2}$$

with a numerical quadrature

$$y_m = y_n + \Delta t \sum_{j=0}^{M} w_{m,j} f(y_j), \qquad m = 1, \ldots, M, \tag{4.3}$$

where y_m approximates the solution $y(t_m)$.

The first-order methods for the Stokes equations based on backward Euler presented in the previous section are equivalent to a Gauss–Radau collocation scheme using only one quadrature node. In the numerical experiments presented below, Gauss–Lobatto quadrature nodes of up to order eight are used so that $t_0 = t_n$ and $t_M = t_{n+1}$, and the weights $w_{m,j}$ correspond to the Lobatto IIIA quadrature rules. (See [39] for a list of nodes and weights.) Such collocation formulas give A-stable methods of order $2M$ [36] for initial value ODEs. The principle drawback of collocation methods is that they are fully implicit as the solution of Eq. (4.3) couples the M unknown values y_m together. An effective method for solving these equations based on spectral deferred corrections is described in section 4.3.

4.2. Collocation methods for the Navier–Stokes equations

The specific form of collocation methods when applied to gauge and auxiliary variable formulations of the Navier–Stokes equations is now considered. As in the case of first-order methods presented in section 3, the equivalence of different formulations based on projection and gauge methods is demonstrated. Again discretizing only in time, a generic collocation method for the Navier–Stokes Eqs. (2.1)–(2.4) takes the form

$$\mathbf{u}_m = \mathbf{u}_n + \Delta t \sum_{j=0}^{M} w_{m,j} \left(-\mathbf{u}_j \cdot \nabla \mathbf{u}_j - \nabla p_j + \nu \nabla^2 \mathbf{u}_j + \mathbf{f}_j \right) \text{ in } \Omega, \tag{4.4}$$

$$\nabla \cdot \mathbf{u}_m = 0 \text{ in } \Omega, \tag{4.5}$$

$$\mathbf{u}_m = \mathbf{u}_b(t_m) \text{ on } \partial\Omega, \tag{4.6}$$

for $m = 1, \ldots, M$. Clearly the collocation formulation couples $\mathbf{u}_m$ and ∇p_m at each quadrature node with the additional coupling of the boundary conditions and divergence constraint for $\mathbf{u}_m$.

A subtle issue regarding the collocation formulations used here is that the pressure at the collocation nodes is not defined uniquely. This is due to the fact that the quadrature matrix based on the Lobatto IIIA nodes (defined by the weights $w_{m,j}$) has a non-trivial null space [37]. Hence one can write a more generic form of Eq. (4.4) as

$$\mathbf{u}_m = \mathbf{u}_n - I_{\nabla p,m} + \Delta t \sum_{j=0}^{M} w_{m,j} \left(-\mathbf{u}_j \cdot \nabla \mathbf{u}_j + \nu \nabla^2 \mathbf{u}_j + \mathbf{f}_j \right), \tag{4.7}$$

where

$$I_{\nabla p,m} \approx \int_{t_n}^{t_m} \nabla p(t)\, dt \tag{4.8}$$

is the unique gradient that enforces the divergence constraint (4.5). The non-uniqueness of pressure values at the quadrature nodes has implications in evaluating the order of accuracy of the pressure approximation in the numerical methods examined later. Numerical experiments in section 6 demonstrate that only the error in the integral of the pressure is expected to achieve the optimal order of accuracy while individual values ∇p_m can be less accurate.

Applying the collocation method to the gauge variable formulation given by Eqs. (2.22)–(2.23) and (2.26)–(2.27) yields

$$\mathbf{m}_m = \mathbf{m}_n + \Delta t \sum_{j=0}^{M} w_{m,j} \left(-\mathbf{u}_j \cdot \nabla \mathbf{u}_j + \nu \nabla^2 \mathbf{m}_j + \mathbf{f}_j \right) \text{ in } \Omega, \tag{4.9}$$

$$\mathbf{m}_m - \nabla \chi_m = \mathbf{u}_b(t_m) \text{ on } \partial\Omega, \tag{4.10}$$

for $m = 1, \ldots, M$, with

$$\mathbf{u}_m = \mathbb{P}(\mathbf{m}_m) = \mathbf{m}_m - \nabla \chi_m \text{ in } \Omega, \tag{4.11}$$

$$\nabla^2 \chi_m = \nabla \cdot \mathbf{m}_m \text{ in } \Omega, \tag{4.12}$$

$$\nabla \chi_m \cdot n = 0 \text{ on } \partial\Omega, \tag{4.13}$$

for $m = 0, \ldots, M$.

Rewriting these equations by substituting $\mathbf{u}_m = \mathbf{m}_m - \nabla\chi_m$ gives a collocation form for $\mathbf{u}_m$,

$$\mathbf{u}_m = \mathbf{u}_n + \nabla\chi_n - \nabla\chi_m + \Delta t \sum_{j=0}^{M} w_{m,j}\left(-\mathbf{u}_j\cdot\nabla\mathbf{u}_j + \nu\nabla^2(\mathbf{u}_j + \nabla\chi_j) + \mathbf{f}_j\right) \text{ in } \Omega, \tag{4.14}$$

$$\mathbf{u}_m = \mathbf{u}_b(t_m) \text{ on } \partial\Omega. \tag{4.15}$$

The first two terms containing $\nabla\chi$ can be simplified as follows. Let D_m denote the discrete differentiation operator defined by taking the analytical derivative of the polynomial interpolant of a function defined at the quadrature nodes, or

$$D_m\chi = \frac{1}{\Delta t}\sum_{j=0}^{M} w'_{m,j}\chi_j \approx \frac{\partial}{\partial t}\chi(t_m). \tag{4.16}$$

The weights $w'_{m,j}$ can be found by standard interpolation theory. Since the quadrature rules defined by the weights $w_{m,j}$ are assumed to be exact for polynomials of order M,

$$\Delta t\sum_{j=0}^{M} w_{m,j}D_j\chi = \chi_m - \chi_n, \tag{4.17}$$

and hence

$$\mathbf{u}_m = \mathbf{u}_n + \Delta t\sum_{j=0}^{M} w_{m,j}\left(-\mathbf{u}_j\cdot\nabla\mathbf{u}_j - (D_j(\nabla\chi) - \nu\nabla^2\nabla\chi_j) + \nu\nabla^2\mathbf{u}_j + \mathbf{f}_j\right). \tag{4.18}$$

Hence an approximation to the pressure gradient term given in Eq. (2.28) is implicitly being computed according to

$$\nabla p_m = D_m(\nabla\chi) - \nu\nabla^2\nabla\chi_m. \tag{4.19}$$

Clearly the collocation formulation couples $\mathbf{u}_m$, $\mathbf{m}_m$, and $\nabla\chi_m$ at each collocation node with the additional coupling of the boundary conditions for $\mathbf{m}_m$ and $\nabla\chi_m$.

Similarly, one can consider a collocation formulation based on the auxiliary variable equations

$$\mathbf{u}^*_m = \mathbf{u}_n + \Delta t\sum_{j=0}^{M} w_{m,j}\left(-\mathbf{u}_j\cdot\nabla\mathbf{u}_j - \nabla q_j + \nu\nabla^2\mathbf{u}^*_j + \mathbf{f}_j\right) \text{ in } \Omega, \tag{4.20}$$

$$\mathbf{u}^*_m - \nabla\psi_m = \mathbf{u}_b(t_m) \text{ on } \partial\Omega, \tag{4.21}$$

for $m = 1,\ldots,M$, with

$$\mathbf{u}_m = \mathbb{P}(\mathbf{u}^*_m) = \mathbf{u}^*_m - \nabla\psi_m \text{ in } \Omega, \tag{4.22}$$

$$\nabla^2\psi_m = \nabla\cdot\mathbf{u}^*_m \text{ in } \Omega, \tag{4.23}$$

$$\nabla\psi_m\cdot n = 0 \text{ on } \partial\Omega, \tag{4.24}$$

for $m = 0,\ldots,M$. Rewriting these equations substituting $\mathbf{u}_m = \mathbf{u}^*_m - \nabla\psi_m$ gives the collocation form for $\mathbf{u}_m$,

$$\mathbf{u}_m = \mathbf{u}_n + \Delta t\sum_{j=0}^{M} w_{m,j}\left(-\mathbf{u}_j\cdot\nabla\mathbf{u}_j - (\nabla q_j + D_j(\nabla\psi) - \nu\nabla^2\nabla\psi_j) + \nu\nabla^2\mathbf{u}_j + \mathbf{f}_j\right) \text{ in } \Omega, \tag{4.25}$$

$$\mathbf{u}_m = \mathbf{u}_b(t_m) \text{ on } \partial\Omega. \tag{4.26}$$

As above, an approximation to the pressure update Eq. (2.31) given by

$$\nabla p_m = \nabla q_m + D_m(\nabla\psi) - \nu\nabla^2\nabla\psi_m \tag{4.27}$$

is implicitly being computed.

4.3. Spectral deferred corrections

This section provides a brief overview of the semi-implicit spectral deferred correction scheme adopted for the temporal integration of the gauge and auxiliary variable methods. Spectral deferred corrections (SDC), introduced in [23], is a variant of traditional deferred or defect correction schemes (see, e.g., [67,54,55]), which employs an integral-based correction formulation coupled with Gaussian or "spectral" integration rules. As in earlier deferred and defect correction methods, SDC is iterative in nature with each correction iteration (often referred to as a *sweep*) applying a lower-order method to an equation for the error or correction. In [23], first-order correction sweeps are used which resemble forward- and backward-Euler methods. SDC methods using a fixed number of correction sweeps can be formulated as Runge–Kutta methods with each sweep corresponding to new stage values. From this perspective, the first-order accurate sweepers used in [23] add one order of formal accuracy to the overall method for each sweep (up to the accuracy of the underlying Gaussian quadrature scheme). For Lobatto IIIA quadrature discussed above, this implies that if $2M$ SDC sweeps are performed using $M+1$ quadrature nodes, the overall method is formally order $2M$.

Alternatively, SDC methods can be considered as a fixed point iteration scheme that converges to the fully coupled Gauss collocation (or equivalently the fully implicit Gauss Runge–Kutta) solution discussed in the previous section. This point of view is necessary for variants such as multi-level SDC (MLSDC) where some SDC iterations are computed on a coarser representation of the problem to reduce the overall computational cost. Another issue with using a fixed number of SDC iterations is that of order reduction for very stiff equations, discussed further in section 4.5. An important distinction to keep in mind is that even though the underlying Gauss collocation scheme may be A-stable, the SDC iterations may not converge for arbitrary time step size and hence are not A-stable. However, when SDC schemes converge to the collocation scheme, the overall scheme is stable. This is true of the semi-implicit schemes discussed next that have a restriction on the time step due to the explicit treatment of the nonlinear terms.

