Journal of Computational Physics 403 (2020) 109057

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A fractional-step method for steady-state flow

J. Blair Perot [a,*], Martin Sanchez-Rocha [b], Paul Malan [b]

[a] Theoretical and Computational Fluid Dynamics Laboratory, University of Massachusetts, Amherst, MA, 01003, USA
[b] Dassault Systemes Simulia Corp, 1301 Atwood Avenue, Suite 101W, Johnston, RI 02919, USA

ARTICLE INFO

Article history:
Received 27 February 2019
Received in revised form 15 October 2019
Accepted 20 October 2019
Available online 31 October 2019

Keywords:
Fractional-step method
Projection method
Steady-state algorithms
Navier-stokes
Saddle-point
Incompressible flow

ABSTRACT

A fractional step method for solving the steady-state, incompressible, Navier-Stokes equations is presented. The proposed iterative method uses an estimate for the optimal under-relaxation coefficient at each iteration. This estimate uses prior information that is already available in the iterative method so the calculation has negligible cost. Numerical tests show that the convergence rate of the fractional step method using this optimal relaxation parameter is nearly independent of the mesh size, resulting in significant cost savings for problems with fine meshes. The ideas demonstrated in this paper are general and can be used to improve many existing iterative methods for steady flows, as well as methods for the unsteady Navier-Stokes equations and methods for other saddle-point problems.

1. Introduction

Fractional-step methods [1] are an approach to solving complex partial differential equation (PDE) problems in which various coupled physical processes are treated uncoupled and sequentially. The original formulation was used to split advection and diffusion processes, or just the diffusion process in different directions [2]. However, fractional-step methods have now found their most lasting appeal in the context of saddle-point PDE problems and are used to split dynamics equations from kinematic constraints. The saddle-point problem of primary interest in this work is the incompressible Navier-Stokes equations for fluid flow. In these equations the dynamics are given by the conservation of momentum and the kinematic constraint is that the velocity field should be divergence-free (conservation of mass). In the context of constraint satisfaction, the fractional-step method is essentially identical to the projection method which was developed separately [3].

Fractional-step methods for incompressible flow were originally developed for unsteady applications, such as the direct numerical simulation (DNS) of turbulence [4–7]. The steady-state analog of the fractional-step method, in which the time derivative is absent but it is replaced by an under-relaxation term, is a solution approach typically referred to as "segregated" methods. Well known examples of segregated algorithms are SIMPLE [8] and its family of derivatives such as SIMPLEC [9] and SIMPLER [10]. These algorithms have even been extended to the unsteady case by treating the unsteady term as yet another source term. Similar to dual-time-stepping methods, the unsteady versions are useful for transient problems where the time scales are larger than time-step size required by the time-accurate fractional-step approach. The PISO [11] method is a modification of segregated methods that is more efficient for smaller time scales by reducing or eliminating the need for sub-iteration, similar to the classic fractional step method.

In this work, the mathematical analysis used in the fractional-step method [12][13] is used to develop a new algorithm with superior convergence. It is important to mention that the mathematical analysis and insight developed in this work

* Corresponding author.
E-mail address: perot@umass.edu (J.B. Perot).

https://doi.org/10.1016/j.jcp.2019.109057
0021-9991/

can be applied to other segregated algorithms. In addition, the proposed steady-state fractional-step method (termed the M-method) has the desirable characteristic of automatically reverting to the classical unsteady fractional-step method in the limit of small (time accurate) time steps. The method presented herein is regarded as a steady-state version of the fractional-step method even though the classical fractional-step method cannot be applied to the steady limit. In this work, the fractional-step (developed for unsteady and time-accurate) and segregated approaches (developed primarily for steady or large timesteps) are considered to be different historically, and we show how theoretically they are very similar and can be seamlessly unified.

Classical segregated methods (such as SIMPLE) have traditionally been the workhorse of most commercial and open-source computational fluid dynamics (CFD) software. They have been applied to industrial simulations all over the world, and likely consume an extraordinarily large number of CPU cycles today. However, they suffer from two major drawbacks. The primary deficiency is that the classical segregated methods exhibit mesh-dependent convergence rates. This means that, for the same problem (flow conditions do not change), high spatial accuracy and fine-mesh resolution simulations converge far more slowly than their coarse mesh counterparts. This behavior is shown in Fig. 1a for the SIMPLE method applied to steady 2D cavity flow at a Reynolds number of 1000 (more details on this test case are found in section 6).

Fig. 1 shows the error norm (not the residual) of the velocity (the primary variable in equation (1) below) as a function of the iteration number for the same physical problem using different-sized uniform triangular meshes. It is important to highlight that the residual has different convergence properties than the error. Observations of the residual can be deceptive about the actual method performance, because the error is the true measure of solution convergence. As a result, errors (and not residuals) will always be used in this paper. The mesh sizes in the legend refer to the number of triangles in the domain. The figure shows that the convergence rate of SIMPLE is roughly inversely proportional to the number of unknowns. For example, the 24k mesh (green triangles) takes 3x more iterations than the 8k mesh (blue squares) to obtain the same error level. So with classic segregated methods like SIMPLE to achieve the same error level, doubling the number of mesh cells requires doubling the number of iterations (as well as doubling the work per iteration). So to achieve a fixed error level, the CPU time is $O(N^2)$ where N is the number of unknowns. All the existing segregated algorithms show this type of convergence and cost behavior for high Reynolds number steady flows. One possible remedy is to use a coupled solution algorithm, but this adds a significant overhead in terms of memory and the computational cost per iteration. Additionally, any bottleneck in convergence affects all the equations being solved.

Mesh dependency means that existing segregated solution methods, which once were reasonably efficient when computations only used moderate mesh sizes, have become less practical now that mesh sizes have increased to take full advantage of modern hardware and to solve problems of increasing geometric complexity and with higher accuracy. Iteration counts in the many thousands are now routine in commercial CFD applications where element counts number in the millions. In contrast, the M-method described in this work converges at rates that are nearly mesh independent, as shown in Fig. 1b. Like coupled methods, CPU costs for the proposed method are nearly the optimal order, $O(N)$. But the cost per iteration is expected to be far less than a coupled method.

Another major drawback of the classical segregated solution methods is that under-relaxation factors need to be specified by the user. For example, SIMPLE has two user-specified under-relaxation parameters, and PISO has four. In Fig. 1a, the velocity under-relaxation is set to 0.8 and the pressure under-relation is set to 0.2. These are typical default values found in many commercial and open-source codes. However, those default values are not guaranteed to give convergent answers and often have to be changed for problems with difficult physics (for example flow in porous media), resulting in slower convergence. The steady-state fractional-step method (M-method shown in Fig. 1b) has no parameters to specify, although it can be tuned if needed. The M-method behaves similarly to the classical segregated methods, but with optimal under-relaxation parameters calculated automatically at each iteration.

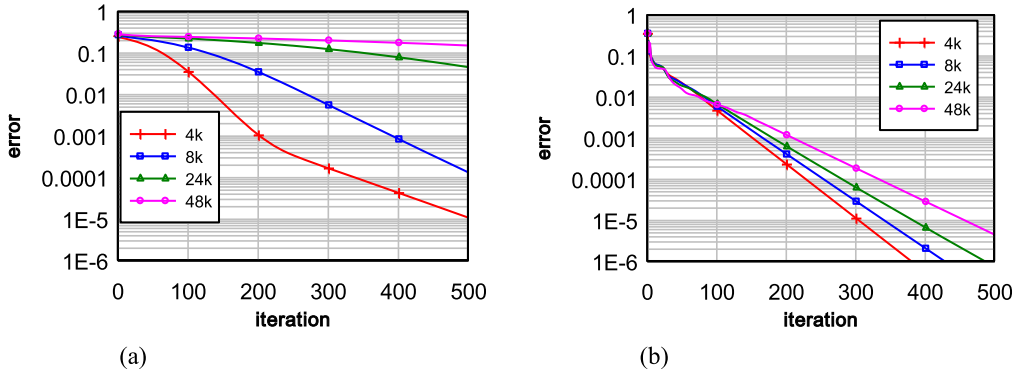

Fig. 1. Steady cavity flow at Reynolds number of 1000. Error versus iteration number for a variety of nearly uniform triangular meshes. Mesh size is given by the number of triangles. (a) SIMPLE algorithm showing strong mesh dependence. (b) M-method showing only mild mesh dependence. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Note that there are two convergence regimes for the velocity error. SIMPLE typically converges slowly at first and then speeds up as the iterations progress. The 8k mesh shows this behavior the most clearly in Fig. 1a. On the other hand, the M-method (Fig. 1b) shows rapid convergence of the error at early iterations, and a slowing of the convergence and the appearance of mesh dependence towards the end (after about iteration 100). This work presents a mathematical analysis that explains these behaviors. The analysis shows that the early iterations are controlled by velocity (or primary unknown) error and the later iterations by pressure (or constraint) error. The various segregated solver methods affect these two coupled errors in different ways that can be mathematically analyzed. Also note that it is only important to reduce the convergence error to a level that it is commensurate with the other modeling errors in the problem, and not to machine zero. Practical applications will often terminate at a relative error of 10^{-3} or 10^{-4}.

Section 2 of this paper provides the mathematical background of the saddle-point problem and section 3 the direct relationship to the incompressible Navier-Stokes equations. In section 4, a general linear algebra framework for understanding the performance of all segregated solution methods for the saddle-point problem is presented, and section 5 relates the most widely known segregated solution methods to this single unified framework. Section 6 uses the mathematical framework to derive the M-method, and section 7 shows the performance of the M-method. In section 8, the M-method is shown to be insensitive to the two possible tunable parameters.

2. Background

The specific problem of interest for all segregated and fractional-step methods (steady and unsteady) can be represented as a non-linear block-matrix problem,

$$\begin{bmatrix} F(\boldsymbol{u}) & G \\ D & 0 \end{bmatrix} \begin{pmatrix} \boldsymbol{u} \\ p \end{pmatrix} = \begin{pmatrix} \boldsymbol{b} \\ c \end{pmatrix} \tag{1}$$

The matrix F is invertible. The matrices G and D are rectangular and not invertible, and the matrix 0 is a large block of zeros. The vector-valued array, $\boldsymbol{u}$, is the list of primary unknowns, and the scalar array, p, is the list of Lagrange multipliers that enforce the constraints. Linear algebraic systems of equations of this type can arise from any physical problem with constraints. Although the present focus is on the application of these ideas to incompressible fluid flow, the M-method is just as applicable to the solution of any block matrix problem of the form given by equation (1).

Saddle point equation systems such equation (1) are difficult to solve because they are indefinite (both positive and negative real-parts to the eigenvalues). In addition, the system is not diagonally dominant because of the zero block on the diagonal, so methods like Gauss-Seidel and Jacobi iteration and their fast descendants like multigrid methods, cannot be directly implemented. Because the system is assumed to be large and sparse, it also cannot be efficiently solved with a direct Gauss-elimination method.

This system can be solved as a fully coupled system (such as in [14]) using some specialized iterative methods such as BiCGstab or GMRES. But the fully-coupled approach also has some practical drawbacks. The fully coupled system converges at the rate of its weakest link, which happens to be the scalar pressure, p. This forces the more complex velocity vector, $\boldsymbol{u}$, part of the problem (with the expensive nonlinear F matrix), to be solved far more often than it needs to be. Uncoupling (or segregating) the system allows existing and well understood solution methods for Poisson's equation and the advection-diffusion equations to be applied separately to each part of the uncoupled problem. With a segregated algorithm, computational power is thus directed most efficiently to each part of the problem.

In practice, even when the fully coupled system is solved, a preconditioner is used. The best preconditioners have been shown to be the segregated or fractional-step methods [15,16], such as those that are discussed in this paper. So the innovations described below are directly applicable whether the system is solved fully coupled or segregated. An interesting exception to the applicability of this paper is the exact fractional-step method [17]. That approach uses special properties of the operators G and D (if those properties exist, i.e., the discretization is mimetic [18–20]) to transform equation (1) into a basis where the constraints are automatically satisfied.

