An Iterative Pressure-Correction Method for the Unsteady Incompressible Navier-Stokes Equation

by

Jean Philippe Aoussou

Submitted to the School of Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Computation for Design and Optimization

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

© Massachusetts Institute of Technology 2016. All rights reserved.

Signature redacted

Author ........................................

School of Engineering
May 11, 2016

Signature redacted

Certified by ..............

Pierre F.J. Lermusiaux
Associate Professor, Department of Mechanical Engineering
Thesis Supervisor

Signature redacted

Accepted by .................

Nicholas Hadjiconstantinou
Co-Director, Computation for Design and Optimization
An Iterative Pressure-Correction Method for the Unsteady Incompressible Navier-Stokes Equation

by

Jean Philippe Aoussou

Submitted to the School of Engineering on May 11, 2016, in partial fulfillment of the requirements for the degree of Master of Science in Computation for Design and Optimization

Abstract

The pressure-correction projection method for the incompressible Navier-Stokes equation is approached as a preconditioned Richardson iterative method for the pressure-Schur complement equation. Typical pressure correction methods perform only one iteration and suffer from a splitting error that results in a spurious numerical boundary layer, and a limited order of convergence in time. We investigate the benefit of performing more than one iteration.

We show that that not only performing more iterations attenuates the effects of the splitting error, but also that it can be more computationally efficient than reducing the time step, for the same level of accuracy. We also devise a stopping criterion that helps achieve a desired order of temporal convergence, and implement our method with multi-stage and multi-step time integration schemes. In order to further reduce the computational cost of our iterative method, we combine it with an Aitken acceleration scheme.

Our theoretical results are validated and illustrated by numerical test cases for the Stokes and Navier-Stokes equations, using Implicit-Explicit Backwards Difference Formula and Runge-Kutta time integration solvers. The test cases comprises a now classical manufactured solution in the projection method literature and a modified version of a more recently proposed manufactured solution.

Thesis Supervisor: Pierre F.J. Lermusiaux
Title: Associate Professor, Department of Mechanical Engineering
Acknowledgments

Those to whom I am eternally indebted are too many to number, and it would prove futile to try and mention them all. However, it is seemly that I should express my most sincere gratitude to Pierre, my advisor, for giving me a chance. I came to know him as one of those men who "are busy by night and by day with meticulous research into such sciences as are profitable to mankind [...] devote themselves to the training of students of capacity", and to whose "discerning taste, the proffered treasures of kings would not compare with a single drop of the waters of knowledge, and mountains of gold and silver could not outweigh the successful solution of a difficult problem". While times were not always easy, what I gained from my time at MIT, more and than a mere degree, is invaluable life experience, and if asked a hundred thousand time, I would not hesitate to make the same choice.
## Contents

1 Introduction .................................................. 11

2 Method outline ............................................... 17

3 Iterative pressure-correction schemes ....................... 23
   3.1 General scheme ........................................... 23
      3.1.1 Scheme .............................................. 23
      3.1.2 Error types .......................................... 24
      3.1.3 Definitions .......................................... 25
   3.2 Convergence properties of implicit-explicit schemes .... 25
      3.2.1 Number of iterations needed to achieve a certain error 28
      3.2.2 Comparison of cost of regular projection vs iterated projection 29
   3.3 Stopping criterion ........................................ 30
   3.4 Convergence acceleration ................................... 32

4 Implementation with common time-integration schemes ....... 35
   4.1 Detailed scheme .......................................... 35
   4.2 Generalization for IMEX schemes .......................... 36

5 Test Cases ....................................................... 39
   5.1 Guermond and Shen test case .............................. 39
      5.1.1 Time step reduction vs. iterations .................. 40
      5.1.2 Increased order of temporal convergence ............ 41
      5.1.3 Aitken acceleration ................................... 42
5.1.4 Effects of the rotational correction ........................................ 43
5.2 Modified Colomés and Badia test case ...................................... 44
  5.2.1 Stopping criterion ............................................................ 45
  5.2.2 Runge-Kutta schemes ......................................................... 46

6 Conclusion and future work ....................................................... 49
List of Figures

2-1 analytic solution, along with the errors obtained using a fully coupled scheme, a regular pressure-correction scheme, and an iterative pressure-correction scheme after 12 iterations. The time-integration method is a BDF2 scheme with $\Delta t = .1$. The space resolution ($N = 200$, 4th order discretization in space) is chosen to be high enough so that the spatial errors are negligible compared to the temporal and splitting errors. This means that the error of the fully coupled scheme is essentially the error of the time integration scheme. As more iterations are performed, the error obtained with the iterative scheme is almost identical to that obtained with the fully coupled scheme. .......... 21