The higher-order temporal methods used here are based on an IMEX or semi-implicit spectral deferred corrections (SISDC) scheme. Such methods first appeared in [47,48]. Only a brief review of these methods is provided; the reader is referred to the citations for more details.

Begin with the generic ODE

$$y' = f_E(y) + f_I(y) + f(t), \tag{4.28}$$

where the function f_E will be treated explicitly and f_I implicitly. Written in Picard integral form, Eq. (4.28) becomes

$$y(t) = y(t_0) + \int_{t_0}^{t} f_E(y(\tau)) + f_I(y(\tau)) + f(t)\, d\tau. \tag{4.29}$$

SISDC methods attain higher-order accurate solutions by solving a series of correction equations on substeps within a time step. Throughout this paper, the superscript k will refer to the SISDC iteration and the subscript m the substep.

To construct a semi-implicit SDC method, two approximate quadrature rules $w^E_{m,j}$ and $w^I_{m,j}$ are defined where it is assumed that $w^E_{m,j} = 0$ for $j > m-1$ and $w^I_{m,j} = 0$ for $j > m$ so that the following substeps are explicit in f_E and implicit in f_I. An SISDC sweep then takes the form

$$\begin{aligned} y_m^{k+1} = y_n &+ \Delta t \sum_{j=0}^{m-1} w^E_{m,j} \left(f_E(y_j^{k+1}) - f_E(y_j^k) \right) \\ &+ \Delta t \sum_{j=0}^{m} w^I_{m,j} \left(f_I(y_j^{k+1}) - f_I(y_j^k) \right) \\ &+ \Delta t \sum_{j=0}^{M} w_{m,j} \left(f_E(y_j^k) + f_I(y_j^k) + f(t_j) \right), \end{aligned} \tag{4.30}$$

for $m = 1, \ldots, M$. This system of equations can be solved by stepping through the nodes sequentially,

$$\begin{aligned} y_m^{k+1} - \Delta t\, w^I_{m,m} f_I(y_m^{k+1}) = y_n &+ \Delta t \sum_{j=0}^{m-1} w^E_{m,j} f_E(y_j^{k+1}) + w^I_{m,j} f_I(y_j^{k+1}) \\ &- \Delta t \sum_{j=0}^{m-1} w^E_{m,j} f_E(y_j^k) - \Delta t \sum_{j=0}^{m} w^I_{m,j} f_I(y_j^k) \\ &+ \Delta t \sum_{j=0}^{M} w_{m,j} \left(f_E(y_j^k) + f_I(y_j^k) + f(t_j) \right). \end{aligned} \tag{4.31}$$

All the quadrature terms on the right-hand side either depend on the solution at node $j < m$ or on the previous iteration k. Denote all terms that depend on k as

$$S_m^k = \Delta t \sum_{j=0}^{M} (w_{m,j} - w_{m,j}^E) f_E(y_j^k) + (w_{m,j} - w_{m,j}^I) f_I(y_j^k) + w_{m,j} f(t_j). \tag{4.32}$$

Likewise, let H_m^{k+1} represent the collected terms at iteration $k+1$,

$$H_m^{k+1} = \Delta t \sum_{j=0}^{m-1} w_{m,j}^E f_E(y_j^{k+1}) + w_{m,j}^I f_I(y_j^{k+1}). \tag{4.33}$$

Then one can write one substep in the SISDC sweep compactly as

$$y_m^{k+1} - \Delta t\, w_{m,m}^I f_I(y_m^{k+1}) = y_n + H_m^{k+1} + S_m^k. \tag{4.34}$$

The SISDC iteration is started by setting $y_m^0 = y_n$ for all nodes. The choice of quadrature weights $w_{m,j}^E$ correspond to the usual forward-Euler time stepping (i.e., a left-hand rectangle rule for integration), while $w_{m,j}^I$ are based on the DIRK type sweepers introduced in [66] and further discussed in the Appendix.

4.4. SISDC for the Navier–Stokes equations

SISDC methods are first considered for the solution of the Navier–Stokes equations in [47] where approaches based on both a gauge-type formulation and two auxiliary variable formulations are considered. Similar methods are used in [49,41, 2], but in none of these publications is the question of imposing boundary conditions at domain boundaries fully addressed. In this section the details of how SISDC methods are applied to the gauge and auxiliary variable formulations are described including the aspect of boundary conditions. As above, the spatial operators are left undiscretized.

To begin, consider SISDC methods applied to the gauge variable equations. Since the implicit part of the evolution equation is only the diffusive term, one substep of the SISDC sweep for the gauge equation is

$$\mathbf{m}_m^{k+1} - \Delta t\, w_{m,m}^I \nu \nabla^2 \mathbf{m}_m^{k+1} = \mathbf{m}_n + H_m^{k+1} + S_m^k, \tag{4.35}$$

which has the form of a backward-Euler step for a forced heat equation. The terms S_m^k and H_m^{k+1} are given by

$$S_m^k = \Delta t \sum_{j=0}^{M} (w_{m,j} - w_{m,j}^E) \left(-\mathbf{u}_j^k \cdot \nabla \mathbf{u}_j^k \right) + (w_{m,j} - w_{m,j}^I) \nu \nabla^2 \mathbf{m}_j^k + w_{m,j} \mathbf{f}_j, \tag{4.36}$$

$$H_m^{k+1} = \Delta t \sum_{j=0}^{m-1} w_{m,j}^E \left(-\mathbf{u}_j^{k+1} \cdot \nabla \mathbf{u}_j^{k+1} \right) + w_{m,j}^I \nu \nabla^2 \mathbf{m}_j^{k+1}, \tag{4.37}$$

with $\mathbf{u}_j^k$ defined through Eqs. (4.11)–(4.13). The boundary condition for Eq. (4.35) uses an approximation to the correct boundary condition (2.26)–(2.27),

$$\mathbf{m}_m^{k+1} - \nabla \tilde{\chi}_m^{k+1} = \mathbf{u}_b(t_m) \tag{4.38}$$

since the true value $\nabla \chi_m^{k+1}$ cannot be computed until after the projection of $\mathbf{m}_m^{k+1}$. This is a direct analog of Eq. (3.44) for a first-order method.

As mentioned above, in the original gauge method papers the analog of the approximate boundary condition for linear multistep methods is found by extrapolation in time to the order of the method. In the current context of methods with order six and higher, extrapolation produces unacceptably large errors. An iteration similar to that described in Eqs. (3.48)–(3.49) can be employed to improve this boundary condition, but at the additional expense of multiple iterations. An alternative is to use the value from the previous SISDC iteration as the boundary condition, as in

$$\mathbf{m}_m^{k+1} - \nabla \chi_m^k = \mathbf{u}_b(t_m). \tag{4.39}$$

This boundary condition is used in the numerical results in section 6.2.

A similar strategy can be applied to the auxiliary variable formulation, and several choices for the pressure update procedure are possible. The methods in [49] choose an initial pressure approximation as the variable q_m at each node which is not updated during each iteration. The resulting SISDC substeps take the form

$$\mathbf{u}_m^{*,k+1} - \Delta t\, w_{m,m}^I \nu \nabla^2 \mathbf{u}_m^{*,k+1} = \mathbf{u}_n + H_m^{k+1} + S_m^k \ \text{ in } \ \Omega, \tag{4.40}$$

$$\mathbf{u}_m^{*,k+1} - \nabla \tilde{\psi}_m^{k+1} = \mathbf{u}_b(t_m) \ \text{ on } \ \partial\Omega, \tag{4.41}$$

where now

$$S_m^k = \Delta t \sum_{j=0}^{M} (w_{m,j} - w_{m,j}^E) \left(-\mathbf{u}_j^k \cdot \nabla \mathbf{u}_j^k \right) + (w_{m,j} - w_{m,j}^I) \nu \nabla^2 \mathbf{u}_j^{*,k} + w_{m,j}(-\nabla q_j + \mathbf{f}_j), \tag{4.42}$$

$$H_m^{k+1} = \Delta t \sum_{j=0}^{m-1} w_{m,j}^E \left(-\mathbf{u}_j^{k+1} \cdot \nabla \mathbf{u}_j^{k+1} \right) + w_{m,j}^I \nu \nabla^2 \mathbf{u}_j^{*,k+1}, \tag{4.43}$$

and $\mathbf{u}_m^k = \mathbb{P}(\mathbf{u}_m^{*,k})$.

Unlike in the first-order auxiliary variable method, using the approximation $\nabla \tilde{\psi}_m^{k+1} = 0$ is not sufficiently accurate to achieve higher-order accuracy for the velocity. Similarly to what was done for the gauge method, one can use values from the previous SISDC iteration to set the boundary condition so that Eq. (4.41) becomes

$$\mathbf{u}_m^{*,k+1} - \nabla \psi_m^k = \mathbf{u}_b(t_m). \tag{4.44}$$

After the SISDC iterations are completed at iteration K for a given time step, the new pressure gradient can then be approximated by

$$\nabla p_{n+1} = \nabla q_M + D_M(\nabla \psi^K) - \nu \nabla^2 \nabla \psi_M^K. \tag{4.45}$$

It is a fortunate feature of SDC methods that an accurate value $\nabla \psi_m^K$ is available at each quadrature node. However, the order of the differentiation operator D_M is not spectral as it is for integration, so the pressure approximation will not have the same order of accuracy as that of the velocity. This is the procedure suggested in [49].

Following the example for first-order methods, one could instead try to actively correct the approximation to the pressure contained in q after individual SISDC iterations. After each iteration, the values of $\nabla \psi_m^{k+1}$ are used to improve the value of q^k by approximating Eq. (2.31),

$$\nabla q_m^{k+1} = \nabla q_m^k + D_m(\nabla \psi^{k+1}) - \nu \nabla^2 \nabla \psi_m^{k+1}. \tag{4.46}$$

To initialize the iteration, ∇q_m^0 is assigned the average in time of the pressure gradient from the previous time step for all m. This is computed by applying the spectral quadrature rule to the values of q_m^k after the final iteration in the previous time step. For the initial time step, where no previous information is known, one could simply set $\nabla q_m^0 = 0$ or to the initial pressure gradient if it is available. In the numerical tests presented here, the initial pressure is known analytically and hence is used for the initial time step.

Updating the pressure gradient approximation ∇q_m^k after each sweep has a subtle effect on the boundary condition in Eq. 4.44, namely that the magnitude of the correction terms in Eq. (4.45) decreases with each iteration so that it is again sufficient to impose

$$\mathbf{u}_m^{*,k+1} = \mathbf{u}_b(t_m) \tag{4.47}$$

in each sweep. The resulting accuracy of the pressure term is discussed in detail in section 6. Since the pressure approximation is being updated during the SISDC iterations, we call this scheme spectral deferred pressure corrections (SDPC). The SDPC scheme is summarized in Algorithm 1.