3. Application to fluid dynamics

For those with an interest in the application to fluid dynamics, the direct connection with equation (1) is presented in this section. In the context of incompressible fluid dynamics, the vector valued unknowns $\boldsymbol{u}$ are the velocity and the scalar Lagrange multipliers, p, are the pressure unknowns. The first row of equation (1) is the momentum equation,

$$\frac{M^{n+1}\boldsymbol{u}^{n+1} - M^n\boldsymbol{u}^n}{\Delta t} + \sum_f (\boldsymbol{u}\cdot\boldsymbol{n}^f \rho \boldsymbol{u}^{n+1} - \mu \frac{\partial \boldsymbol{u}^{n+1}}{\partial n}) A^f = -Gp^{n+1} + \boldsymbol{f} \tag{2a}$$

where G is the discrete gradient operation, μ is the non-constant dynamic viscosity, ρ is the density which is not necessarily constant (such as in an incompressible oil/water situation), M is the mass in a control volume formulation or a mass matrix for a finite element (FE) method, $\boldsymbol{f}$ represents all additional forces such as gravity or surface tension, A^f are the control volume face areas and $\boldsymbol{n}^f$ are the control volume outward face normal vectors. Although a finite element method might

represent the advection and diffusion terms slightly differently, those differences are irrelevant to the discussion presented in this work. A different time advancement scheme (than the implicit Euler method shown in equation (2a)) would also not affect the results of this paper.

Equation (2a) can be rearranged to collect the unknowns and isolate the operators,

$$[\frac{M^{n+1}}{\Delta t}+\sum_f A^f(\boldsymbol{u}^f\cdot\boldsymbol{n}\rho^f-\mu\frac{\partial}{\partial n})]\boldsymbol{u}^{n+1}+Gp^{n+1}=\frac{M^n}{\Delta t}\boldsymbol{u}^n+\boldsymbol{f} \tag{2b}$$

So the matrix F in this example is the unsteady advection-diffusion operator,

$$F(\boldsymbol{u})=\left[\frac{M^{n+1}}{\Delta t}+\sum_f A^f(\boldsymbol{u}\cdot\boldsymbol{n}\rho^f-\mu\frac{\partial}{\partial n})\right] \tag{3a}$$

In the case of steady-state flow, which we focus on in this work, the matrix F is the sum of all the face flux terms (hence the name F). The steady-state case is the unsteady system in the limit of $\Delta t\to\infty$. Note that the matrix F is not constant in the fluid flow example, but is a function of the solution due to the advection term. This nonlinearity is also a reason why the fully coupled solution of equation (1) may be less advantageous than a segregated approach. Since nonlinear iteration is also required, high accuracy solution of equation (1) is unnecessary, and uncoupled solution methods have the advantage of being able to control the velocity and pressure error levels independently. The known vector on the right-hand side in the first row of equation (1) is,

$$\boldsymbol{b}=\frac{M^n}{\Delta t}\boldsymbol{u}^n+\boldsymbol{f} \tag{3b}$$

Boundary conditions terms also appear in this vector.

The second row of equation (1) represents the constraint equations in block form,

$$D\boldsymbol{u}^{n+1}=\sum_f A^f\boldsymbol{n}^f\cdot\boldsymbol{u}^{n+1}=0 \tag{4}$$

where D is the discrete divergence operator. The right-hand side of equation (1), the scalar list c, is mostly 0 with the exception of Dirichlet boundary condition entries for the normal velocity on boundaries. In a well-constructed discretization, $G=-D^T$ which mimics the underlying symmetry of the continuous gradient and divergence operators. However, this property is not a requirement of the analysis in this paper, because it is not true in most commercial or open-source CFD codes (where the proposed technology might be adopted).

The literature on segregated methods would tend to write equation (2a) or (2b) as,

$$a_p\boldsymbol{u}_p^{n+1}+\sum_{nb}a_{nb}\boldsymbol{u}_{nb}^{n+1}+Gp^{n+1}=\boldsymbol{b} \tag{5}$$

where a_p is the diagonal value of the matrix F, and a_{nb} are the off-diagonal entries of the matrix F on the same row as a_p. Often, the derivation is also for a Cartesian mesh where the gradient operator simplifies further, but we leave it general to illustrate the connection with equation (2a) or (2b). The various segregated methods differ in their approximation of the off-diagonal term in (5).

Segregated solver algorithms rarely frame the algorithm in terms of linear algebra as is done in equation (1). Similarly, early fractional-step papers focused on time advancement as the issue, and did not use a matrix representation of the problem. However, references [12] and [17] show how posing the problem in terms of a matrix representation can provide significant insight into the fractional-step method and its properties. In this paper, the matrix formulation of the problem enables the derivation of the optimal under-relaxation parameter.

4. General framework and insights

The key insight of reference [12] was that the fractional-step method can be formulated as an approximate block LU factorizations of equation (1). This section will show that all segregated solution algorithms can also be described and analyzed in this same way. Using this general framework, this paper identifies the three key approximations that parameterize all such methods, including the new steady-state fractional step method (M-method) which is discussed in this work.

The approximate block LU decomposition that captures many well-known segregated solution methods is,

$$\begin{bmatrix} Q_1+F & 0 \\ D & -DQ_2^{-1}G \end{bmatrix}\begin{bmatrix} I & Q_2^{-1}GQ_3^{-1} \\ 0 & Q_3^{-1} \end{bmatrix}\begin{pmatrix}\delta\boldsymbol{u}\\ \delta p\end{pmatrix}=\begin{pmatrix}\boldsymbol{r}_u\\ r_p\end{pmatrix}=\begin{pmatrix}\boldsymbol{b}-F\boldsymbol{u}-Gp\\ c-D\boldsymbol{u}\end{pmatrix} \tag{6}$$

This is posed in incremental form, where the goal is to find the corrections $\delta\boldsymbol{u}$ and δp to an existing solution guess $\boldsymbol{u}$ and p. If the incremental form is iterated, and it converges, it will always converge to the solution of equation (1). This is

because the right-hand side of the incremental form is the residual of equation (1). In incremental form, the matrix on the left-hand side does not need to be exact for convergence to the solution of equation (1) to occur. Segregated solvers take advantage of this fact, and use lower and upper triangular block approximations of the matrix that are easier to invert.

Equation (6) represents the subset of segregated solution methods that exactly enforce the discrete divergence constraint at the end of every iteration. Methods that allow error in the divergence constraint can be thought of as pseudo-compressible approximations, and those approximations have been explored extensively by Quarteroni et al. [21] and by Elman et al. [15]. In our experience, methods that do not exactly conserve mass can cause odd and unphysical numerical effects, so segregated methods based on artificial compressibility are not included in the decomposition given by equation (6).

Equation (6) is solved using an iterative scheme, which accomplishes two very different goals. It is iterated to address the non-linearity in F, and it is also iterated to eventually satisfy both the top row of equation (1) (the dynamics) and the bottom row of equation (1) (the constraints), without solving them both at the same time.

The three Q matrices define the different segregated solver methods. In many segregated solver derivations, the Q matrices are equal to each other, which can lead to a loss of clarity about what is occurring, as the three approximations act in very different ways. Note that Q_2 should be easy to invert for practical reasons because it occurs inside a pseudo-Laplacian, $L = DQ_2^{-1}G \approx \nabla \cdot Q_2^{-1}\nabla$. Viguerie and Veneziani [22] explore the possibilities when Q_2 is the viscous term in the Navier-Stokes equations. But in this work, and most segregated methods, Q_2 is restricted to be a diagonal (or easy to invert) matrix. In contrast, Q_3 does not need to be easy to invert because in actual implementation we will show that it does not appear as an inverse.

Segregated and fractional step methods are usually described in terms of four steps. It is useful to see that those four steps actually come from equation (6) which is a block LU problem that contains two sub-problems,

$$\begin{bmatrix} Q_1 + F & 0 \\ D & -DQ_2^{-1}G \end{bmatrix} \begin{pmatrix} \delta \boldsymbol{u}^* \\ \delta p^* \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}_u \\ r_p \end{pmatrix} \quad \text{and} \quad \begin{bmatrix} I & Q_2^{-1}GQ_3^{-1} \\ 0 & Q_3^{-1} \end{bmatrix} \begin{pmatrix} \delta \boldsymbol{u} \\ \delta p \end{pmatrix} = \begin{pmatrix} \delta \boldsymbol{u}^* \\ \delta p^* \end{pmatrix} \tag{7}$$

The first problem expands into the first two steps, and the second problem expands into the last two steps,

$$\begin{aligned} &(Q_1 + F)\delta \boldsymbol{u}^* = \boldsymbol{r}_u = \boldsymbol{b} - F\boldsymbol{u} - Gp \\ &(DQ_2^{-1}G)\delta p^* = D(\delta \boldsymbol{u}^*) - r_p = D(\delta \boldsymbol{u}^*) - c + D\boldsymbol{u} = D(\boldsymbol{u} + \delta \boldsymbol{u}^*) - c \\ &\delta \boldsymbol{u} = \delta \boldsymbol{u}^* - Q_2^{-1}G\delta p^* \\ &\delta p = Q_3 \delta p^* \end{aligned} \tag{8}$$

The first step is the solution of a modified advection diffusion problem, F, with a lagged constraint force term (the pressure gradient term in this case). This is segregated from the second step which is a pseudo-Poisson problem for the pressure correction. Well-developed and fast methods, such as multigrid, exist for the segregated pseudo-Laplacian problem in step 2. The final two steps are often explicit and therefore fast. They are the corrections to the two uncoupled solves.

Numerical or mathematical processes are often better understood when there is a physical analogy to accompany the mathematics. For this reason, we provide such an analogy here for segregated methods given by equation (8), in the form of calculating the motion of a simple pendulum using Cartesian (x,y) coordinates for the pendulum position. This is an appropriate analogy because it involves dynamics (like the first row of equation (1)) and a constraint that the pendulum length is fixed (like the second row of equation (1)). Fig. 2a shows the exact pendulum position at time n and time n+1.

The segregated solver methods first move the pendulum without enforcing the constraint. They include the tension force at time n in that calculation (the pressure gradient force in our case), but the tension force at time n is not correct for enforcing the pendulum length constraint at time n+1. This step is shown in Fig. 2b. In the second step the segregated solver evaluates how far from the constraint it is, and calculates the force that is required to move the pendulum onto the

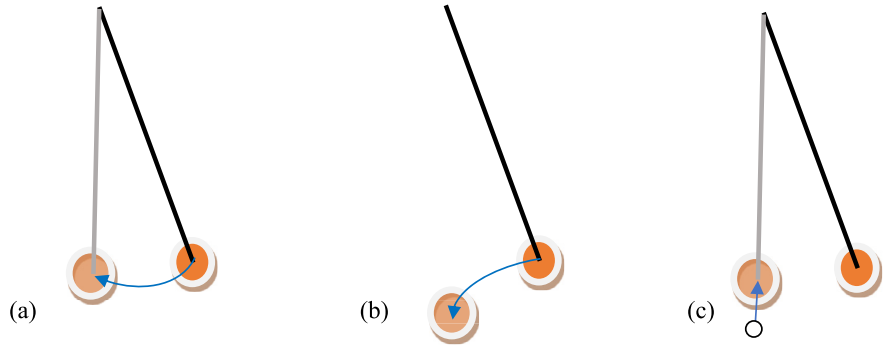

Fig. 2. (a) Exact motion of a pendulum. (b) Step 1 of a segregated solver. Motion without constraint. (c) Step 3 of a segregated solver. Correction onto the constraint using the information that was calculated in step 2. This figure visualizes the variable **u** (as position) but not the rod tension, p.

constraint (this is the elliptic equation for δp^* in our case). The third step uses this newly calculated tension force to move the pendulum and, as planned, that motion will now put the pendulum back onto the constraint. This is shown in Fig. 2c. However, note that the tension force required to move a misbehaving pendulum onto the constraint, is *not* the same as the tension force found in a properly behaving pendulum, even though they are related. The fourth step attempts to get the actual pendulum tension δp from the tension force required to push the pendulum back onto the constraint, δp^*.