3-1 evolution of the shape of the pressure error as more iterations are performed for the Stokes problem using a BDF3 scheme. After 25 iterations, the final shape is already distinguishable, while after 35 iterations, the splitting error is negligible compared to the space error and the shape of the total error is virtually identical to that obtained with a fully coupled scheme. The space resolution ($N = 400$, 4th order discretization in space) is chosen to be high enough so that the spatial errors are negligible compared to the temporal and splitting errors. . . 24

5-1 pressure error against the total number of iterations, using a BDF3 time integration solver. For each curve, the number of iteration during one time step is prescribed and fixed, while the time step varies. The first point of each curve represents the case where $\Delta t = .0625$. . . . 40
5-2 Stokes problem. The order of convergence of the time-integration scheme is recovered when enough iterations are performed to eliminate the splitting error. ................................. 41

5-3 Navier-Stokes problem. The order of convergence of the time-integration scheme is recovered when enough iterations are performed to eliminate the splitting error. ................................. 42

5-4 convergence of the iterative pressure-correction scheme with and without Aitken acceleration. A BDF2 with time step scheme $\Delta t = .1$ at time $T = .1$ was used ........................................ 42

5-5 convergence of the iterative process with and without rotational correction. A BDF2 time scheme was used and two time-step values are compared. Both results are for time $T = .1$ ................................. 43

5-6 spatial error shapes obtained with discretizations of different orders . 44

5-7 temporal convergence of the pressure with a BDF3 for the Stokes problem 45

5-8 the number of iterations required to achieve a certain order of temporal convergence using the stopping criterion for different time-steps ... 46

5-9 Stokes problem with 3rd order stopping criterion .................. 47

5-10 Navier-Stokes problem with 3rd order stopping criterion .......... 48

10
Chapter 1

Introduction

Ocean models are of vital importance not only for weather forecast but also for assessing the impact of human activity on the environment. While at their inception models involved mostly finite difference methods of low order, a number of more efficient and accurate schemes have been developed with each passing year. One of the first ocean models was introduced by Bryan (1969) and used a hydrostatic, rigid lid model and an energy conserving numerical scheme. A review of first generation ocean models was produced by Griffies et al. (2000). Many ocean models have since then been devised, with increasing level of sophistication, such as complex data assimilation schemes, adaptive modelling capabilities, free surface and open boundary conditions. In particular, more recent models have departed from low-order finite difference and finite volume schemes on structured grids and promising results were obtained with finite volume and finite elements methods on unstructured grids and curvilinear structured grids. Improvements have also been made on the vertical discretization, such as terrain following coordinate schemes, isopycnal vertical coordinates, z-coordinates, and hybrid schemes (see Uckermann (2009) for more details). Pain et al. (2005) and Slingo et al. (2009) offer a more detailed review of diverse ocean modelling endeavors.

The results presented here are part of an overarching research project of the MIT Multidisciplinary Simulation, Estimation, and Assimilation Systems (MSEAS) group, the aim of which is to design more accurate and efficient ocean models. The MSEAS software (Leslie and Xu, 2010) is used for fundamental research as well as simula-
tions, monitoring, real-time ecosystem and acoustic predictions and environmental management (see for example Beşiktepe et al., 2003; Colin et al., 2013; Cossarini et al., 2009; Gangopadhyay et al., 2011; Haley et al., 2009; Lermusiaux et al., 2007; Leslie et al., 2008; Onken et al., 2008; Ramp et al., 2011; Xu et al., 2008; Ueckermann and Lermusiaux, 2010).