4.5. Temporal order reduction

The formal order of accuracy of a temporal method for ODEs is a property of the method in the limit as the time step Δt approaches zero. There are several instances however when the formal order of accuracy is not observed in practice and the generic term for this phenomenon is *order reduction*. In the following, order reduction in the current context of SISDC and Gauss collocation methods applied to the Navier–Stokes equations with non-trivial boundary conditions is discussed. The numerical experiments in section 6 demonstrate different types of order reduction. We enumerate the types of order-reduction to simplify the commentary in the discussion of the numerical examples.

(i) Order reduction can occur for higher-order numerical methods for stiff problems for which explicit methods are severely limited in the size of stable time step that can be taken. In the case of implicit numerical methods, this type of order reduction was first documented by Prothero and Robinson in 1974 [56] and led to the formulation of so called B-convergence by Frank, Schneid, and Ueberhuber in 1981 [30]. The essential point is that although certain single-step methods may be formally rth-order accurate and be stable for very stiff problems, the observed convergence rate of the numerical method for a range of time step size will be lower than r. Semi-implicit or IMEX methods based on Runge–Kutta or SDC have this type of order reduction as well [10,42,46]. Some suggestions for improving Runge–Kutta based methods appear in [10]. The reduction in order for SDC methods is shown in [38] to be caused by the slow convergence of SDC iterations to the collocation solution, and a procedure for accelerating this convergence is presented

Algorithm 1 Spectral Deferred Pressure Correction (SDPC) scheme.

Initialize $\mathbf{u}_0$ and $\nabla \bar{q}_0$ at time $t = 0$.
for $n = 0, 1, 2, \ldots$ **do**
  Let Δt denote the current time step, $\Delta t = t_{n+1} - t_n$.
  Initialize state variables at $k = 0$ at Lobatto collocation points $t_m \in [t_n, t_{n+1}]$:

$$\mathbf{u}_m^0 = \mathbf{u}_n, \quad \nabla q_m^0 = \nabla \bar{q}_n, \quad m = 0, \ldots, M.$$

  for sweep $k = 0, 1, \ldots, K-1$ **do**
    Compute source terms: for $m = 1, \ldots, M$,

$$S_m^k = \Delta t \sum_{j=0}^{M} (w_{m,j} - w^E_{m,j})\left(-\mathbf{u}_j^k \cdot \nabla \mathbf{u}_j^k\right) + (w_{m,j} - w^I_{m,j})\nu \nabla^2 \mathbf{u}_j^k + w_{m,j}(-\nabla q_j^k + \mathbf{f}_j).$$

    for $m = 1, \ldots, M$ **do**
      Set $H_m^{k+1} = \Delta t \sum_{j=0}^{m-1} w^E_{m,j} \left(-\mathbf{u}_j^{k+1} \cdot \nabla \mathbf{u}_j^{k+1}\right) + w^I_{m,j} \nu \nabla^2 \mathbf{u}_j^{*,k+1}$.
      Solve for $\mathbf{u}_m^{\star,k+1}$ such that

$$\mathbf{u}_m^{*,k+1} - \Delta t\, w^I_{m,m} \nu \nabla^2 \mathbf{u}_m^{*,k+1} = \mathbf{u}_n + H_m^{k+1} + S_m^k \ \text{ in } \Omega,$$
$$\mathbf{u}_m^{*,k+1} = \mathbf{u}_b(t_m) \ \text{ on } \partial\Omega.$$

      Compute $\mathbf{u}_m^{k+1} = \mathbb{P}(\mathbf{u}_m^{*,k+1})$ wherein $\nabla \psi_m^{k+1} = \mathbf{u}_m^{*,k+1} - \mathbf{u}_m^{k+1}$.
    end for
    Using $\nabla \psi^{k+1}_{m=0,\ldots,M}$ (with $\psi^{k+1}_{m=0} \equiv 0$), update q for the next sweep:

$$\nabla q_m^{k+1} = \nabla q_m^k + D_m(\nabla \psi^{k+1}) - \nu \nabla^2 \nabla \psi_m^{k+1}, \quad m = 0, \ldots, M.$$

  end for
  For the beginning of the next time step, set the velocity $\mathbf{u}_{n+1}$ equal to $\mathbf{u}_M^K$, and the time-step-averaged q equal to $\nabla \bar{q}_{n+1} = \sum_{j=0}^{M} w_{M,j} \nabla q_j^K \approx \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} \nabla p$.
end for

in [38]. Another strategy outlined in [66] for implicit SDC methods is to choose the coefficients of the approximate quadrature rule to maximize the rate of convergence of SDC iterations in the stiff limit. A similar strategy for building the coefficients w^I for Lobatto nodes is employed here which is explained in the Appendix. Since the numerical tests presented in section 6 are not extremely stiff, order reduction of this type is not observed as long as the SDC iterations converge sufficiently close to the underlying Gauss collocation schemes.

(ii) When numerical methods are applied to differential algebraic equations (DAEs), the order of accuracy can be different from that attained when solving ODEs, and the formal order might depend on the index of the DAE. In [37], the reduction in order for Gauss collocation schemes for index 1 and 2 DAEs is carefully considered. For Lobatto IIIA methods as considered here, it is conjectured in [37] that no order reduction in the differential variable should be observed for index two problems, but order reduction by a factor of two should occur for the algebraic variable. Since the incompressible Navier–Stokes equations can be considered an index 2 DAE, the results in [37] would imply that collocation methods applied to these equations would see optimal order for the velocity, but not for the pressure. This is further discussed in section 6.2.

(iii) Order reduction can be caused from applying time-dependent boundary conditions to PDEs. This was first illustrated for the method of lines and explicit Runge–Kutta methods in [16]. The cause of order reduction is due to a mismatch in the error of the solution at Runge–Kutta stages in the interior and that at the boundary. The result is that the temporal mismatch becomes a spatial numerical boundary layer. Suggestions for mitigating this type of order reduction have been proposed for explicit Runge–Kutta schemes [52,15,4,3]. Since SDC methods also employ lower-order solutions, they are also susceptible to this type of order reduction. Some recent results demonstrate that even if the SDC iterations are allowed to converge to the collocation solution, order reduction similar to that observed in the infinitely stiff limit is observed [68], and similar behavior is shown in the numerical examples in section 6.

(iv) Order reduction can be caused by a mismatch in the initial and boundary conditions imposed on the PDE, which typically can cause boundary layers in the solution. This type of order reduction is discussed for the gauge method in section 6.2.

(v) Finally, order reduction can occur when the method of lines is used for PDE solutions of insufficient regularity or smoothness. The numerical examples chosen here avoid a loss of regularity, so this type of order reduction should not occur.

5. Spatial discretization

For the numerical tests in this work we have used a high-order accurate discontinuous Galerkin (DG) spatial discretization. The methodology and algorithms employed are essentially identical to those detailed in [60,61], and as such, only a brief description is provided here.

We begin with a mesh covering the domain Ω, i.e., a collection of elements $\mathcal{E} = \bigcup_i E_i$. In all but one of our test problems, $\mathcal{E}$ is a standard uniform Cartesian grid, whereas in the circular domain test problem in section 6.4, the mesh is implicitly defined by a level set function along with a cut-cell and cell-merging procedure. Discretized quantities of state (e.g., fluid velocity $\mathbf{u}$, gauge variables $\mathbf{m}$, pressure p, etc.) are piecewise-polynomial functions defined on the mesh, i.e., when restricted to a particular element E, state functions are scalar- or vector-valued polynomials. In particular, our implementation uses tensor-product polynomials of one-dimensional degree s, e.g., $s = 3$ denotes a piecewise bicubic solution space. Following standard DG methodology, discrete solutions and associated differential operators are defined via certain kinds of weak formulations in which polynomials on adjacent elements are coupled via numerical fluxes defined on element faces.

In the above discussed SISDC schemes, discretized differential operators are required in a variety of different forms, with boundary conditions playing different roles depending on the context, as follows:

- The nonlinear advection term $\mathbf{u} \cdot \nabla \mathbf{u}$ is implemented in conservative form using an upwinding numerical flux; on mesh faces coinciding with the domain boundary, the numerical flux adopts the problem-prescribed boundary conditions for $\mathbf{u}|_{\partial\Omega}$ whenever the upwinding direction points exterior to the domain.
- Each of the discrete Laplacian operators (whether in the projection operator or semi-implicit viscous solve) is defined according to a local discontinuous Galerkin (LDG) method [5] using one-sided numerical fluxes. Briefly, the scheme can be stated as formulating a discrete gradient operator G (using a numerical flux which upwinds from the "left" on internal mesh faces) such that the accompanying discrete divergence operator (using numerical fluxes which upwind from the "right") is the adjoint of G. The primary component of the resulting discrete Laplacian operator then has the form of $G^\star G$ where $G^\star$ is the adjoint of G; boundary data enters in the form of source terms, derived from modifications to the numerical fluxes on boundary mesh faces taking into account Dirichlet and Neumann boundary conditions. In addition, following the discussion in [60], penalty stabilization parameters are included to ensure well-posedness of the discrete system. Ultimately, one obtains a symmetric positive definite linear system for the viscous solve, and a symmetric positive semidefinite linear system for the projection operator.
- Except for the influence of small penalty parameters in the LDG method, the resulting discrete projection operator is idempotent, i.e., $\mathbb{P}_h\mathbb{P}_h = \mathbb{P}_h$. In particular, $\mathbb{P}_h(\mathbf{m}) = \mathbf{m} - G\left[G^\star G + \tau A\right]^{-1}\left[G^\star \mathbf{m} - J(\mathbf{u}_\partial \cdot \mathbf{n})\right]$, where G is the discrete gradient operator, τA relates to penalty stabilization, and J is a lifting operator taking into account the prescribed normal-component boundary conditions of the projection operator in Eq. (2.8) or Eq. (2.10). It is natural, then, to use the same discrete gradient operator for the pressure source terms in Eqs. (4.42), (4.45), and (4.46) to evaluate the contribution of ∇q.
- Forcing terms, given as problem-prescribed closed-form expressions, are incorporated via L^2-projections onto the space of piecewise polynomial functions.

In regard to computational cost, the main components are that of the viscous and projection solves at each node m and SISDC sweep k. We have used a conjugate gradient method to solve these linear systems, preconditioned by an efficient geometric multigrid solver [60]. Typically, each solve requires at most 20 iterations of conjugate gradient to converge to a relative error of about 10^{-14}; the number of iterations decreases as the sweep count increases, owing to the convergence of SDPC to the fixed-point collocation solution.