The exact fractional step method [17] may also be explained using this analogy. Exact fractional-step methods change the coordinate system of the saddle-point problem so that the constraints are trivial to satisfy. In the case of the pendulum this means changing the equations of motion into radial coordinates r, θ. Then any numerical approximations to the solution for θ has no impact on the trivially enforced constraint that r is constant. That change of coordinates in the saddle-point problem (and incompressible Navier-Stokes case) is always possible using the Helmholtz decomposition, but it is simple numerically only if the discrete gradient and divergence operators behave fundamentally like their continuous counterparts (that is, they are mimetic, [20]).

By multiplying the block LU components in equation (6), it is seen that the segregated methods given by the four steps of equation (8) only approximate equation (1) on each iteration. The segregated methods actually invert a related matrix,

$$\begin{bmatrix} Q_1 + F & (Q_1 + F)Q_2^{-1}GQ_3^{-1} \\ D & 0 \end{bmatrix} \begin{pmatrix} \delta \boldsymbol{u} \\ \delta p \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}_u \\ r_p \end{pmatrix} \tag{9}$$

The divergence equation (constraint equation) on the second row is satisfied perfectly, but the momentum equation (top row) is approximated during each iteration, The matrix Q_1 is added to remove the near-singularity in the matrix F that sometimes occurs at steady state as well as to achieve non-linear convergence by adding necessary under-relaxation.

5. Segregated methods

The major segregated methods are tabulated in Table 1 in terms of their block LU splitting (equation (6)). This makes them easier to compare to each other. In all these methods, Q_1 and Q_2 are diagonal matrices although this is not a requirement of segregated methods. Additional methods, which are variations on these primary approximation themes, are summarized in the review paper of reference [23].

Table 1
Summary of segregated-solver approximations in the general splitting framework given by equation (6) (or equation (8)).

	Q_1	Q_2	Q_3
Classical Fractional-Step	0	$\frac{M^{n+1}}{\Delta t}$	I
SIMPLE	$\frac{1-\omega_u}{\omega_u} F_D$	$Q_1 + F_D$	$\omega_p I \ \omega_p \approx (1-\omega_u)$
SIMPLEC	$\frac{1-\omega_u}{\omega_u} F_D$	$Q_1 + \frac{M^{n+1}}{\Delta t}$	$\omega_p I \ \omega_p \approx 1$
SIMPLER	$\frac{1-\omega_u}{\omega_u} F_D$	$Q_1 + F_D$	$(DQ_4^{-1}G)^{-1}[DQ_4^{-1}(Q_1+F)Q_2^{-1}G]$
M-Method	αF_D	$Q_1 + \frac{M^{n+1}}{\Delta t}$	$Q_3 = \omega_p(I - \beta \tilde{D}\frac{\mu}{Q_2}\tilde{G})$

5.1. Classical fractional-step method

This method is the simplest of the segregated methods. Classical fractional-step methods are restricted to unsteady situations because the limit $\Delta t \to \infty$ makes the inverse of $Q_2 = \frac{M^{n+1}}{\Delta t}$, ill-defined. Q_1 can be zero because the small and time-accurate time-steps used by classical fractional-step methods means that nonlinearity will not cause a convergence issue. Similarly, the projection onto the constraint (step 3 of equation (8)) is small for small time-steps, so $Q_3 = I$ (identity matrix) is sufficient.

Since $F = [\frac{M^{n+1}}{\Delta t} + \sum_f A^f(\boldsymbol{u}\cdot\boldsymbol{n}\rho^f + \mu\frac{\partial}{\partial n})]$, the approximation (equation (9)) being used for fractional-step methods is $\begin{bmatrix} F & ZG \\ D & 0 \end{bmatrix} \begin{pmatrix} \delta \boldsymbol{u} \\ \delta p \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}_u \\ r_p \end{pmatrix}$ where $Z = I + \Delta t(M^{n+1})^{-1}\sum_f A^f(\boldsymbol{u}\cdot\boldsymbol{n}\rho^f + \mu\frac{\partial}{\partial n})]$ is reasonably close to the identity matrix. Note that the size of the dimensionless matrix entries in Z that are due to the advection term are $O(\frac{\Delta t \boldsymbol{u}\cdot\boldsymbol{n}}{\Delta x})$ and those due to the diffusion term are $O(\frac{\mu\Delta t}{\rho\Delta x^2})$. If the method is time accurate, then all these matrix entries are on the order of 1 or less, because those dimensionless parameters also define time-accuracy.

The classical fractional-step method does not typically use iteration (though it can). So one application of equation (8) (steps 1 through 4) yields the solution at the next time step after the corrections are added to the time n values.

When there is no iteration, the extra terms in Z beyond the identity matrix lead to first-order accuracy in time. This extra error is referred to as splitting error, and higher order time-accuracy may be obtained by improving the approximation [12]. For example, $Q_2 = \frac{M^{n+1}}{\Delta t}(2\frac{M^{n+1}}{\Delta t} - F)^{-1}\frac{M^{n+1}}{\Delta t}$ gives second order accuracy (but can produce an indefinite pseudo-Laplacian in step 2). Third-order (and positive definiteness) can be achieved using $Q_2 = \frac{M^{n+1}}{\Delta t}(3\frac{M^{n+1}}{\Delta t} - 3F + F\frac{\Delta t}{M^{n+1}}F)^{-1}\frac{M^{n+1}}{\Delta t}$. In practice, iteration is probably easier than including these approximations inside the pseudo-Laplacian in step 2, because these make the pseudo-Laplacian asymmetric when advection is present in F.

5.2. SIMPLE

The diagonal of the matrix F is a critical parameter for SIMPLE and its derivatives. We will call this diagonal matrix, F_D. The diagonal matrix scales like $F_D \approx M^{n+1}[\frac{1}{\Delta t} + O(\frac{\boldsymbol{u}\cdot\boldsymbol{n}}{\Delta x}) + O(\frac{\mu}{\rho \Delta x^2})]$. If central differencing is used for advection there is zero advection contribution to the matrix diagonal. However, most commercial and open-source implementations of SIMPLE use full upwinding for the advection term in F and then the diagonal of the matrix gets an advection contribution from all downwind faces. We recommend using the upwind version of F_D even if central differencing is actually used in F (though this means F_D is now only a representative large frequency, and not precisely the diagonal of F).

SIMPLE introduces two relaxation parameters. The velocity under-relaxation ω_u is typically less than, but close to, 1, and the pressure relaxation ω_p is typically greater than, but close to, zero. The default values are usually 0.7 or 0.8 for ω_u and 0.3 or 0.2 for ω_p. The choice of $\omega_p = 1 - \omega_u$ is commonly regarded as optimal.

The velocity under-relaxation term $Q_1 = \frac{1-\omega_u}{\omega_u} F_D$ is necessary because F is close to singular in the steady-state limit and non-linear iterations require some under-relaxation. It is appropriate to base the amount of under-relaxation on the problem itself. When equation (1) is linear (such as for Stokes flow with no advection), velocity under-relaxation is not necessary for the non-linearity, but is still required to converge the segregated iterative method that solves the dynamics and the constraint separately.

SIMPLE also uses the diagonal of the matrix F for the approximation of Q_2. However, the two approximation matrices actually have actions that are entirely different, and the overlap in the two approximations is not required (and leads to confusion in some analyses of SIMPLE).

For SIMPLE, the iteration matrix is now, $\begin{bmatrix} \frac{1-\omega_u}{\omega_u} F_D + F & (\frac{1-\omega_u}{\omega_p} I + \frac{\omega_u}{\omega_p} F F_D^{-1}) G \\ D & 0 \end{bmatrix} \begin{pmatrix} \delta \boldsymbol{u} \\ \delta p \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}_u \\ r_p \end{pmatrix}$. Since F is diagonally dominant, the matrix $F F_D^{-1}$ has entries which are 1 on the diagonal and have magnitude less than 1 for the off-diagonal contributions. The error matrix $Z = (\frac{1-\omega_u}{\omega_p} I + \frac{\omega_u}{\omega_p} F F_D^{-1})$ is therefore also always equal to $\frac{1}{\omega_p}$ on the diagonal and has magnitude less than $\frac{\omega_u}{\omega_p}$ on the off-diagonals. While $\omega_p \to 1$ appears to be attractive, it is not stable. On the other hand, the common choice of $\omega_p = 1 - \omega_u$ gives an error term, Z, for the pressure gradient that is a perturbation of the identity tensor (similar to the classical fractional-step method) that is always stable (if ω_u is small enough).

The approximation $\frac{1-\omega_u}{\omega_u} F_D$ in the momentum equation (top left block of the matrix) can be interpreted as a pseudo-time step that varies locally with the mesh size. Define a local time step τ as $\frac{M^{n+1}}{\tau} = \frac{1-\omega_u}{\omega_u} F_D$ then, $\frac{1}{\tau} \approx \frac{1-\omega_u}{\omega_u}[\frac{1}{\Delta t} + O(\frac{\boldsymbol{u}\cdot\boldsymbol{n}}{\Delta x}) + O(\frac{\mu}{\rho \Delta x^2})]$ and the local time step is based on the sum of the local diffusion and CFL numbers, $[\frac{\tau}{\Delta t} + O(\frac{\boldsymbol{u}\cdot\boldsymbol{n}\tau}{\Delta x}) + O(\frac{\mu\tau}{\rho \Delta x^2})] = \frac{\omega_u}{1-\omega_u}$. For $\omega_u = 0.8$ the total-CFL number (left hand side of the equation) using the effective local time step is always required to be equal to 4. Fine meshes therefore always result in smaller pseudo-time steps and produce slower convergence rates (as seen in Fig. 1a). This behavior, though undesirable, is inherent to all of the classical segregated methods. All classical segregated methods are therefore stepping in pseudo time to steady state, using local timesteps set by the local CFL number.

We note that, in SIMPLE, the pseudo-time step varies spatially but is roughly inversely proportional to the largest singular values of the matrix F. In contrast, the M-method proposed in this paper attempts to choose the local time step so that it is roughly inversely proportional to the smallest non-zero singular value of F. The smallest singular value of F does not vary with mesh size (as shown in [27]) and allows for much larger time-steps while still allowing convergence. However, the smallest singular value is much more difficult to estimate, which possibly explains why this approach has not been proposed before.

5.3. SIMPLEC

SIMPLEC uses a different Q_2 than SIMPLE, but it performs similarly in the steady-state limit. It uses the approximations, $Q_1 = \frac{1-\omega_u}{\omega_u} F_D, \quad Q_2 = Q_1 + \frac{M^{n+1}}{\Delta t}, \quad Q_3 = \omega_p I$. This version of Q_2 is a result of lumping the $Q_1 + F$ matrix rather than taking its diagonal as in SIMPLE. Lumping adds all the values in a matrix row together. Lumping is the typical way that finite element mass matrices are diagonalized to preserve the underlying essential physics. Lumping is the equivalent of multiplying the matrix by an array of ones. A constant field (equal to 1) applied to the advection and diffusion terms produces zero result for all internal cells and cells with Neumann boundary conditions. The advection and diffusion terms therefore do not contribute to Q_2.

The variation of SIMPLEC used in this paper ignores the advection and diffusion terms entirely in computing Q_2, even on the boundaries. This is perhaps even more physical than pure lumping. In this paper, the SIMPLEC (and M-method) Q_2 is purely the physical time stepping term and the pseudo-time stepping term added together, to make a total time stepping term. This is a straight-forward extension of what the classical fractional-step method does for Q_2.