In summary, the resulting DG method is high-order accurate with order of accuracy depending on the underlying polynomial degree s. In our test cases on uniform Cartesian grids of size 4×4 to 16×16, we have used $s = 9$ to obtain a 10th order method with about 13 digits of spatial accuracy on the finest mesh. For the last problem in a circular domain, $s = 4$ and a much finer mesh is used, as discussed later.

6. Numerical experiments

In this section, the performance of the gauge and SDPC methods on a variety of test cases are presented. Examples using periodic, solid-wall, and time-dependent boundary conditions are considered. The examples are presented in a simple rectangular domain in two spatial dimensions with the exception of one test that is performed in a circular domain. The examples were chosen specifically to highlight the difference in temporal convergence rates of the methods depending on the type of boundary condition applied and to highlight order reduction of different types.

6.1. Numerical tests in a rectangular domain

The following tests are all computed in the two-dimensional unit square $\Omega = (-\frac{1}{2}, \frac{1}{2})^2$ but using different boundary and initial conditions. The details of each test are presented first.

Example 1 *(Taylor–Green vortex in a periodic box).* The first test utilizes periodic boundary conditions and a smooth initial condition coinciding with a Taylor–Green vortex exact solution of the Navier–Stokes equations,

$$u(x,y,t) = 1 + \exp(-8\pi^2\nu t)\sin\big(2\pi(x-t)\big)\cos\big(2\pi(y-\tfrac{1}{8}-t)\big), \tag{6.1}$$

$$v(x,y,t) = 1 - \exp(-8\pi^2\nu t)\cos\big(2\pi(x-t)\big)\sin\big(2\pi(y-\tfrac{1}{8}-t)\big), \tag{6.2}$$

$$p(x,y,t) = \tfrac{1}{4}\exp(-16\pi^2\nu t)\big[\cos\big(4\pi(x-t)\big) + \cos\big(4\pi(y-\tfrac{1}{8}-t)\big)\big]. \tag{6.3}$$

The viscosity here is set to $\nu = 0.02$, corresponding to a moderate Reynolds number of 100, and the numerical examples are run to final time $T = 0.25$. A tenth-order spatial discretization is used. As will be shown, this test presents no difficulties for either gauge of SDPC method, and the main reason we present results from this problem in detail is so that they can be compared with those from Example 2.

Example 2 *(Taylor–Green vortex in a periodic square channel).* This example is included to demonstrate the difficulty in imposing time-dependent boundary conditions and differs from Example 1 in the way boundary conditions are handled. Periodic boundary conditions are used in the x-direction only with time-dependent velocity boundary conditions imposed on the top and bottom wall of the domain matching the exact solutions (6.1) and (6.2). Note that both the normal and tangential components of the prescribed velocity boundary conditions are non-zero and non-constant in time, as is also the pressure gradient at the top and bottom walls. The viscosity and final time are $\nu = 0.01$ and $T = 0.25$. This example will be used to demonstrate order reduction due to the time-dependent boundary conditions. A mesh of size 8×8 is used with 10th-order spatial accuracy.

Example 3 *(Manufactured solution in a periodic square channel).* This example is included to demonstrate the difference in convergence behavior between problems with time-dependent and time-independent boundary conditions. As in Example 2, periodic boundary conditions are imposed in the x-direction and velocity boundary conditions are imposed on the top and bottom of the domain. In this example, an exact solution, reminiscent of a vortex translating to the right with constant speed and with time-independent slip on the top and bottom walls, is manufactured by including a forcing term in the Navier–Stokes equations so that the exact solution is given by

$$u(x,y,t) = 1 - \exp(-8\pi^2\nu t)\sin\big(2\pi(x-t)\big)\sin\pi y\cos\pi y, \tag{6.4}$$

$$v(x,y,t) = -\exp(-8\pi^2\nu t)\cos\big(2\pi(x-t)\big)\cos^2\pi y, \tag{6.5}$$

$$p(x,y,t) = \tfrac{4}{17}\exp(-16\pi^2\nu t)\cos\big(4\pi(x-t)\big)\cos\pi y, \tag{6.6}$$

with $\nu = 0.01$. The boundary conditions are taken from the exact solution, hence the non-zero slip condition $u = 1$ is imposed at the top and bottom boundary along with $v = 0$. In the numerical tests, the final time $T = 0.125$ is used. This case is used to demonstrate that order reduction can be avoided in the case the boundary conditions are time-independent but nevertheless non-homogeneous. A mesh of size 8×8 is used with 10th-order spatial accuracy.

6.2. Convergence and order reduction for collocation formulations

The first set of tests presented are designed to demonstrate the temporal convergence rate of the collocation formulations described in 4.2. Two cases are considered, the first corresponding to Eqs. (4.4)–(4.6) ("primary variable" formulation) and the second corresponding to the gauge collocation formulation described by Eqs. (4.9)–(4.13). To compute the collocation solutions, the semi-implicit SDC methods described in 4.4 are used. For all of the computations, enough SDC iterations are executed per time step so as to ensure the residual is close to machine precision and thus the numerical results are accurately approximating the collocation solution. It should be noted that it is not particularly relevant for these examples that the collocation formulation solution is computed via SDC iterations since the convergence behavior of the collocation methods do not depend on how the solution was computed.

Consider first the collocation solution for the primary variable formulation of Example 1 (Taylor–Green vortex in periodic boundary conditions). The numerical results in Fig. 6.1 measure errors against the exact solution using the L^2 norm in space[2] and analyze several different aspects, including: (i) temporal order of accuracy as a function of the number of nodes M in the Gauss–Lobatto scheme; (ii) spatial order of accuracy as the grid is refined; and (iii) three different possible methods for evaluating the pressure. Three different grid resolutions are used, 4×4, 8×8, and 16×16, colored black, blue, and green, respectively. The horizontal lines in Fig. 6.1 indicate the points at which the error is saturated by spatial discretization effects; note that the spatial error decreases by a factor of about 1000 each time the grid is refined by a factor of two,

[2] Throughout this work, the L^2 norm in space is used; similar conclusions regarding the temporal order of accuracy for the presented methods hold when instead measuring errors with the L^∞ norm in space. However, for sufficiently small time steps Δt, the maximum norm is more sensitive to the limits of spatial discretization accuracy imposed by, e.g., conditioning and double-precision arithmetic.

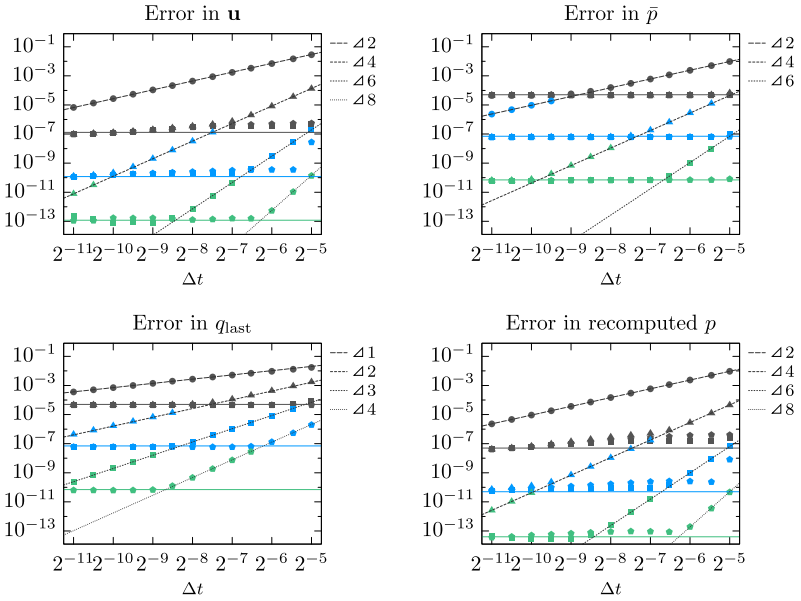

Fig. 6.1. Convergence results for the collocation solution of Example 1, varying the time step size, spatial grid resolution, and node count M of the Gauss–Lobatto scheme. In each panel, data points indicate the measured errors at final time $T = 0.25$, whereas the lines of indicated slope illustrate the observed order of accuracy and the horizontal lines indicate the point at which spatial discretization errors dominate. Results computed on 4×4, 8×8, and 16×16 Cartesian grids are colored black, blue, and green, respectively; note the spatial discretization error decreases by a factor of about 1000 each time the grid is refined, consistent with 10th order accuracy in space for the underlying DG scheme using tensor-product degree-nine polynomials. Gauss schemes of different order are denoted by ●, ▲, ■, and ⬟ for $M = 1, 2, 3$, and 4, respectively. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

consistent with 10th order spatial accuracy. Note that the temporal convergence rate for the velocity field is $2M$, which is optimal for the Gauss–Lobatto collocation scheme.

Concerning the evaluation of pressure, included here are three different possible approaches:

(i) Use the value of p at the last quadrature node of the last time step; in Fig. 6.1 this is referred to as q_{last}.
(ii) Compute the mean of p over the last time step, using the collocation method's associated quadrature rule to perform a weighted sum over all quadrature nodes q_j, $j = 0, \ldots, M$; this is referred to as $\bar{p}$.
(iii) Recompute p directly from the velocity field (at the last quadrature node of the last time step) by projecting the Navier–Stokes equations, i.e., by taking the divergence of the momentum equations, $\nabla^2 p = -\nabla \cdot (\mathbf{u} \cdot \nabla \mathbf{u}) + \nabla \cdot \mathbf{f}$, applying the associated boundary condition $\nabla p \cdot n = (-\mathbf{u}_t - \mathbf{u} \cdot \nabla \mathbf{u} + \nu \nabla^2 \mathbf{u}) \cdot n$, and inverting the resulting Poisson problem. Note, however, in the example currently under consideration having periodic boundary conditions, there are no boundary conditions on p.

As demonstrated in Fig. 6.1, these approaches can have different orders of accuracy as well as errors of differing magnitude. In general, the third approach, which recomputes pressure directly from the computed velocity field, retains the same temporal order as the velocity itself; however, this assumes a method of sufficient accuracy is available to compute $\mathbf{u}_t \cdot n$, and, furthermore, requires significant spatial accuracy because of the high-order spatial derivatives needed to evaluate source terms and boundary conditions. The first approach, while the most simplistic, is observed not to have optimal order of accuracy: the value of p at the last quadrature node is order M accurate, not $2M$ as would be optimal. This relates to the discussion in section 4.2 concerning the non-uniqueness of the pressure values in collocation schemes. The remaining approach, option (ii) above, is optimal order accurate, but only determines the mean of pressure over the last time step. Although option (iii) yields the best errors in this particular example, in the following examples (which all require boundary conditions) the same approach exhibits significantly degraded accuracy owing to the method's excessive requirements on spatial accuracy. As such, in the remainder of this paper, only the method which computes the mean of p is shown.

Fig. 6.2 illustrates results for the collocation solution in the primary variable formulation of Example 2 (Taylor–Green vortex in a periodic channel with time-dependent boundary conditions). These results are computed on a 8×8 grid in the DG method having 10th order accuracy. Note that both the velocity and pressure mean demonstrate order reduction, with asymptotic order M instead of the optimal $2M$. This level of order reduction is similar but not identical to the theory for index 2 DAEs where only the algebraic variable has order reduction [37]. Since the following example with time independent boundary condition exhibits no order reduction in the velocity, it is logical to attribute the observed order reduction to the time-dependent boundary conditions in this example.

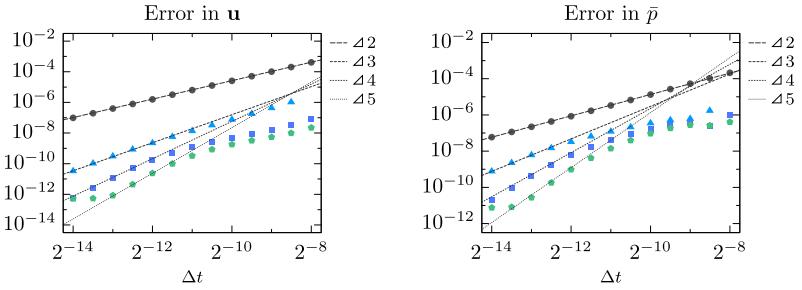

Fig. 6.2. Convergence results for the collocation solution of Example 2, exhibiting order reduction owing to its associated time-dependent boundary conditions. In each panel, data points indicate the measured errors at final time $T = 0.25$, whereas the lines of indicated slope illustrate the observed order of accuracy. Results computed on a 8×8 Cartesian grid with a 10th order accurate DG method; spatial accuracy affects only the highest-order scheme with $M = 4$ and time steps smaller than 2^{-13}. Gauss schemes of different order are denoted by ●, ▲, ■, and ⬟ for $M = 1, 2, 3$, and 4, respectively.

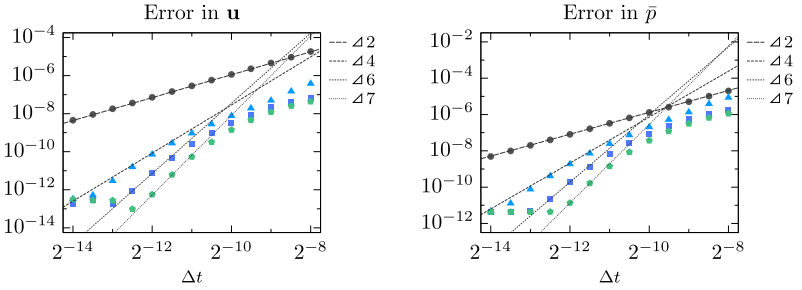

Fig. 6.3. Convergence results for the collocation solution of Example 3, varying the time step size and node count M of the Gauss–Lobatto scheme. In each panel, data points indicate the measured errors at final time $T = 0.125$, whereas the lines of indicated slope illustrate the observed order of accuracy. Results computed on a 8×8 Cartesian grid with a 10th order accurate DG method; measured errors plateau when the spatial discretization errors dominate. Gauss schemes of different order are denoted by ●, ▲, ■, and ⬟ for $M = 1, 2, 3$, and 4, respectively.

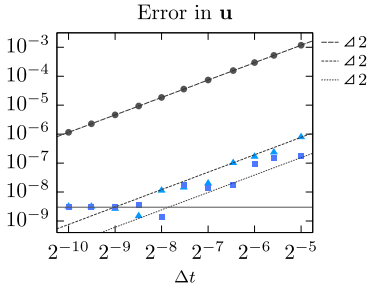

Fig. 6.4. Convergence results for the collocation solution of Example 2 using gauge variables, varying the time step size and node count M of the Gauss–Lobatto scheme. The data points indicate the measured error in the velocity field (i.e., projection of the primary solution variable $\mathbf{m}$) at final time $T = 0.25$, together with illustrative lines of slope two and a horizontal line indicating the spatial discretization error saturation point. Gauss schemes of different order are denoted by ●, ▲, and ■ for $M = 1, 2$, and 3, respectively. Note the severe order reduction, attributed to the incompatible initial and boundary conditions inherent in this example when using gauge variables; see also Fig. 6.5.

Results for Example 3 (time-independent boundary conditions), again for the collocation solution in the primary variable formulation, are shown in Fig. 6.3 and show that optimal order accuracy is recovered in this case. Ideally, a slope of eight should be seen with five Gauss–Lobatto quadrature nodes ($M = 4$), however spatial accuracy (itself limited by double-precision arithmetic) inhibits seeing the asymptotic convergence rate in this particular problem.

In the last example of this section, we again consider the collocation solution of Example 3, but instead in the gauge variable formulation, Eqs. (4.9)–(4.13). Fig. 6.4 shows that under this formulation, the collocation solution for the gauge method demonstrates significant order reduction: the observed order of accuracy is two, independent of M. The primary reason for this is due to the fact that, in general, gauge methods exhibit boundary conditions which are *incompatible* with initial values, resulting in boundary layers in the solution. To wit, the simplest gauge method solves Eq. 2.22 with initial condition $\mathbf{m}(x, t = 0) = \mathbf{u}(x, t = 0)$ and $\chi(x, t = 0) = 0$ in Ω and boundary conditions $\mathbf{m} \cdot n = \mathbf{u}_b \cdot n$ and $\nabla\chi \cdot n = 0$ on $\partial\Omega$. Differentiating the Neumann boundary condition for χ in time shows that $\nabla\chi_t \cdot n = 0$ on $\partial\Omega$. In addition, the gradient of the relation for pressure, Eq. (2.28), gives $\nabla p = \nabla\chi_t - \nu\nabla^2\nabla\chi$. Evaluating the normal component of this last expression at $t = 0$ implies $\nabla p \cdot n = 0$ at $t = 0$ on $\partial\Omega$, which, in general, is not true. In other words, χ is the solution to a heat equation $\chi_t - \nu\nabla^2\chi = p$, whose right-hand-side may be such that the initial condition $\chi \equiv 0$ and boundary condition $\nabla\chi \cdot n = 0$ are

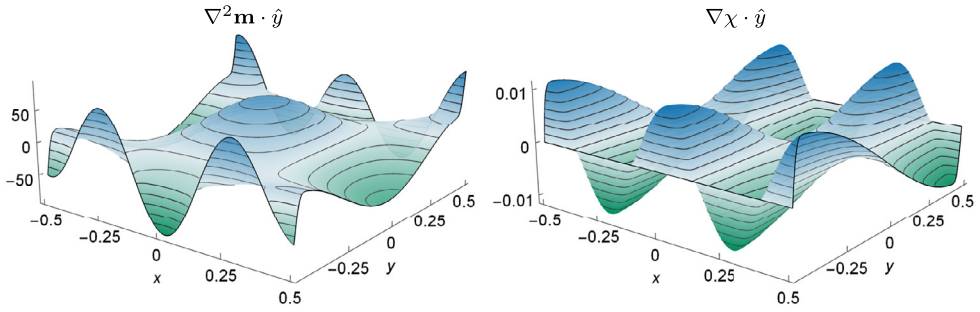

Fig. 6.5. Graphs of the second components of $\nabla^2\mathbf{m}$ and $\nabla\chi$ at time $t = 0.015625$, where $\mathbf{m}$ and χ are the gauge solutions of the incompressible Navier–Stokes equations with initial and boundary conditions specified by Example 3. Note the pronounced boundary layers near $y = \pm 0.5$, with typical thickness about 0.02 in the transverse direction $\hat{y}$. These boundary layers are not numerical but rather physical, similar in nature to solving a PDE with incompatible initial and boundary conditions. (Closed-form expressions for the functions shown here are not known; in these graphs, the functions were computed with the DG method with enough spatial resolution to ensure a faithful representation of the exact solution. In particular, the boundary layer is assuredly resolved by four biquartic elements in the transverse direction.)

incompatible with a smooth solution in the strong sense. As such, χ may develop boundary layers, and because $\mathbf{u} = \mathbf{m} - \nabla\chi$ with $\mathbf{u}$ presumed smooth, it follows that $\mathbf{m}$ must *also* develop boundary layers. Consequently, high-order temporal accuracy cannot be obtained with traditional approaches. Fig. 6.5 demonstrates this fact by illustrating the collocation solution of Example 3 in the gauge variable formulation shortly after $t = 0$. Note the pronounced boundary layers in $\nabla\chi$ and second derivatives of $\mathbf{m}$; to emphasize, these boundary layers are present in the exact solution and are not numerical artifacts.

On this note, one may consider altering the boundary or initial conditions for χ in an attempt to prevent incompatibilities of this kind. For example, one possibility is to set χ at $t = 0$ as the solution to the Poisson problem $-\nu\nabla^2\chi = p|_{t=0}$ in Ω with $\nabla\chi\cdot n = 0$ on $\partial\Omega$. On the surface, this approach results in compatible initial and boundary conditions, however the issue remains in higher-temporal derivatives of incompatibility, e.g., according to this procedure, $\nabla p_t\cdot n = 0$ on $\partial\Omega$ at $t = 0$, which, again, is not true in general. A second possibility may be to include a source term q in the Navier–Stokes equations as in the auxiliary variable methods of [49]; in this case, the heat equation for χ becomes $\chi_t - \nu\nabla^2\chi = p - q$, and so, if q is chosen equal to p at $t = 0$ and held constant in time, then, again, incompatibilities arise because this approach necessitates $\nabla p_t\cdot n = 0$ on $\partial\Omega$ at $t = 0$. In other words, such auxiliary variable methods are also subject to order reduction owing to incompatible initial and boundary conditions, although these incompatibilities are buried inside higher-temporal derivatives, and so higher-than-second order accuracy may be attained. We are unaware of any procedure to initialize a gauge method such that initial and boundary data is compatible for every order of temporal derivative, beyond having knowledge of p, p_t, p_{tt}, ... at $t = 0$, which is unlikely. In essence, although gauge variables provide a reformulation of the incompressible Navier–Stokes equations, a smooth solution (in the strong sense) in primary variables may not result in a smooth solution (in the strong sense) in gauge variables. The preceding discussion highlights the need to iteratively update the pressure at the end of each SDC sweep, as in SDPC, in order to achieve high-order temporal accuracy. The remainder of the results here are computed in the primary variable formulation.