The iteration matrix for SIMPLEC is

$$\begin{bmatrix} \frac{1-\omega_u}{\omega_u} F_D + F & \frac{1}{\omega_p}\{I + (F - \frac{M^{n+1}}{\Delta t})(\frac{1-\omega_u}{\omega_u} F_D + \frac{M^{n+1}}{\Delta t})^{-1}\} G \\ D & 0 \end{bmatrix} \begin{pmatrix} \delta \boldsymbol{u} \\ \delta p \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}_u \\ r_p \end{pmatrix}$$

In the steady state limit $Z = \frac{1}{\omega_p}\{I + \frac{\omega_u}{1-\omega_u} F F_D^{-1}\}$ which is the same as for SIMPLE if $\omega_p^{SIMPLE} = \omega_p^{SIMPLEC}(1-\omega_u)$. This means that the standard value for pressure under-relaxation in SIMPLEC is $\omega_p^{SIMPLEC} = 1$. Note that pressure over-relaxation (up to a value of 2) can be used with SIMPLEC if desired, but normally the under-relaxation of pressure is 1 (no under- or over-relaxation at all).

In the unsteady limit, SIMPLEC and SIMPLE differ. SIMPLEC naturally goes to the classical unsteady fractional-step method with $Z^{FS} = I + (F - \frac{M^{n+1}}{\Delta t})\frac{\Delta t}{M^{n+1}}$ if both under-relaxation parameters go to 1 in the unsteady limit. SIMPLE does not transition to the unsteady case as trivially as SIMPLEC does and produces less accurate answers than both the fractional-step method and SIMPLEC at CFL numbers on the order of 1. For this reason, the M-method derived below more closely resembles SIMPLEC than SIMPLE.

5.4. SIMPLER

SIMPLER solves two elliptic pseudo-Poisson equations per iteration, resulting in roughly twice the cost of SIMPLE and SIMPLEC. It must therefore converge twice as fast to be competitive. It performs a second elliptic solve (in step 4 of equation (8)) in an attempt to improve the pressure solution. If the pressure error is not controlling the convergence (which is often the case), then SIMPLER is not advantageous and can be more computationally expensive.

SIMPLER attempts to approximate, $Q_3 = (D Q_4^{-1} G)^{-1} D Q_4^{-1}(Q_1 + F) Q_2^{-1} G$. Both SIMPLER and PISO also both use $Q_4 = Q_2$. However, any Q_4 is technically possible, so that fact has been highlighted in the equation. The only difference between SIMPLER and SIMPLE is the Q_3, so this idea can therefore also be applied to the SIMPLEC versions of Q_3, which has been done in reference [24]. The actual implementation of SIMPLER is far less complicated than implied by the complexity of the expression for the target Q_3. Step 4 of equation (8) (the segregated approach) becomes an elliptic equation (similar to step 2, but with a different right-hand-side),

$$(D Q_4^{-1} G)\delta p = D Q_4^{-1}(Q_1 + F) Q_2^{-1} G \delta p^* = D Q_4^{-1}(Q_1 + F)(\delta \boldsymbol{u}^* - \delta \boldsymbol{u}) = D Q_4^{-1}[\boldsymbol{r}_u - (Q_1 + F)\delta \boldsymbol{u}].$$

The classic version of SIMPLER neglects Q_1 in the right-hand-side (and uses $Q_4 = Q_2$), so step 4 becomes, $(D Q_2^{-1} G)\delta p = D Q_2^{-1} \boldsymbol{r}_u^{new}$. Classic SIMPLER also presents this as step 0 of the algorithm (and step 4 is dropped) but, since it is an iterative loop, that ordering detail is immaterial in actual practice.

Although SIMPLER was derived intuitively and not using matrix algebra, the linear algebra iteration matrix shows why this approximation is a good one for the pressure. The approximation term for the gradient now has an error term of $Z = (Q_1 + F) Q_2^{-1} G [D Q_4^{-1}(Q_1 + F) Q_2^{-1} G]^{-1} D Q_4^{-1}$. Although complex, this term is actually a pseudo-identity matrix. If D and G could be inverted, then Z would be the identity matrix and the gradient term in the iteration matrix would be exact. Note that $D Q_4^{-1} Z = D Q_4^{-1}$ and $Z(Q_1 + F) Q_2^{-1} G = (Q_1 + F) Q_2^{-1} G$ is the sense in which Z is formally a pseudo-identity matrix. Any vector array that can be formed by $\boldsymbol{v} = (Q_1 + F) Q_2^{-1} G \phi$, for any choice of the array phi, will see multiplication by Z as an identity matrix.

Note that SIMPLER and the extra elliptic solve improves the pressure approximation significantly but does not remove the error induced by Q_1 in the momentum equation. This, not the pressure solution, is the pseudo-time step approximation that limits time steps to a total CFL of roughly 4, resulting in a significantly slowed convergence rates on fine meshes on all segregated solvers. From the linear algebra perspective, SIMPLER addresses the wrong problem.

5.5. M-method

The M-method's primary improvement comes from the approximation $Q_1 = \alpha F_D$. The paper will also discuss an improved version of the pressure approximation

$$Q_3 = \omega_p (I - \beta \tilde{D} \frac{\mu}{Q_2} \tilde{G}) \tag{10}$$

which is inexpensive to compute and does not require an extra elliptic solve like SIMPLER does.

The key aspect of the M-method (derived in section 6) is the automatic calculation of the parameter α which for steady state is,

$$\alpha = \frac{1}{2m}\left(\frac{\sum F_D^{-1}\boldsymbol{r}_u \cdot F_D^{-1}\boldsymbol{r}_u V_c}{\sum \delta \boldsymbol{u}_u^* \cdot \delta \boldsymbol{u}_u^* V_c}\right)^{1/2} \tag{11}$$

which is calculated using data saved from step 1 of equation (8) of the prior iteration. V_c is the cell volume, and the summation is over all the cells in the domain, so the dimensionless α is a single number (roughly proportional to the inverse of the condition number of F). On the first iteration, $\alpha = 0.5$ is used in the results in this paper (numerical tests, not shown, suggest that any value in the range 0 to 1 works). The parameter $m = 2$ is used in this work, however it can be adjusted by the user. Theoretically m must be greater than 1, and 1 will always work (but converges slightly slower). Too large a value for m (greater than 10), could slow the convergence rates or potentially cause a lack of convergence (see

section 8). The value for ω_p is safely set to 1 (like in SIMPLEC), but it is more aggressively set to 1.8 (under the neutral stability limit of 2) for all the cases tested in this paper (see section 8).

The approximation for Q_3 has a viscous correction. But many of the results (such as Fig. 1b) are shown using $\beta = 0$. The formula for Q_3 is not a critical aspect of the M-method. Equation (11) is the key result. The viscous correction in Q_3 has tildes on the divergence and gradient operators because this matrix operates on quantities like pressure corrections, which may require different approximations for those operators than those used for the velocity unknowns.

Unlike classical segregated solvers, the M-method self-calculates the velocity under-relaxation factor and varies it with every iteration. The effective under-relaxation is therefore $\omega_u = \frac{1}{1+\alpha}$. When α varies from 0 to infinity, the effective under-relaxation varies from 1 to 0 (which is the correct bounds). In most test cases $\alpha \ll 1$. The physical meaning of α is that it is an easily-computed approximation for the smallest singular value of FF_D^{-1}. This causes the pseudo-time step to be proportional to the smallest singular value of the problem (which is the largest physical timescale in the problem). So the pseudo-time steps are of the order of the flow evolution and have no relationship to the mesh size. This makes the M-method converge at a nearly mesh independent rates.

Note that the version of the M-method presented herein is closest to an extension of SIMPLEC because of the form used for $Q_2 = \alpha F_D + \frac{M^{n+1}}{\Delta t}$ which is a matrix lumping (which is typically much smaller than the matrix diagonal). However, equation (11) can be applied to any segregated solver to significantly improve its convergence rates.

The objective of finding a useful functional expression for the velocity under-relaxation parameter has been attempted in the past by [25] and [26]. However, none of those prior methods has any overlap or similarity with the M-method. Optimization methods have also been used for segregated methods in the past. Viguerie, A. Veneziani [22] lagged the some of the advection term in their approach and found an optimal value for the amount of advection lagging.

6. Error analysis

In this section, the formula for calculating the optimal under-relaxation parameter (equation (11)) is derived. Given a matrix problem, A, which iterates using an approximate inverse $\hat{A}\delta x^{n+1} = r^n$ (such as the segregated methods), then $\hat{A}(x^{n+1} - \bar{x} + \bar{x} - x^n) = A(\bar{x} - x^n)$ and therefore $e^{n+1} - e^n = -\hat{A}^{-1}A(e^n)$, which produces the well known result for iterative methods that,

$$e^{n+1} = (I - \hat{A}^{-1}A)e^n \tag{12}$$

The M-method chooses the under-relaxation parameter to reduce the error as much as possible on each iteration. It therefore focuses on the properties of the matrix $I - \hat{A}^{-1}A$.

For segregated methods, the approximate matrix is given by equation (6) and is repeated below,

$$\hat{A} = \begin{bmatrix} Q_1 + F & 0 \\ D & -L \end{bmatrix} \begin{bmatrix} I & Q_2^{-1}GQ_3^{-1} \\ 0 & Q_3^{-1} \end{bmatrix}$$

where $L = DQ_2^{-1}G$ is the pseudo-Laplacian that determines the first pressure correction, δp^*.

The inverse of a block LU decomposition can be computed explicitly and is,

$$\hat{A}^{-1} = \begin{bmatrix} I & -Q_2^{-1}G \\ 0 & Q_3 \end{bmatrix} \begin{bmatrix} (Q_1 + F)^{-1} & 0 \\ L^{-1}D(Q_1 + F)^{-1} & -L^{-1} \end{bmatrix} \tag{13}$$

So the error reduction matrix is,

$$I - \hat{A}^{-1}A = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} - \begin{bmatrix} I & -Q_2^{-1}G \\ 0 & Q_3 \end{bmatrix} \begin{bmatrix} (Q_1 + F)^{-1} & 0 \\ L^{-1}D(Q_1 + F)^{-1} & -L^{-1} \end{bmatrix} \begin{bmatrix} F & G \\ D & 0 \end{bmatrix}$$

Multiplying everything gives a daunting expression,

$$= \begin{bmatrix} I - (Q_1 + F)^{-1}F + Q_2^{-1}GL^{-1}D\{(Q_1 + F)^{-1}F - I\} & -(Q_1 + F)^{-1}G + Q_2^{-1}GL^{-1}D(Q_1 + F)^{-1}G \\ Q_3L^{-1}D\{I - (Q_1 + F)^{-1}F\} & I - Q_3L^{-1}D(Q_1 + F)^{-1}G \end{bmatrix}$$

which can be simplified slightly to give,

$$\begin{pmatrix} e_u^{n+1} \\ e_p^{n+1} \end{pmatrix} = \begin{bmatrix} \{I - Q_2^{-1}GL^{-1}D\}(Q_1 + F)^{-1}Q_1 & -\{I - Q_2^{-1}GL^{-1}D\}(Q_1 + F)^{-1}G \\ Q_3L^{-1}D(Q_1 + F)^{-1}Q_1 & I - Q_3L^{-1}\{D(Q_1 + F)^{-1}G\} \end{bmatrix} \begin{pmatrix} e_u^n \\ e_p^n \end{pmatrix} \tag{14}$$

Note that Q_3 (the pressure correction approximation and pressure under-relaxation) only appears in the lower row, and in the pressure error. But Q_1, the velocity under-relaxation appears in every block of the matrix. Q_2 which is an easy to invert approximation for $(Q_1 + F)$ appears explicitly only in the top row and both times only in the projection matrix, $\{I - Q_2^{-1}GL^{-1}D\}$. But Q_2 is also imbedded into L which appears in all blocks.

The error reduction matrix can also be LU split though it is not clear if this is a useful rearrangement,

$$\begin{pmatrix} e_u^{n+1} \\ e_p^{n+1} \end{pmatrix} = \begin{bmatrix} (I - Q_2^{-1} G L^{-1} D) & 0 \\ Q_3 L^{-1} D & I \end{bmatrix} \begin{bmatrix} (Q_1 + F)^{-1} Q_1 & -(Q_1 + F)^{-1} G \\ 0 & I \end{bmatrix} \begin{pmatrix} e_u^{n} \\ e_p^{n} \end{pmatrix} \tag{15}$$

For the classical fractional-step method $Q_1 = 0$, and the error reduction matrix is,

$$\begin{pmatrix} e_u^{n+1} \\ e_p^{n+1} \end{pmatrix} = \begin{bmatrix} 0 & -\{I - Q_2^{-1} G L^{-1} D\} F^{-1} G \\ 0 & I - Q_3 L^{-1} \{D F^{-1} G\} \end{bmatrix} \begin{pmatrix} e_u^{n} \\ e_p^{n} \end{pmatrix} \tag{16}$$

and all the error comes from the pressure error convergence. However, the classical fractional-step method rarely iterates (to a final solution) and remember the n+1 index in these equations refers to the iteration number and not the time step.

For the M-method, substituting in for Q_1, allows a direct assessment of the influence of α on the error reduction,

$$\begin{bmatrix} \{I - Q_2^{-1} G L^{-1} D\} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} (\alpha F_D + F)^{-1} \alpha F_D & -(\alpha F_D + F)^{-1} G \\ Q_3 L^{-1} D (\alpha F_D + F)^{-1} \alpha F_D & I - Q_3 L^{-1} \{D (\alpha F_D + F)^{-1} G\} \end{bmatrix} \tag{17}$$

Because the left matrix is a projection (applying it twice is the same as applying it once) it has eigenvalues that are 0 or 1. It either kills error modes entirely or does nothing at all to them, so it does not affect convergence rates. The focus is therefore on the right matrix. The optimum α is the one that minimizes the maximum singular value of this matrix. Note that the two left blocks get smaller as α gets smaller (like the classical fractional-step limit), so the two right blocks are the issue when α gets small. If α gets small then the smallest singular value of F dominates the inverse in the two right blocks and can make those contributions greater than 1 (causing lack of convergence). It is therefore critical that αF_D be larger than, or the same size as, the smallest non-zero singular value of F. The "size" of the inverse of a matrix is proportional to the one divided by the smallest singular value of the matrix.

The lower right block matrix is $\approx I - Q_3 Q_2 (\alpha F_D + F)^{-1}$ if we ignore the influence of the pseudo-Laplacian and its inverse on the singular values (because they are a type of projection again). For the M-method this is $\approx I - Q_3 (\alpha F_D + \frac{M}{\Delta t}) F_D^{-1} (\alpha I + F F_D^{-1})^{-1}$ which becomes $\approx I - Q_3 (\alpha I + \frac{M}{\Delta t} F_D^{-1}) (\alpha I + F F_D^{-1})^{-1}$ and has a maximum singular value that is roughly proportional to $1 - \frac{\omega_p \{\alpha + \max(\frac{M}{\Delta t} F_D^{-1})\}}{\alpha + \sigma_{min}(F_D^{-1} F)}$ (for $\beta = 0$). A solution that nearly minimizes this expression (as well as the other three blocks in equation (17)) is

$$\alpha = \sigma_{min}(F_D^{-1}\{F - \frac{M}{\Delta t}\}) \quad \text{and} \quad \omega_p = 1 + \frac{\alpha}{\sigma_{\min}(F_D^{-1} F)}. \tag{18}$$

This approaches the classical fractional-step method in the time-accurate limit where $F \approx \frac{M}{\Delta t}$. In the time-accurate unsteady limit, the unsteady term in the matrix F dominates and $F \to \frac{M}{\Delta t}$ (which is the important property of F used by classical unsteady fractional step methods). This means that equation (18) makes $\alpha \to 0$ and the classical fractional step method is directly recovered in the time-accurate unsteady limit. For the steady-state limit, equation (18) becomes $\alpha = \sigma_{min}(F_D^{-1} F)$ and $\omega_p = 2$.

In order to impose slower variation in the parameters from one iteration to the next, and also in order to make the calculation less expensive by using the residual which is already computed we will use,

$$\alpha_{new} = \frac{1}{2}[\alpha_{old} + \sigma_{min}(F_D^{-1}\{F - \frac{M}{\Delta t}\})] = \frac{1}{2}\sigma_{min}(F_D^{-1}\{\alpha_{old} F_D + F - \frac{M}{\Delta t}\}) \tag{19}$$

Then, to estimate the minimum singular value we use a Rayleigh quotient estimate. This estimation approach to finding the minimum singular value, and the choice of volume norms, is described in detail in Rao [27]. Using that estimate gives,

$$\alpha = \frac{||F_D^{-1}\{r_u - \frac{M}{\Delta t}\delta u\}||^V}{2m||\delta u||^V} \tag{20a}$$

and

$$\omega_p = 1 + \frac{||F_D^{-1}\{r_u - \frac{M}{\Delta t}\delta u\}||^V}{||F_D^{-1} r_u||^V} \tag{20b}$$

where $m > 1$. This paper uses a value of $m = 2$ unless otherwise noted, and section 8 shows that the method is not sensitive to this choice (m can vary from 1-10 with little change in the results). These norms can be calculated with a negligible cost. The method assumes that these iteration parameters change slowly and can therefore be lagged from the previous iteration, and all the test cases (and Fig. 13) support this assumption.

We note that the work of Rao [27] uses very similar information as that used in equation (20), to construct an error estimator for the velocity. This provides indirect evidence that the optimal under-relaxation parameter is closely connected with error estimation, a result that is not surprising since it is designed to reduce error maximally at each iteration.

7. Results

Unless otherwise specified, we will use $\beta = 0$, $m = 2$ and $\omega_p = 1.8$, and focus on the pseudo-time stepping. We will sometimes compare with SIMPLE (with $\omega_u = 0.8$ and $\omega_p = 0.2$) which is identical for these test cases to SIMPLEC (with $\omega_u = 0.8$ and $\omega_p = 1$). A second order finite volume code is used in the computations. The advection term uses central differencing. There are no lagged terms (except the pressure) in the residual calculation.

The error norm is calculated by running each simulation out to 1500 iterations using the fastest converging method, and then saving that solution. This solution is considered to be the "exact solution". The test case is then run again and the square of the error norm at each iteration is calculated using a midpoint approximation for the volume integral of the squared error,

$$e_n = \left(\frac{\sum V_c (\boldsymbol{u}_n - \boldsymbol{u}_{exact}) \cdot (\boldsymbol{u}_n - \boldsymbol{u}_{exact})}{\sum V_c} \right)^{1/2} \tag{21}$$

The error tracked in this paper is therefore the iteration error. This is the error compared to the exact solution to equation (1) (the discrete problem). Our error does not include the error in equation (1) when it is compared to the exact PDE that it discretizes.

7.1. Reynolds number

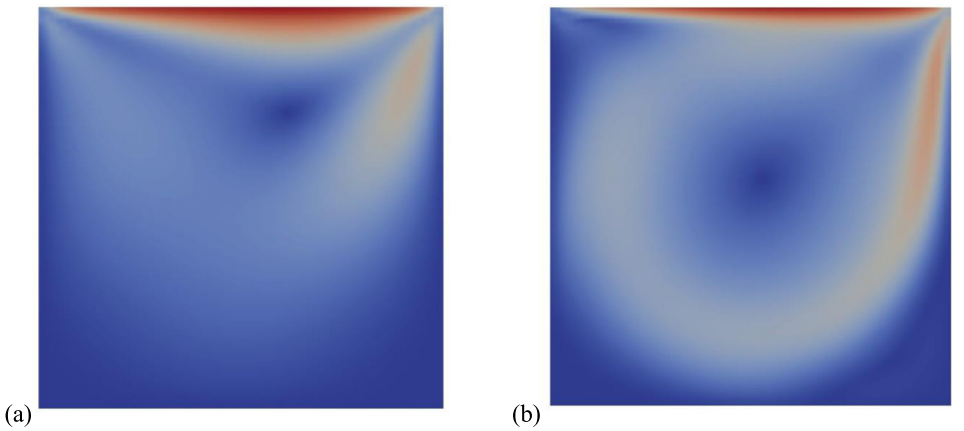

Fig. 3. Magnitude of the velocity for steady flow in a driven cavity. (a) Reynolds number of 100. (b) Reynolds number of 1000.

The canonical test case chosen for the purposes of demonstrating the method is flow in a lid driven cavity already considered in Section 1. This case is sufficiently complex to document the essential properties and advantages of the method.

Fig. 3 shows the solutions for the cavity flow at Reynolds number of 100 and 1000. The higher Reynolds number case is the solution corresponding to the errors shown previously In Fig. 1. The error for the first solution (Fig. 3a), the Re=100 test case, is shown in Fig. 4 which shows the error convergence of SIMPLE (or SIMPLEC) and for the M-method. The mesh dependence of the M-method is slightly stronger at lower Reynolds numbers. The M-method can be made much less

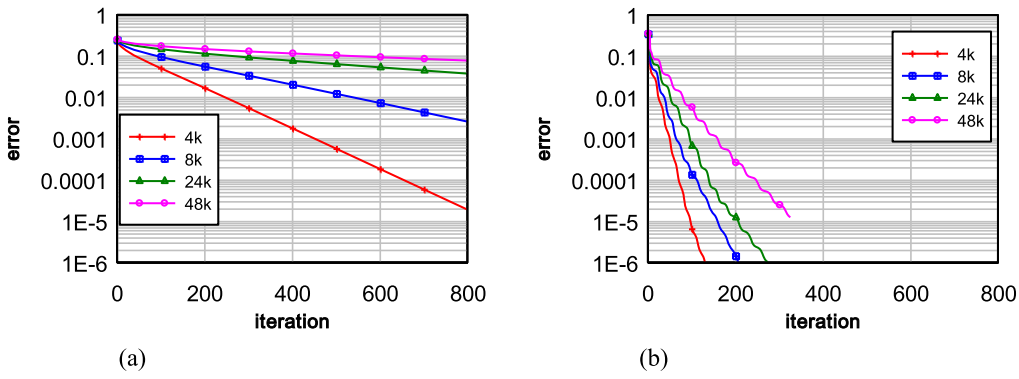

Fig. 4. Steady cavity flow at Reynolds number of 100. Error versus iteration number for a variety of nearly uniform triangular meshes. Mesh size is given by the number of triangles. (a) SIMPLE algorithm showing strong mesh dependence. (b) M-method showing much milder mesh dependence.

mesh dependent at low Reynolds numbers by implementing the viscous dependent Q_3 term. However, in Fig. 4 this is not turned on, (i.e., $\beta = 0$), in order to focus on the effects of the automatically calculated under-relaxation coefficient (given by equation (11)). SIMPLE again shows strong mesh dependence in its convergence rate. A 12x increase in unknowns requires 12x more iterations for the same error level. Other mesh independent segregated methods exist for Stokes flow and can be extended to low Re number situations, but they all require a difficult to invert approximation for Q_2 and a very expensive pseudo-Laplacian (Schur complement) solve. In contrast the M-method adds no perceptible additional computational cost to classical segregated methods.

At a Reynolds number of 100, the M-method converges more quickly (roughly twice as fast as the Re=1000 case) and the SIMPLE method converges more slowly (about half as fast as the Re=1000 case). This is because SIMPLE uses the effective pseudo-time step of $\tau = \frac{\omega_u}{1-\omega_u} \frac{1}{[\frac{1}{\Delta t} + O(\frac{\mathbf{u}\cdot\mathbf{n}}{\Delta x}) + O(\frac{\mu}{\rho \Delta x^2})]}$. So the larger the viscosity, the smaller the time step must be to enforce the equality. Smaller pseudo-time steps imply slower convergence. The M-method becomes faster at lower Reynolds numbers because the M-method pseudo-time-step scales with the physical timescale of the problem. As Fig. 3 shows, the lower Reynolds number solution is smoother and easier to solve, and has a bigger large timescale.

The solution time for all these cavity flow simulations is 1.17×10^{-5} seconds plus or minus 11% per solution unknown per iteration, on one core of an Intel i7-6700 CPU operating at 2.6 GHz. The solution time is dominated (92%) by the two linear solvers in the first two steps of equation (8). BiCGstab2 with diagonal preconditioning is used for the first (velocity) solve and Conjugate Gradients with diagonal preconditioning is used for the second (pressure) solve. There is no measurable difference in cost per iteration per unknown between the simulations in Fig. 4a (using SIMPLE) and Fig. 4b (using the M-method). Fluctuations in the timings between identical runs are the same as fluctuations between SIMPLE and the M-method. Since the information needed to compute equation (20) is already available the cost of one dot product to find the optimal under-relaxation value is trivial compared to the two linear matrix inversions.