Finally, we note that in the all of the above examples, the observed time step threshold corresponds to a CFL number of about 1/20, which is about standard for a DG method using degree nine polynomials [53,44,18]. (In the presented convergence plots of temporal accuracy, e.g., Figs. 6.1, 6.2, 6.3, etc., the largest time steps displayed are near this threshold.) As mentioned in section 4.3, this constraint on the time step comes not from the underlying collocation schemes (which are A-stable), but rather from the failure of the SDC iterations to converge to this solution for larger CFL numbers due to the explicit treatment of the advective terms.

6.3. Convergence of SDC Iterations for SDPC

One question that arises from the observed order reduction for collocation methods in the previous section is how many iterations of the SDPC method are required to reach the maximum (possibly reduced) order of accuracy in these tests. This question is addressed by computing the error after each SDPC iteration for a fixed number of nodes using the same initial conditions as in the previous section. For these tests however, only one time step is performed for each case so that the computed error corresponds to the truncation error of the method. Four Lobatto nodes ($M = 3$) are used corresponding to an optimal order of 7 for the truncation error.

Fig. 6.6 displays the results for the error in velocity. For Example 1 and 3, where optimal convergence rates are observed for the collocation methods, the SDPC methods appear to add one order of accuracy per iteration as expected by theory. On the other hand, in Example 2, where order reduction is observed for the collocation methods due to time dependent boundary conditions, the SDPC iterations appear to add only one half order of accuracy per iteration. To the authors' knowledge, this type of half-order increase per iteration in SDC methods has not been demonstrated in the past, and the cause of this behavior is currently not understood.

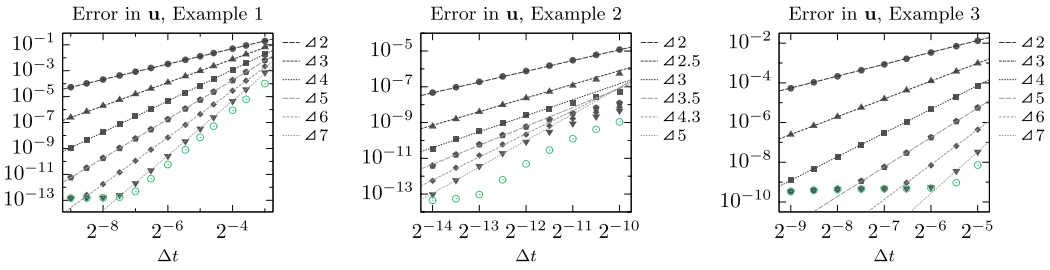

Fig. 6.6. Convergence results examining the truncation error as a function of number of sweeps for an SDPC scheme with node count $M = 3$. In each panel, data points indicate the measured error in the velocity field after a single time step, whereas the lines of indicated slope illustrate the observed rate of reduction in truncation error. The number of iterations (sweeps) of the SDPC method is denoted by ●, ▲, ■, ⬟, ◆, and ▼ for 1, 2, 3, …, 6 sweeps, respectively. Data points shown with a ○ symbol denote errors obtained after 9 sweeps and are representative of the errors of the collocation solution.

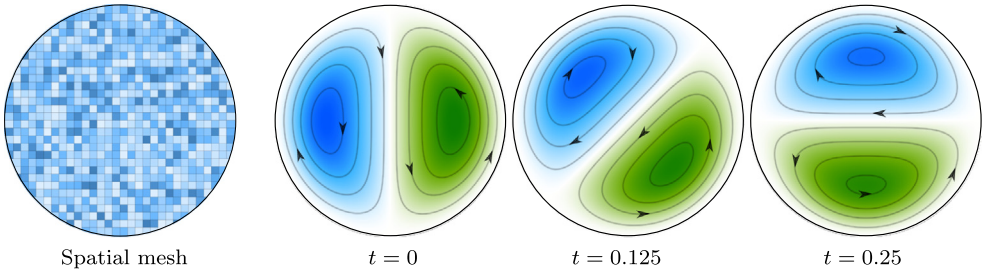

Fig. 6.7. *(left)* Coarsened representation of the spatial mesh used for the circular domain test problem in §6.4. *(right three)* Streamlines of the flow at three instances in time, showing two main vortices at either end of a dipole-like structure rotating clockwise in time.

6.4. An example with curved boundaries

The final test case is included to demonstrate the temporal accuracy of the methods for problems with curved boundaries. The domain is now a circle of radius one, $\Omega = \{x^2 + y^2 < 1\}$, with no-flow and no-slip boundary conditions prescribed on the boundary. The underlying spatial discretization remains a regular Cartesian grid except that cells near the domain boundary are modified following a cut-cell and cell-merging method detailed in [60]. Unlike the previous tests, which used tensor-product degree-nine polynomials, in this test the DG spatial discretization uses tensor-product degree-four polynomials but on a finer mesh: this is owing to subtleties in our implicit mesh DG implementation wherein very high-degree polynomial spaces lead to increased ill-conditioning, so instead medium-degree polynomials are employed but on a finer mesh. A coarsened representation of the spatial mesh is shown in the left side of Fig. 6.7; the mesh used in the following results consists of 51,360 elements each of size of about 1/128.

A non-trivial exact solution similar to two vortices at either end of a dipole rotating about the center of the circle is manufactured by including a forcing term in the equations. The exact velocity field is given by the curl of the stream function

$$\psi = \exp(-20\nu t)\cos\left(\tfrac{\pi}{2}(x^2+y^2)\right)^2(x\cos 2\pi t - y\sin 2\pi t), \tag{6.7}$$

with pressure

$$p = -\tfrac{1}{4}\exp(-40\nu t)\sin\left(2\pi(x^2+y^2)\right)(x\cos 2\pi t - y\sin 2\pi t)(x^2+y^2). \tag{6.8}$$

The viscosity is $\nu = 0.02$ giving a Reynolds number approximately 160. The final time for each run is $T = 0.25$, after which the dipole has rotated 90°. Fig. 6.7 shows the contours of the stream function at $t = 0$, $t = T/2$, and the final time $t = T$.

For this test we again compare the convergence rate of the velocity and pressure for different values of M and SDPC iterations K. Specifically, we consider $M + 1 = 2, 3, 4$, and 5 Lobatto nodes with two different choices for the number of SDPC iterations K per time step, namely $K = 2M$ and $K = 3M$. In Fig. 6.8, the error in velocity and pressure is displayed for $2M$ iterations per time step. For this case, the optimal convergence rate of $2M$ is not clearly observed.

In Fig. 6.9, the number of SDPC iterations is increased to $3M$, and now the optimal order of convergence is recovered. With $3M$ sweeps, the SDPC scheme is sufficiently converged to the collocation schemes, hence Fig. 6.9 can be compared with the collocation results in section 6.2. For this example, we observe a slight increase in the number of SDPC iterations required to reach the optimal order of accuracy of the associated collocation scheme. Numerical tests using a more accurate Stokes solver in each SDPC substep regain optimal order of accuracy, hence there is a trade-off between the effort required in each substep and the number of SDPC sweeps required. This issue will be addressed more carefully in future work.

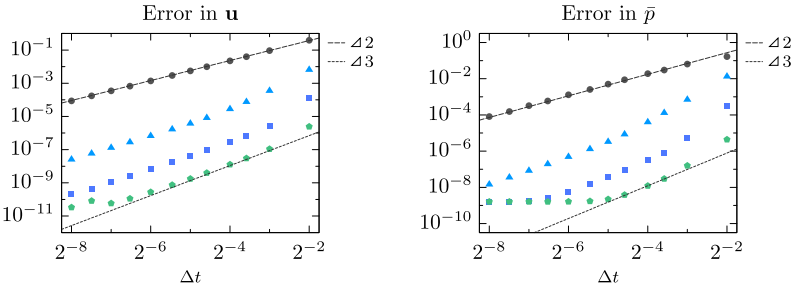

Fig. 6.8. Convergence results for the SDPC method, using $2M$ sweeps per time step, applied to the circular domain test problem in §6.4, varying the time step size and node count M of the underlying Gauss–Lobatto scheme. In each panel, data points indicate the measured errors at final time $T = 0.25$, whereas the lines of indicated slope illustrate the typical rate of error reduction. Gauss schemes of different order are denoted by ●, ▲, ■, and ⬟ for $M = 1, 2, 3$, and 4, respectively.

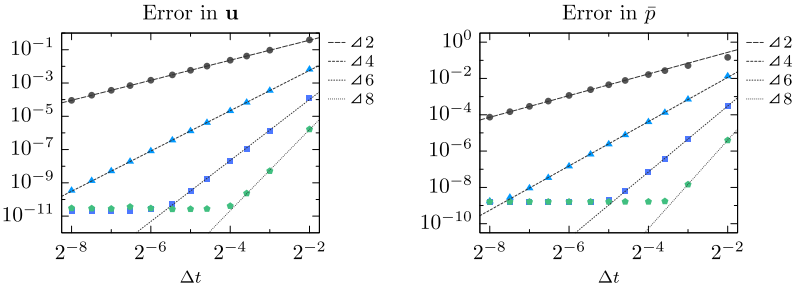

Fig. 6.9. Convergence results for the SDPC method, using $3M$ sweeps per time step, applied to the circular domain test problem in §6.4, varying the time step size and node count M of the underlying Gauss–Lobatto scheme. In each panel, data points indicate the measured errors at final time $T = 0.25$, whereas the lines of indicated slope illustrate the observed order of accuracy. Gauss schemes of different order are denoted by ●, ▲, ■, and ⬟ for $M = 1, 2, 3$, and 4, respectively.

7. Conclusions and discussion of future directions

In this paper, we discussed higher-order, semi-implicit temporal integration strategies for the incompressible Navier–Stokes equations. Three different, but analytically-equivalent, formulations of historical significance formed the basis of this discussion, namely projection, gauge, and auxiliary variable formulations. In section 3, it is shown that all three formulations can be configured to be mathematically equivalent for a variety of first-order time-stepping strategies (provided spatial derivative operators are in the continuum and left undiscretized); these strategies included one-step semi-implicit schemes as well as iterative schemes converging to first-order Stokes-like problems in which velocity is coupled to pressure. Hence, in this setting, there is no clear benefit in choosing one particular formulation over another. However, in section 6 we showed that certain formulations can be problematic when higher-order accuracy is sought. In particular, we showed that the gauge formulation exhibits severe order reduction in the sense that smooth solutions (in the strong sense) cannot generally be expected for the gauge variables $\mathbf{m}$ and χ, i.e., the gauge solution exhibits physical boundary layers reminiscent to solving PDEs with incompatible initial and boundary conditions, affecting both spatial and temporal accuracy.