7.2. Non-uniform mesh

The approximation matrices Q_1 and Q_2 are diagonal matrices for all the classical segregated methods. On a uniform mesh, the values in these diagonal matrices are all roughly the same size, so these matrices are close to proportional to identity matrices. Non-uniform meshes change this property and force the values in the diagonal matrices to vary significantly. The uniform mesh used in the previous test case and the non-uniform mesh used in this test case are shown in Fig. 5 for two of the meshes that were tested.

Fig. 6 shows the error for the Reynolds number of 1000 cavity flow case on a series of non-uniform meshes. This figure corresponds to Fig. 1 which is at the same Reynolds number but using uniform meshes. For the same size mesh, the non-uniform meshes converge slightly faster than the uniform meshes for both SIMPLE and the M-method. This is because the mesh is refined where the solution needs it to be.

For the M-method, the convergence is again nearly independent of mesh size. The mesh-dependence of the SIMPLE method on the non-uniform mesh is less obvious, because the smallest mesh (of 2k triangles) converges anomalously (due to its coarseness). However, the difference in convergence rates between the larger 6k and 16k meshes is again nearly directly proportional to their size difference.

7.3. Optimal under-relaxation

In this section we endeavor to show that the expression given by equation (11) is nearly optimal. To do this, the M-method result is compared to a variety of SIMPLEC simulations with differing ω_u (and $\omega_p = 1$ in all cases). This is shown

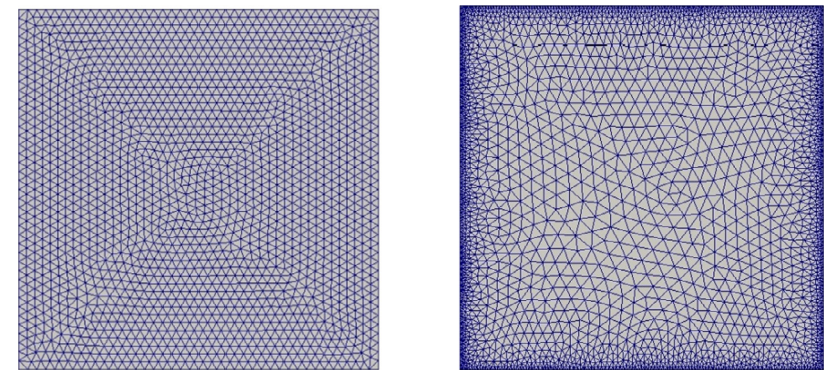

Fig. 5. Cavity flow meshes. (a) 4k uniform triangles. (b) 6k non-uniform triangles.

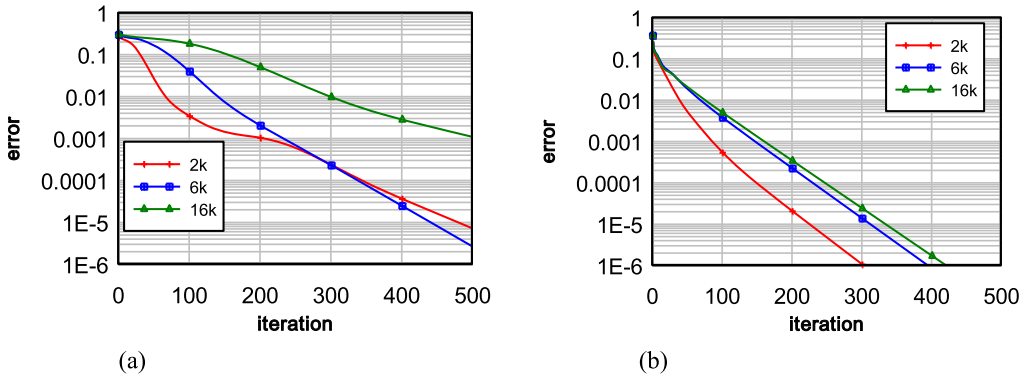

Fig. 6. Non-uniform mesh. Steady cavity flow at Reynolds number of 1000. Error versus iteration number. (a) SIMPLE algorithm showing mesh dependence. (b) M-method showing much milder mesh dependence.

in Fig. 7 for the cavity flow at Re=100 on the 8k mesh, and Re=1000 on the 24k mesh. It is seen that the default under-relaxation value for SIMPLEC of $\omega_u = 0.8$ (magenta line with $+$) is far too conservative a value for these two problems.

In Fig. 7a, the optimal under-relaxation parameter is about 0.97. At higher values of the under-relaxation parameter (than the optimum) the early iterations (governed by velocity error) converge faster but the later iterations (governed by the pressure error) converge slower. This causes an elbow in the convergence plots that gets more extreme as the under-relaxation parameter gets closer to 1 and farther from its optimum value. The analysis in section 6 mathematically explain these visual observations. It shows that there are two different errors, velocity error and pressure error, which are coupled to each other. The elbow forms when the velocity error is being reduced faster at the expense of the pressure error being reduced more slowly. The pressure error is typically smaller initially so the fast convergence is seen initially. But eventually the slower converging pressure error is all that is left and it dominates the convergence. Because of the error coupling, the pressure error still shows up within the velocity convergence at every iteration.

The M-method (solid black line) is close to the optimum under-relaxation value. We define the optimum under-relaxation as the value that causes the fastest convergence but yet does not cause an "elbow". This is optimum in the sense that it tends to be the best overall value if the target error level is not specified.

Note that the M-method has an under-relaxation value that varies mildly with iteration number and is not perfectly constant (like it is in SIMPLEC). So the M-method and the SIMPLEC cases never will be identically the same.

The higher Reynolds number and finer mesh case (Fig. 7b) shows similar behaviors. The optimum value of the under-relaxation parameter is now about 0.97. The M-method is again close to the optimum convergence rate. The observed trend is that for the same physical problem the optimal relaxation parameter tends to get closer to 1 as the number of mesh unknowns increases. However, depending on the problem, the optimal under-relaxation value can vary in practice anywhere from 0.5 to 1.

In general, the optimal under-relaxation parameter depends on both the mesh and on the problem. The M-method gets close to predicting that optimal value on its own by using prior iteration information (equation (11)). In these two cases, the M-method (or the optimal under-relaxation value) leads to a number of iterations (for the same error level) that are an order of magnitude fewer than the standard SIMPLE or SIMPLEC methods (magenta lines with x symbols).

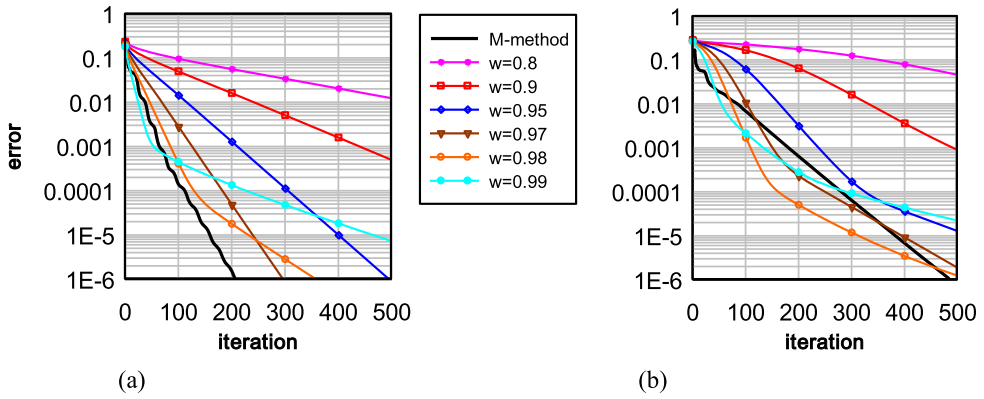

Fig. 7. SIMPLEC with various under-relaxation values compared to the M-method (solid black line). (a) 8k uniform mesh and Re = 100. (b) 24k uniform mesh and Re = 1000.

7.4. Diffusion correction

In this section, we evaluate the effect of the approximation matrix Q_3 that determines the pressure approximation from the projection pressure (that pushes the solution back onto the constraint). The segregated solvers SIMPLER and PISO are primarily focused on this part of the approximation. However, in contrast to those classical methods, the M-method will use an approximation for Q_3 that is far less costly than a second elliptic solve (as used with SIMPLER and PISO).

The proposed approximation is $Q_3 = \omega_p(I - \beta\tilde{D}\mu Q_2^{-1}\tilde{G})$ where $\beta = 1$. The matrix part of Q_3 adds a viscous diffusion term. Note that this diffusion term acts on a scalar pressure, rather than a vector velocity, so the gradient and divergence operators in this diffusion term may be slightly different than in the matrix F, but the goal is to make them as similar as possible.

The effect of this addition to the M-method ($\beta = 1$) is shown in Fig. 8. Fig. 8a shows all the meshes (except the very coarse non-uniform mesh) for the mildly viscous Re = 100 case. This plot corresponds to Fig. 4b which shows the M-method with $\beta = 0$. The extra diffusion term, accelerates the convergence (about 2x faster than $\beta = 0$) and reduces the mesh dependence. The M-method is now converging this problem to an error of 10^{-6} in roughly 100 iterations almost irrespective of the mesh size.

Fig. 8b shows the effect of the diffusion term on the higher Re = 1000 case. This figure corresponds to Fig. 1b (which shows the M-method with $\beta = 0$). Now the viscous term is 10x smaller than in the Re=100 case and the effect of the viscous term in Q_3 on the convergence rates is much smaller (less than 10% faster convergence rates).

The expression for Q_3 was developed by considering the linear algebra expression given by equation (9), that describes what the segregated methods are actually solving. The segregated methods end up using a pressure gradient operation that looks like $(Q_1 + F)Q_2^{-1}GQ_3^{-1}$ when it should just be G. Assume for the moment that the matrix G commutes with the normalized advection diffusion matrix $Q_1Q_2^{-1} + FQ_2^{-1}$. Then $G(Q_1 + F)Q_2^{-1}Q_3^{-1}$ equals G exactly if $Q_3 = (Q_1 + F)Q_2^{-1}$. This suggests that a good approximation for Q_3 might be to simply apply the (pre-normalized) advection/diffusion matrix. This approximation is explicit, fast, and not elliptic, and what we propose for Q_3.

However, in the M-method, only the diffusion portion of F is used in Q_3. This is because only the diffusion term in F commutes with the gradient operator in the continuous (PDE) limit, and this is an important component of the derivation. Even if the actual discrete diffusion and gradient matrices do not exactly commute, it is anticipated that they approximately commute (with an error that goes to zero as the mesh is refined). However, this commutation property does not exist for the advection term and adding advection to the expression for Q_3 (that is using the full $Q_3 = (Q_1 + F)Q_2^{-1}$) does not improve the method performance and tends to add some instabilities in certain cases.

The expression $Q_3 = \omega_p(I - \beta\tilde{D}\mu Q_2^{-1}\tilde{G})$ is ideal for low Reynolds number and Stokes flow cases, and leads to very low numbers of iterations. However, we do not have an equivalent solution that addresses the convergence rates of high Reynolds number (advection dominated) flows. Furthermore, such a solution may not be fundamentally possible because high Reynolds number drives solutions towards chaos, turbulence, and unsteadiness, and this method is seeking a generic fast path towards a steady state (which may not exist).

7.5. U-bend

In this test case, the M-method is tested on the flow through a channel that bends rapidly to reverse direction. The Reynolds number of the simulation is 1000 based on the inflow velocity and channel width. The flow is reasonably complex with a separation bubble and reverse flow after at the top of the turn. The solution (on the finest mesh) is shown in Fig. 9a. The mesh is non-uniform and refined around the separation point. The mesh with roughly 10k triangles is shown in Fig. 9b.

Fig. 10a shows the solution error for the finest 90k cell mesh as a function of iteration count for the M-method (red line) and SIMPLEC (blue dashed line) with the standard under-relation value of $\omega_u = 0.8$, as well as the SIMPLEC error

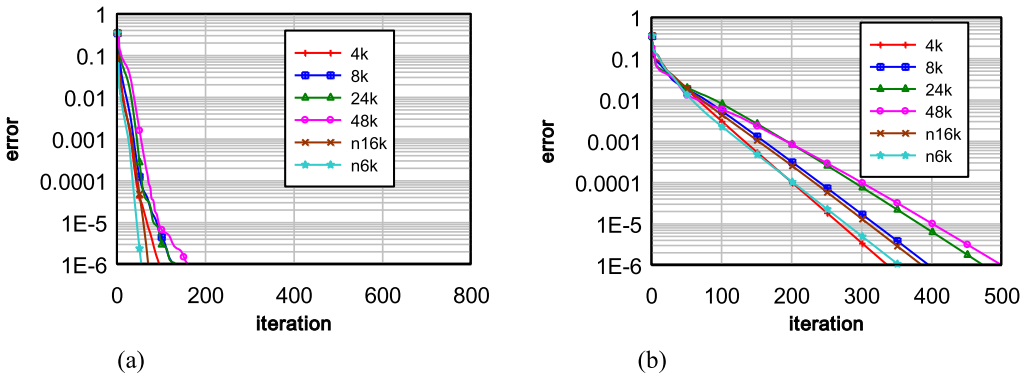

Fig. 8. M-method including the diffusion correction for cavity flow with uniform and non-uniform meshes of different sizes. (a) Re = 100 (using the same scale as Fig. 4). (b) Re = 1000 (using the same scale as Fig. 1).

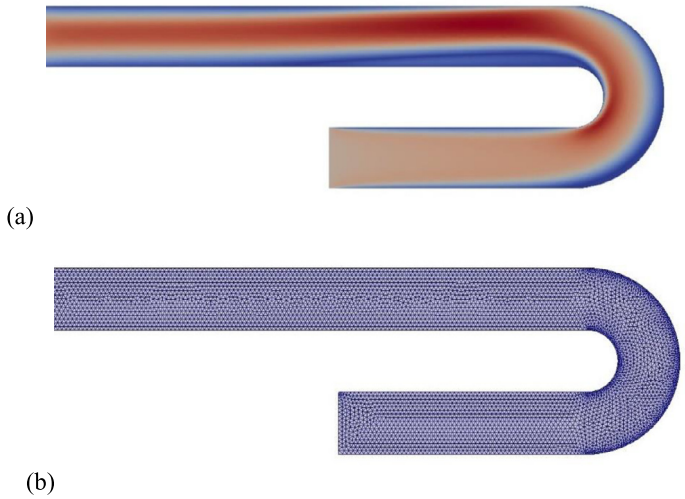

Fig. 9. (a) U-Bend velocity magnitude. Flow enters at the bottom and exits at the top. (b) Non-uniform unstructured triangular mesh using 10k triangles. The calculation uses this mesh refined twice to create 90k cells.

(green dotted line) when using the optimal under-relaxation value (of $\omega_u = 0.98$). The M-method automatic prediction for the optimal under-relaxation value as a function of the iteration is shown in Fig. 10b. At later iterations, the M-method prediction is very close to 0.98.

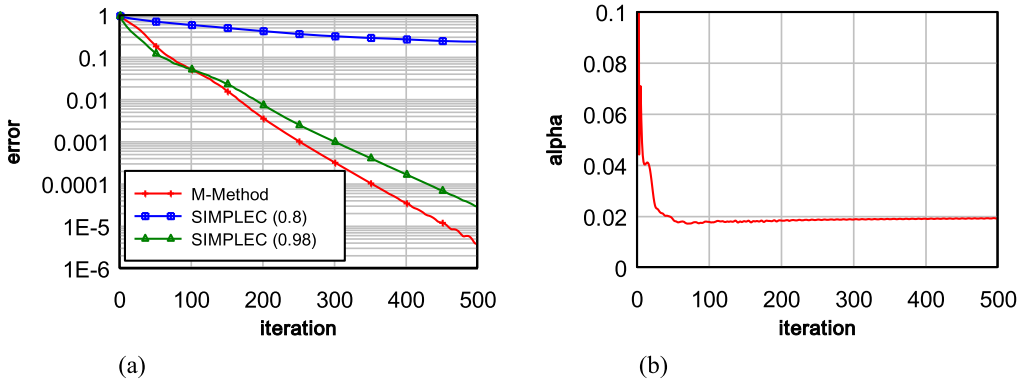

Fig. 10. Error for the U-Bend at Re=1000 on a 90k non-uniform triangle mesh. (a) Convergence error as a function of iterations. (b) Optimal under-relaxation, $\alpha = \frac{1-\omega_u}{\omega_u}$, as computed by the M-method. SIMPLEC run at the M-method's predicted optimal value achieves similar performance to the M-method.

7.6. NACA 2412 airfoil

In this test, the external flow over an airfoil was considered. Internal flows have their largest lengthscales (and therefore timescales) dictated by the external domain geometry. External flows have lengthscales (and timescales) that are dictated purely by the physics (boundary layer thickness in this test case). This test shows the ability of the M-method formula (eqn (20)) to predict the large timescale, and hence the correct under-relaxation parameter, for external flows. The flow is show in Fig. 11. The triangular mesh of 68k cells is non-uniform and is heavily refined at the leading and trailing edges.

Fig. 12 shows the error reduction as a function of the iteration count for the M-method (red line) and the SIMPLEC method (dashed blue line) with the standard relaxation value of 0.8.

8. Parameter sensitivity

The M-method has three parameters ($m = 2$, $\omega_p = 1.8$, $\beta = 1$) that do not need to be adjusted, but which can be changed if the user so desires. It is important to show that the method is not sensitive to these parameter choices.

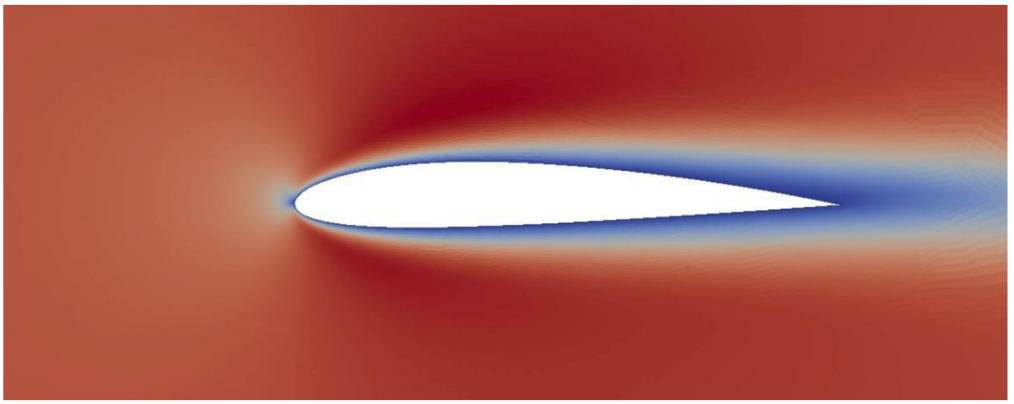

Fig. 11. Velocity magnitude around a NACA 2412 airfoil at 0 degrees of attack and at Re=1000 based on the free-stream velocity and chord length. This is a zoom on the airfoil, the full domain is 10x larger.

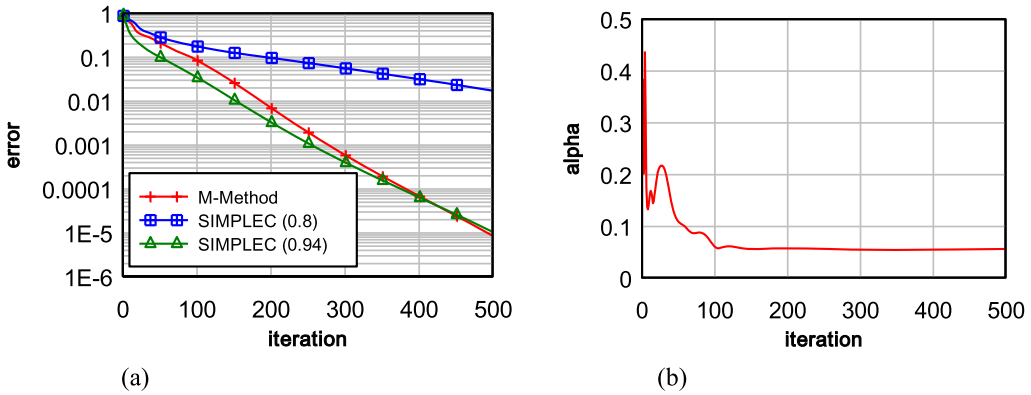

Fig. 12. Error for the NACA 2412 Airfoil at Re=1000 on a 68k non-uniform triangle mesh. (a) Convergence error as a function of iterations. (b) Under-relaxation, $\alpha = \frac{1-\omega_u}{\omega_u}$, as computed by the M-method. The solution takes about 100 iterations and α settles to a value of approximately 0.06 for this mesh.

8.1. Sensitivity to m

The parameter m, comes from the fact the Rayleigh quotient estimation of the smallest singular value (in equation (18)) is always an over-estimation. The estimate is always polluted by some of the other singular values, which are always larger than the minimum. The multiplicative factor between the actual minimum singular value and the estimate is the parameter m. It is shown below that the method is not sensitive to this choice, and the residuals rapidly adjust to produce roughly the same estimate for α irrespective of the choice for m. Fig. 13 shows this behavior.

Fig. 13a shows the error convergence for the cavity flow case at Re = 1000. The fairly fine 24k triangle mesh is used and the parameter m is varied from 1 (its theoretical lower limit) up to 10. The convergence is essentially the same irrespective of the value of m. This independence of the method from the value chosen for m is true for other meshes and Reynolds numbers, though not shown here. However, for some cases, using $m = 10$ is unstable at early iterations. A large value of m keeps the under-relaxation too close to zero (0 equals no under-relaxation) for too long during early iterations. To stay safe, this paper recommends a choice of a value of $m = 2$. The value of $m = 1$ is the theoretical limit and therefore unnecessarily low.

Fig. 13b shows the value of the under-relaxation parameter, α, for each case. Note that even though m appears in the denominator of equation (11) and differs in each run, the residuals quickly adjust to the different m to produce an α that is roughly the same size (within a factor of 2) for every value of m. The final value is remarkably similar for all choices of m and is equal to 0.026 in this test case. Since $\omega_u = \frac{1}{1+\alpha}$ this corresponds to $\omega_u = 0.9747$. Remember that, in Fig. 7b, this same case was tested with a variety of under-relaxation parameters and the optimal value of roughly 0.97 was determined via experiment. The M-method has automatically obtained this value via equation (11). The optimal value of ω_u (or α) varies with every problem and with every mesh for that problem, but the value of m=2 remains constant.

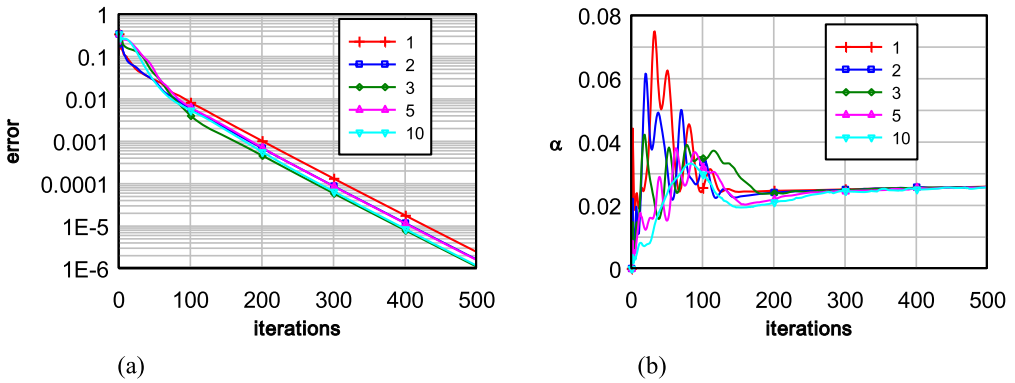

Fig. 13. M-method sensitivity to the parameter m. Re = 1000 and 24k uniform mesh. (a) Error convergence is very insensitive to the choice of m. (b) The method calculated value of α (equation (11)) for different values of m.