Motivated by the need to iteratively update velocity and pressure approximations and associated boundary conditions, in this paper we developed a higher-order time-stepping scheme for the incompressible Navier–Stokes equations. The temporal method is based on applying an iterative semi-implicit deferred correction strategy to the momentum equation and pressure approximation to iteratively improve the solution and ultimately drive it to the Gauss collocation solution. The iterative correction to the pressure approximation allows the velocity boundary conditions to be applied directly in the momentum equation approximation without causing boundary layers in the solution and subsequently a reduction of order (as is demonstrated with a similar gauge method). Since the pressure correction appears to be crucial to attaining very high order of accuracy, we refer to the scheme as the spectral deferred pressure correction method (SPDC). The computational complexity of each substep in each SPDC iteration is essentially that of a first-order projection method requiring the solution to the implicit diffusion equation and the Poisson equation associated with the projection.

In essence, the SDPC scheme combines the iterative nature of an SISDC method together with projection methods and an appropriate pressure update formula. As such, each iteration or sweep of the SISDC scheme does not need to precisely solve a Stokes problem at each substep. Instead, the projection method, which sequentially performs a implicit viscous solve and pressure update, functions as an approximate Stokes solver. Nevertheless, we found that the performance of the iterative approach benefits from having a decent approximate Stokes solver. For example, the circular domain test problem in section 6.4 benefits from increased sweep count, which is conjectured to be related to the more complex mesh topology and tighter dependence between components of velocity field as compared to the simpler Cartesian grid cases. Our results are also tied to a certain kind of high-order DG spatial discretization. For example, in improving the performance

of the approximate Stokes solver, we found that it was sometimes beneficial to apply a filtering operation to the pressure correction step to dampen the high-frequency modes created by the associated and sensitive high-order spatial derivatives in the pressure update. The analysis of SDPC with this and other spatial discretizations, e.g., finite difference methods, finite volume, approximate (rather than idempotent) projection operators is also an important avenue worthy of investigation.

The numerical results included here show that the SDPC method converges to the collocation solution and can obtain optimal temporal accuracy for problems with time-independent boundary conditions in the sense that the convergence order corresponds to spectrally accurate Gaussian quadrature rules (here Gauss–Lobatto rules). However, when time-dependent Dirichlet boundary conditions are enforced, order reduction is observed which is reminiscent in character to when collocation schemes are applied to problems in the infinitely stiff limit. Specifically, for $M+1$ Lobatto nodes, the maximum order observed is M instead of $2M$. In these cases, each SPDC iteration appears to increase the order of accuracy by about one-half rather than one as in the case with time-independent boundary conditions for reasons that are not currently understood. A formal analysis of this behavior based on normal mode analysis is a future research goal.

The number of SDPC substeps required to reach the spectral order of accuracy of the underlying Gauss quadrature rule (namely $2M$ for $M+1$ Lobatto nodes) is substantial. There are however some ways of mitigating this cost. First, when an iterative method is used for the implicit viscous solves as is done with multigrid here, the number of multigrid iterations can be reduced as compared to a full accuracy solve [64]. Also, a multilevel or MLSDC method can be used where some SDPC iterations are performed on coarsened versions of the problem (in time and/or space) [63]. Lastly, SDC type methods can be combined with the PFASST parallel in time strategy [28] to amortize the cost of SDPC iterations over multiple concurrent time steps. Results combining the SDPC method developed here with MLSDC and PFASST will be reported in the future. In addition, there are several avenues for generalizing the SPDC strategy including problems with more general inflow or outflow boundary conditions, flows with nonconstant density (as in [41]), or problems with dynamic interfaces [59], which will be the focus of future research.

Acknowledgements

This research was supported by the Applied Mathematics Program of the U.S. DOE Office of Advanced Scientific Computing Research under contract number DE-AC02-05CH11231. Some computations used the resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

Appendix A

The lower order quadrature rule defined by the weights $w^I_{m,j}$ in section 4.3 is constructed following the idea presented by Weiser in [66]. The discussion in [66] is restricted only to Radau type quadrature nodes, hence the modification for applying this idea to Lobatto nodes is presented here.

For the discussion, it is sufficient to consider the linear model problem $y' = \lambda y$ with $y(0) = 1$. For this equation, one can write an SDC iteration in a matrix-vector form. Denote by $\mathbf{Q}$ the $(M+1)\times(M+1)$ matrix of quadrature weights $w_{m,j}$ from Eq. (4.3) stored in the last M rows. The first row of $\mathbf{Q}$ consists of zeros. Let $\mathbf{Y}$ denote the $M+1$ vector of values y_m satisfying the collocation solution (4.3) at the Lobatto nodes. Then $\mathbf{Y}$ satisfies

$$\mathbf{Y} = \mathbf{1} + \lambda\Delta t\,\mathbf{Q}\mathbf{Y}, \tag{A.1}$$

where $\mathbf{1}$ is the length $M+1$ vector with all values equal 1. Since the first row of $\mathbf{Q}$ is zero, the first row of this equation enforces the initial condition.

Consider now an SDC method using implicit substepping. Denote by $\mathbf{Y}^k$ the vector of solution values y^k_m at the Lobatto nodes for SDC iteration k. Similarly, let $\mathbf{Q}^I$ represent the approximate quadrature weights $w^I_{m,j}$ so that $\mathbf{Q}^I$ is lower triangular with nonzero diagonal entries (except for $m = j = 1$). One SDC sweep takes the form

$$\mathbf{Y}^{k+1} - \lambda\Delta t\,\mathbf{Q}^I\mathbf{Y}^{k+1} = \mathbf{1} + \lambda\Delta t\,(\mathbf{Q} - \mathbf{Q}^I)\mathbf{Y}^k. \tag{A.2}$$

Assuming that the first component of $\mathbf{Y}^0$ is 1 to match the initial condition, it is clear that it remains 1 since the first row $\mathbf{Q}^I$ is zero.

Note that the collocation solution is a fixed point of the SDC iteration (A.2). Therefore

$$(\mathbf{Y} - \mathbf{Y}^{k+1})\lambda\Delta t\,\mathbf{Q}^I\mathbf{Y}^{k+1} = \lambda\Delta t\,(\mathbf{Q} - \mathbf{Q}^I)(\mathbf{Y} - \mathbf{Y}^k). \tag{A.3}$$

This gives an explicit expression of the convergence of the SDC iterations to the collocation solution. Note that the first entry of $\mathbf{Y} - \mathbf{Y}^k$ is always zero, hence the first column of $\mathbf{Q}^I$ does not affect the convergence behavior of the SDC sweeps and can be ignored. Therefore, let $\tilde{\mathbf{Q}}$ and $\tilde{\mathbf{Q}}^I$ denote the $M\times M$ matrices corresponding to $\mathbf{Q}$ and $\mathbf{Q}^I$ without the first row and column. Likewise, if $\tilde{\mathbf{Y}}$ and $\tilde{\mathbf{Y}}^k$ are M vectors omitting the first entry of $\mathbf{Y}$ and $\mathbf{Y}^k$, then

$$\tilde{\mathbf{Y}} - \tilde{\mathbf{Y}}^k = \mathbf{P}(\lambda\Delta t)(\tilde{\mathbf{Y}} - \tilde{\mathbf{Y}}^k), \tag{A.4}$$

where

$$\mathbf{P}(\lambda\Delta t) = \lambda\Delta t(\mathbf{I} - \lambda\Delta t\,\tilde{\mathbf{Q}}^I)^{-1}(\tilde{\mathbf{Q}} - \tilde{\mathbf{Q}}^I). \tag{A.5}$$

The eigenvalues of $\mathbf{P}$ will hence determine the rate of convergence of SDC iterates to the collocation solution. Considering Eq. (A.5) in the stiff limit $\lambda\Delta t \to -\infty$ gives

$$\lim_{\lambda\Delta t\to-\infty} \mathbf{P}(\lambda\Delta t) = (\tilde{\mathbf{Q}}^I)^{-1}(\mathbf{Q} - \tilde{\mathbf{Q}}^I) = (\tilde{\mathbf{Q}}^I)^{-1}\mathbf{Q} - \mathbf{I}. \tag{A.6}$$

The key observation in [66] is to let $\mathbf{LU} = \tilde{\mathbf{Q}}^T$ be the classical LU decomposition with $\mathbf{L}$ lower-triangular with ones on the diagonal. This implies that $\mathbf{U}^T = \tilde{\mathbf{Q}}(\mathbf{L}^T)^{-1}$ or $(\mathbf{U}^T)^{-1} = \mathbf{L}^T\tilde{\mathbf{Q}}^{-1}$. Hence, if we choose $\tilde{\mathbf{Q}}^I = \mathbf{U}^T$ in A.6

$$(\tilde{\mathbf{Q}}^I)^{-1}\tilde{\mathbf{Q}} - \mathbf{I} = \mathbf{L}^T - \mathbf{I}. \tag{A.7}$$

Hence the iteration matrix $\mathbf{P}$ has only zero eigenvalues in the stiff limit. In all the numerical tests in this paper, the coefficients $w^I_{m,j}$ are defined by the corresponding entries in $\mathbf{U}^T$.

References

[1] G. Akrivis, Implicit–explicit multistep methods for nonlinear parabolic equations, Math. Comput. 82 (2013) 45–68.
[2] A.S. Almgren, A.J. Aspden, J.B. Bell, M.L. Minion, On the use of higher-order projection methods for incompressible turbulent flow, SIAM J. Sci. Comput. 35 (2013) B25–B42.
[3] I. Alonso-Mallo, Runge–Kutta methods without order reduction for linear initial boundary value problems, Numer. Math. 91 (2002) 577–603.
[4] I. Alonso-Mallo, C. Palencia, Optimal orders of convergence for Runge–Kutta methods and linear, initial boundary value problems, Appl. Numer. Math. 44 (2003) 1–19.
[5] D.N. Arnold, F. Brezzi, B. Cockburn, L.D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM J. Numer. Anal. 39 (2002) 1749–1779.
[6] U.M. Ascher, S.J. Ruuth, R.J. Spiteri, Implicit-explicit Runge–Kutta methods for time-dependent partial differential equations, Appl. Numer. Math. 25 (1997) 151–167.
[7] U.M. Ascher, S.J. Ruuth, B.T.R. Wetton, Implicit–explicit methods for time-dependent partial differential equations, SIAM J. Numer. Anal. 32 (1995) 797–823.
[8] J.B. Bell, P. Colella, H.M. Glaz, A second-order incompressible projection method for the Navier–Stokes equations, J. Comput. Phys. 283 (1989) 257–283.
[9] J.B. Bell, D.L. Marcus, A second-order projection method for variable-density flows, J. Comput. Phys. 101 (1992) 334–348.
[10] S. Boscarino, Error analysis of IMEX Runge–Kutta methods derived from differential-algebraic systems, SIAM J. Numer. Anal. 45 (2007) 1600–1621.
[11] S. Boscarino, G. Russo, On the uniform accuracy of IMEX Runge–Kutta schemes and applications to hyperbolic systems with relaxation, Comm. SIMAI Congress 2 (2007).
[12] S. Boscarino, G. Russo, On a class of uniformly accurate IMEX Runge–Kutta schemes and applications to hyperbolic systems with relaxation, SIAM J. Sci. Comput. 31 (2009) 1926–1945.
[13] D.L. Brown, R. Cortez, M.L. Minion, Accurate projection methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 168 (2001) 464–499.
[14] M. Cai, A. Nonaka, J.B. Bell, B.E. Griffith, A. Donev, Efficient variable-coefficient finite-volume Stokes solvers, Commun. Comput. Phys. 16 (2014) 1263–1297.
[15] M. Calvo, C. Palencia, Avoiding the order reduction of Runge–Kutta methods for linear initial boundary value problems, Math. Comput. 71 (2002) 1529–1543.
[16] M.H. Carpenter, D. Gottlieb, S. Abarbanel, W.-S. Don, The theoretical accuracy of Runge–Kutta time discretizations for the initial boundary value problem: a study of the boundary error, SIAM J. Sci. Comput. 16 (1995) 1241–1252.
[17] A.J. Chorin, Numerical solution of the Navier–Stokes equations, Math. Comput. 22 (1968) 745–762.
[18] B. Cockburn, C.-W. Shu, Runge–Kutta discontinuous Galerkin methods for convection-dominated problems, J. Sci. Comput. 16 (2001) 173–261.
[19] P. Colella, K. Pao, A projection method for low speed flows, J. Comput. Phys. 149 (1999) 245–269.
[20] E.M. Constantinescu, A. Sandu, Extrapolated implicit–explicit time stepping, SIAM J. Sci. Comput. 31 (2010) 4452.
[21] G.-H. Cottet, P.D. Koumoutsakos, Vortex methods: theory and practice, Meas. Sci. Technol. 12 (2001) 354.
[22] M.S. Day, J.B. Bell, Numerical simulation of laminar reacting flows with complex chemistry, Combust. Theory Model. 4 (2000) 535–556.
[23] A. Dutt, L. Greengard, V. Rokhlin, Spectral deferred correction methods for ordinary differential equations, BIT Numer. Math. 40 (2000) 241–266.
[24] W. E, J.-G. Liu, Finite difference schemes for incompressible flows in the velocity–impulse density formulation, J. Comput. Phys. 130 (1997) 67–76.
[25] W. E, J.-G. Liu, Gauge finite element method for incompressible flows, Int. J. Numer. Methods Fluids 34 (2000) 701–710.
[26] W. E, J.-G. Liu, Projection method III: spatial discretization on the staggered grid, Math. Comput. 71 (2001) 27–48.
[27] W. E, J.-G. Liu, Gauge method for viscous incompressible flows, Commun. Math. Sci. 1 (2003) 317–332.
[28] M. Emmett, M.L. Minion, Efficient implementation of a multi-level parallel in time algorithm, in: 21st International Conference on Domain Decomposition Methods, Rennes, 2012, pp. 1–8.
[29] J. Frank, W. Hundsdorfer, J.G. Verwer, On the stability of implicit–explicit linear multistep methods, Appl. Numer. Math. 25 (1997) 193–205.
[30] R. Frank, J. Schneid, C. Ueberhuber, Order results for implicit Runge–Kutta methods applied to stiff systems, SIAM J. Numer. Anal. 22 (1985) 515–534.
[31] A. Garba, P. Haldenwang, A Helmholtz–Hodge projection method using an iterative gauge computation to solve the 3D generalized Stokes problem, SIAM J. Sci. Comput. 35 (2013) A1560–A1583.
[32] K. Goda, A multistep technique with implicit difference schemes for calculating two- or three-dimensional cavity flows, J. Comput. Phys. 30 (1979) 76–95.
[33] P.M. Gresho, R.L. Sani, On pressure boundary conditions for the incompressible Navier–Stokes equations, Int. J. Numer. Methods Fluids 7 (1987) 1111–1145.
[34] B.E. Griffith, An accurate and efficient method for the incompressible Navier–Stokes equations using the projection method as a preconditioner, J. Comput. Phys. 228 (2009) 7565–7595.
[35] J.L. Guermond, P. Minev, J. Shen, An overview of projection methods for incompressible flows, Comput. Methods Appl. Mech. Eng. 195 (2006) 6011–6045.

[36] E. Hairer, S.P. Norsett, G. Wanner, Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, 2nd revised edition, Springer, Berlin, 1991.
[37] E. Hairer, M. Roche, C. Lubich, The Numerical Solution of Differential-Algebraic Systems by Runge–Kutta Methods, Lect. Notes Math., 1989.
[38] J. Huang, J. Jia, M. Minion, Accelerating the convergence of spectral deferred correction methods, J. Comput. Phys. 214 (2006) 633–656.
[39] L.O. Jay, Lobatto Methods, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015, pp. 817–826.
[40] S. Jiang, M.C.A. Kropinski, B.D. Quaife, Second kind integral equation formulation for the modified biharmonic equation and its applications, J. Comput. Phys. 249 (2013) 113–126.
[41] S.Y. Kadioglu, R. Klein, M.L. Minion, A fourth-order auxiliary variable projection method for zero-Mach number gas dynamics, J. Comput. Phys. 227 (2008) 2012–2043.
[42] C.A. Kennedy, M.H. Carpenter, Additive Runge–Kutta schemes for convection–diffusion–reaction equations, Appl. Numer. Math. 44 (2003) 139–181.
[43] J. Kim, P. Moin, Application of a fractional-step method to incompressible Navier–Stokes equations, J. Comput. Phys. 59 (1985) 308–323.
[44] L. Krivodonova, R. Qin, An analysis of the spectrum of the discontinuous Galerkin method, Appl. Numer. Math. 64 (2013) 1–18.
[45] M. Lai, J.B. Bell, P. Colella, A projection method for combustion in the zero Mach number limit, in: 11th AIAA Computational Fluid Dynamics Conference, 1993.
[46] A.T. Layton, M.L. Minion, Implications of the choice of quadrature nodes for Picard integral deferred corrections methods for ordinary differential equations, BIT Numer. Math. 45 (2005) 341–373.
[47] M.L. Minion, Higher-order semi-implicit projection methods, in: Numerical Simulations of Incompressible Flows, Half Moon Bay, CA, 2001, 2003, pp. 126–140.
[48] M.L. Minion, Semi-implicit spectral deferred correction methods for ordinary differential equations, Commun. Math. Sci. 1 (2003) 471–500.
[49] M.L. Minion, Semi-implicit projection methods for incompressible flow based on spectral deferred corrections, Appl. Numer. Math. 48 (2004) 369–387.
[50] A. Nonaka, A.S. Almgren, J.B. Bell, M.J. Lijewski, C.M. Malone, M. Zingale, Maestro: an adaptive low Mach number hydrodynamics algorithm for stellar flows, Astrophys. J. Suppl. Ser. 188 (2010) 358–383.
[51] L. Pareschi, G. Russo, Implicit–explicit Runge–Kutta schemes and applications to hyperbolic systems with relaxation, J. Sci. Comput. 25 (2005) 129–155.
[52] D. Pathria, The correct formulation of intermediate boundary conditions for Runge–Kutta time integration of initial boundary value problems, SIAM J. Sci. Comput. 18 (1997) 1255–1266.
[53] W. Pazner, P.-O. Persson, Interior penalty tensor-product preconditioners for high-order discontinuous Galerkin discretizations, in: 2018 AIAA Aerospace Sciences Meeting, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2018.
[54] V. Pereyra, On improving an approximate solution of a functional equation by deferred corrections, Numer. Math. 8 (1966) 376–391.
[55] V. Pereyra, Iterated deferred corrections for nonlinear operator equations, Numer. Math. 10 (1967) 316–323.
[56] A. Prothero, A. Robinson, On the stability and accuracy of one-step methods for solving stiff systems of ordinary differential equations, Math. Comput. 28 (1974) 145–162.
[57] Y. Ren, Y. Jiang, M. Liu, H. Zhang, Class of fully third-order accurate projection methods for solving the incompressible Navier–Stokes equations, Acta Mech. Sin./Lixue Xuebao 21 (2005) 542–549.
[58] G. Russo, P. Smereka, Impulse formulation of the Euler equations: general properties and numerical methods, J. Fluid Mech. 391 (1999) 189–209.
[59] R. Saye, Interfacial gauge methods for incompressible fluid dynamics, Sci. Adv. 2 (2016).
[60] R. Saye, Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid-structure interaction, and free surface flow: Part I, J. Comput. Phys. 344 (2017) 647–682.
[61] R. Saye, Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid-structure interaction, and free surface flow: Part II, J. Comput. Phys. 344 (2017) 683–723.
[62] T. Schneider, N. Botta, K.J. Geratz, R. Klein, Extension of finite volume compressible flow solvers to multi-dimensional, variable density zero Mach number flows, in: Variable Density Zero Mach Number Flows, vol. 155(2), 1999, pp. 248–286, J. Comput. Phys. 158 (2000) 262.
[63] R. Speck, D. Ruprecht, M. Emmett, M. Minion, M. Bolten, R. Krause, A multi-level spectral deferred correction method, BIT Numer. Math. (2014).
[64] R. Speck, D. Ruprecht, M. Minion, M. Emmett, Inexact spectral deferred corrections, in: Thomas Dickopf, Martin J. Gander, Laurence Halpern, Rolf Krause, Luca F. Pavarino (Eds.), Domain Decomposition Methods in Science and Engineering XXII, Springer International Publishing, 2016, pp. 389–396.
[65] S. Vater, R. Klein, Stability of a cartesian grid projection method for zero Froude number shallow water flows, Numer. Math. 113 (2009) 123–161.
[66] M. Weiser, Faster SDC convergence on non-equidistant grids by DIRK sweeps, BIT Numer. Math. 55 (2015) 1219–1241.
[67] P. Zadunaisky, Motion of Halley's comet during the return of 1910, Astron. J. 71 (1966).
[68] B. Zhang, D. Chen, J. Huang, On the Gauss Runge–Kutta and method of lines transpose for initial-boundary value parabolic PDEs, Commun. Comput. Phys. (2018).
[69] H. Zhang, A. Sandu, S. Blaise, Partitioned and implicit–explicit general linear methods for ordinary differential equations, J. Sci. Comput. 61 (2014) 119–144.