8.2. Pressure over-relaxation

The effect of the pressure over-relaxation parameter is shown in Fig. 14a for the Re = 1000 case and 24k triangle mesh. Increasing the pressure over-relaxation ω_p from the safe default value of 1 to the upper limit of 2 can reduce the number of iterations by about 25%. Most of that gain is achieved by using an over-relaxation value of 1.8 (which is the value used in this paper). Fig. 14b shows the reason a value slightly lower than 2 is suggested. This figure expands the axis to very small error levels (and twice the number of iterations). At these incredibly small scales, pressure under-relaxation close to 2 can become unstable. This instability is not inherent in the M-method itself, it is an artifact of the tolerances on the elliptic pressure solver (in step 2 of the segregated solver algorithm). Tighter tolerances delay these instabilities further. Looser tolerances cause them to happen at earlier iterations and higher error levels.

There is, therefore, a trade-off between the pressure-over relaxation parameter (which reduces the number of iterations) and the work per iteration which increases as the tolerances must get tighter. The issue of instability is easy to spot in the code and to remedy (by decreasing the pressure over-relaxation). It is also only a real issue if extremely small solution errors are desired. However, it is an issue to be aware of. A pressure over-relaxation of 1 is always stable under any circumstances but requires 25% more iterations than is possible if the pressure over-relaxation is invoked.

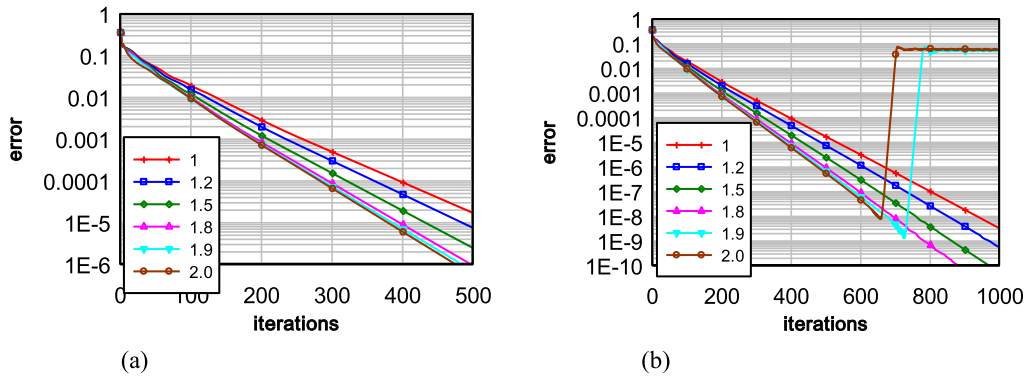

Fig. 14. M-method sensitivity to the pressure over-relaxation parameter ω_p. Cavity flow at Re = 1000 and 24k uniform mesh. (a) Error convergence for different ω_p. (b) Expanded scale for the same figure as in (a) showing the potential instability at very small errors due to solver tolerances.

9. Discussion

Segregated solution algorithms have been the workhorse of industrial computational fluid dynamics for several decades. This work presents a new approach to the mathematical analysis of these methods in the form of a general linear algebra framework. This framework helps our understanding of how all currently known segregated solver methods are similar or different from each other, and how the assumptions they embody affect the overall approximation. It explains various observed convergence phenomena, such as the convergence elbow in the SIMPLE method when the under-relaxation parameter is above the optimum. Furthermore, this framework has allowed the development of two key innovations for solving the saddle point problem. First, we proposed an essentially cost-free method to automatically calculate a good approximation to the optimum under-relaxation value. Second, we developed a low cost, and non-elliptic, correction matrix that can improve the convergence of the constraint variable. These two ideas are not coupled and can be implemented independently.

They can also both be applied to improve any other existing segregated solver approach (with a commutation restriction on the second idea). Implemented as a preconditioner, the M-method can also significantly improve the convergence of fully coupled solution methods.

There were three innovations that produce equation (20) which describes the optimal under-relaxation parameter. First a mathematical expression for the optimal under-relaxation was found in terms of the minimum singular value by analyzing the exact error reduction matrix. Second, a simple and cost effective method for estimating the minimum singular value (via volume weighted Rayleigh quotient estimation) was proposed. And finally, a method for computing and smoothing the estimate using data already computed in the segregated solver methods was proposed leading to a negligible computational cost.

Having the optimum under-relaxation value is attractive because it guarantees convergence in the fewest number of iterations. Just as importantly, however, it also nearly removes the mesh dependence of the convergence rate that hinders existing segregated methods. The M-method only has a small or logarithmic dependence on the number of unknowns. This makes the method particularly attractive for solving large problems with many unknowns, which is very likely the trend of the future. For the fine resolution meshes tested in this work, the M-method is at least an order of magnitude faster than the classic SIMPLE method (or SIMPLEC). The proposed expression for the optimum under-relaxation has just one adjustable parameter, m. The paper showed via experiments that the M-method was remarkably insensitive to the choice of this parameter.

Updating the constraint (pressure in the incompressible Navier-Stokes example) with a matrix that is similar to the commuting part of the problem dynamics matrix, $Q_3 \approx (Q_1 + F^{commute})Q_2^{-1}$, is very inexpensive and was shown to be effective in problems where the commuting term has some influence on the overall physics. For example, the $\mathrm{Re} = 100$ cavity flow problem showed convergence of the errors to the order of 10^{-6} in around 100 iterations for any size mesh. Lower Reynolds numbers (not shown) converge even faster. However, this work was unsuccessful at extending this particular idea to the nonlinear, and non-commutative (with the gradient operator), advection term.

This paper shows via mathematical analysis and experiments that the constraint update can be over-relaxed (via the ω_p parameter) up to a value of 2. This improves the convergence rates on the order of 25% compared to no over-relaxation ($\omega_p = 1$), which is helpful, but not crucial to, the overall method performance.

The net result of this work is concrete proposals that can significantly reduce the computational resources required for flow simulations in many industrial applications. Adoption of an algorithm such as the M-method will have direct benefits of faster and more accurate solutions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The first author was supported in part by the National Science Foundation, grant number 1353942. The work presented in this article is patent pending and the application has published as U.S. Patent Publication No. 17175907.9-1954.

References

[1] R. Temam, Navier–Stokes Equations: Theory and Numerical Analysis, North-Holland, Amsterdam, 1984.
[2] N.N. Yanenko, The Method of Fractional Steps. The Solution of Problems of Mathematical Physics in Several Variables, Springer-Verlag, New York, 1971.
[3] A.J. Chorin, Numerical solution of the Navier-Stokes equations, Math. Comput. 22 (1968) 745.
[4] J. Kim, P. Moin, Application of a fractional-step method to incompressible Navier-Stokes equations, J. Comput. Phys. 59 (2) (1985) 308–323, https://doi.org/10.1016/0021-9991(85)90148-2.
[5] J.B. Perot, P. Moin, Shear-free turbulent boundary layers. Part 2. New concepts for Reynolds stress transport equation modelling of inhomogeneous flows, J. Fluid Mech. 16 (295) (1995) 229–245.
[6] J.B. Perot, Determination of the decay exponent in mechanically stirred isotropic turbulence, 1, http://aip.scitation.org/doi/full/10.1063/1.3582815, 2011.
[7] C.J. Zusi, J.B. Perot, Simulation and modeling of turbulence subjected to a period of uniform plane strain, Phys. Fluids 26 (2014), https://doi.org/10.1063/1.4901188.
[8] S.V. Patankar, D.B. Spalding, A calculation procedure for heat mass and momentum transfer in three dimensional parabolic flows, Int. J. Heat Mass Transf. 15 (1972) 1787.
[9] J.P. van Doormaal, G.D. Raithby, Enhancement of the SIMPLE method for predicting incompressible fluid flow, Numer. Heat Transf. 7 (1984) 147–163.
[10] S.V. Patankar, A calculation procedure for two-dimensional elliptic situations, Numer. Heat Transf. 4 (1981) 409–425.
[11] R. Issa, Solution of implicitly discretized fluid flow equation by operator-splitting, J. Comput. Phys. 62 (1985) 40–65.
[12] J.B. Perot, An analysis of the fractional step method, J. Comput. Phys. 108 (1) (1993) 51–58.
[13] J.B. Perot, Comments on the Fractional Step Method, J. Comput. Phys. 121 (1) (1995) 190–191.
[14] T. Morii, A new efficient algorithm for solving an incompressible flow on relatively fine mesh, Numer. Heat Transf., Part B, Fundam. 47 (6) (2005) 593–610.
[15] H. Elman, V. Howle, J. Shadid, R. Shuttleworth, R. Tuminaro, Block preconditioners based on approximate commutators, SIAM J. Sci. Comput. 27 (5) (2006) 1651–1668, https://doi.org/10.1016/j.jcp.2007.09.026.
[16] C.M. Klaij, C. Vuik, SIMPLE-type preconditioners for cell-centered, colocated finite volume discretization of incompressible Reynolds-averaged Navier–Stokes equations, Int. J. Numer. Methods Fluids 71 (2013) 830–849, https://doi.org/10.1002/fld.3686.

[17] W. Chang, F. Giraldo, J.B. Perot, Analysis of an Exact Fractional Step Method, J. Comput. Phys. 180 (2002) 183–199.
[18] J.M. Hyman, M. Shashkov, The orthogonal decomposition theorems for mimetic finite difference methods, SIAM J. Numer. Anal. 36 (3) (1999) 788–818.
[19] J.B. Perot, Discrete conservation properties of unstructured mesh schemes, Annu. Rev. Fluid Mech. 43 (2011) 299–318.
[20] J.B. Perot, C.J. Zusi, Differential forms for scientists and engineers, J. Comput. Phys., Part B 257 (2014) 1373–1393, https://doi.org/10.1016/j.jcp.2013.08.007.
[21] A. Quarteroni, F. Saleri, A. Veneziani, Factorization methods for the numerical approximation of Navier-Stokes equations, Comput. Methods Appl. Mech. Eng. 188 (2000) 505–526, https://doi.org/10.1016/S0045-7825(99)00192-9.
[22] A. Viguerie, A. Veneziani, Algebraic splitting methods for the steady incompressible Navier-Stokes equations at moderate Reynolds numbers, Comput. Methods Appl. Mech. Eng. 330 (2018) 271–291, https://doi.org/10.1016/j.cma.2017.10.028.
[23] F. Moukalled, M. Darwish, A unified formulation of the segregated class of algorithm for fluid flow at all speeds, Numer. Heat Transf., Part B, Fundam. 37 (2000) 103–139.
[24] X.L. Liu, W.Q. Tao, Y.L. He, A simple method for improving the SIMPLER algorithm for numerical simulations of incompressible fluid flow and heat transfer problems, Eng. Comput. 22 (8) (2005) 921–939, https://doi.org/10.1108/02644400510626488.
[25] Z. Dragojlovic, D. Kaminski, J. Ryoo, Tuning of a fuzzy rule set for controlling convergence of a CFD solver in turbulent flow, Int. J. Heat Mass Transf. 44 (20) (2001) 3811–3822, https://doi.org/10.1016/S0017-9310(01)00029-1.
[26] C.H. Min, W.Q. Tao, An under-relaxation factor control method for accelerating the iteration convergence of flow field simulation, Eng. Comput. 24 (8) (2007) 793–813, https://doi.org/10.1108/02644400710833314.
[27] K. Rao, P. Malan, J.B. Perot, A stopping criterion for the iterative solution of partial differential equations, J. Comput. Phys. 352 (2018) 265–284, https://doi.org/10.1016/j.jcp.2017.09.033.