The MSEAS software currently comprises three solvers of governing fluid and ocean dynamics equations. One of those solvers is used to solve primitive-equation dynamics (Cushman-Roisin and Beckers, 2011) with a nonlinear free surface, and allows to analyze tidal to mesoscale processes over complex geometries (Haley Jr and Lermusiaux, 2010). Despite the numerous merits of this solver, it uses a second-order structured finite volumes scheme and is designed for hydrostatic flows. To overcome this obstacle, taking advantage of recent advances in Finite Element techniques, an hybridizable discontinuous Galerkin (HDG) was designed, aiming for optimal trade-off between accuracy and computational cost. HDG parameterizes element-local solutions using a new global edge space, ensuring conservation of normal fluxes. This reduces the number of globally coupled DOF and renders HDG competitive with CG for elliptic problems (Waluga and Egger, 2012; Kirby et al., 2012). However, the HDG framework is complex, and difficultly amenable to developments other than that of the HDG method itself. A third solver is a more versatile MATLAB-based finite volume framework for two-dimensional (2D) Navier-Stokes flows (Uckermann and Lermusiaux, 2009) which uses a uniform staggered C-grid for its spatial discretization. This framework is particularly amenable to the incubation of new projects. Since the present work focuses on the improvement of the temporal and projection schemes used to solve unsteady fluid dynamics problem, independently of the space discretization used, this third solver is particularly convenient and is the one used to obtain all the numerical results presented.

Solving the fully coupled incompressible Navier-Stokes equations is computationally demanding (Maday et al., 1993; Turek, 1997; Guermond and Quartapelle, 1998; Ueckermann and Lermusiaux, 2016). The projection method introduced by Chorin (1968) and Temam (1968), which decouples the momentum and continuity equations
by replacing the latter with a Poisson equation for the pressure is often a preferred option. A divergent velocity field is first obtained by omitting the pressure term in the momentum equation, and is then corrected using the pressure Poisson equation. This method was reinterpreted as a fractional-step scheme by Kim and Moin (1985) and a number of variations on the original scheme now exists, a review of which can be found in Guermond et al. (2006). In this study, only incremental pressure-correction methods are considered. This time splitting allows to solve matrices much smaller than the fully coupled system and is roughly \((d + 1)^2\) times more efficient, with \(d\) the spatial dimension of the problem (Ueckermann and Lermusiaux, 2016). However, this gain comes at the price of a splitting error which manifests itself as an artificial boundary layer in space and as a limited order of convergence in time (Liu, 1995; Shen, 1996; Guermond and Shen, 2004; Guermond et al., 2006).

Incremental pressure-correction methods use the pressure at the previous time-step as a guess for the current time. This guess is then updated, and the scheme moves on to the next time step. In order to obtain a more accurate solution, instead of directly moving on to the next time step, it is possible to repeat the process and use the newly computed pressure as the guess. Indeed, doing so reduces the splitting error, and if the process is repeated enough times, renders it negligible compared to the error intrinsic to the time integration scheme. In fact, other methods, such as the Uzawa algorithm (Maday et al., 1993), the SIMPLE method and their variants (Ferziger and Peric, 2012), which like projection schemes decouple the system and replace the incompressibility condition with a pressure Poisson equation, perform similar inner loop iterations in addition to the time integration outer loop. They are particularly efficient for stationary problems (Ferziger and Peric, 2012). However, since the splitting error scales with the time step, it is not clear whether performing those pressure correction iterations offers a real benefit as opposed to using smaller time steps in order to achieve the same accuracy. Our objective is to investigate the convergence properties of such an iterative pressure-correction scheme and assess the benefit of performing more iterations as opposed to reducing the time step.

The pressure-correction method can also be viewed as a relaxation method for
the pressure-Schur complement equation. This is in fact also true for the Uzawa and SIMPLE schemes (Maday et al., 1993; Mardal and Langtangen, 2003; Olshanskii and Vassilevski, 2007). More precisely, those methods can be understood to be preconditioned Richardson iterations and only differ with respect to the preconditioner used. In the case of the projection method, the preconditioner is the Laplacian operator and with a regular scheme, only one iteration is performed (Turek, 1999). If more iterations are performed, the numerical solution converges to the values that would be obtained if the pressure-Schur equation were directly solved, without relaxation. With this approach, it is possible to estimate how fast the splitting error decreases with each iteration and thus assess how performing more iterations compares to reducing the time step in terms of computational cost and accuracy. When doing so, one must take care in differentiating between the different types of error, namely the spatial error, the splitting error and the temporal error intrinsic to the time integration scheme used.

A wide range of time-integration schemes is available to solve the incompressible Navier-Stokes equations, for both fully coupled and projection-type methods. A common choice are Backward Difference Formulae (BDF) schemes (Canuto et al., 2007). Those fully implicit schemes usually enjoy good stability for relatively large time-steps. However, they require the use of an iterative solver because of the non-linearity of the advective terms (Turek, 1999; Nguyen and Peraire, 2012). Implicit-explicit (IMEX) schemes thus offer a good alternative and have indeed been commonly used with projection-like schemes (Kim and Moin, 1985; Löhner, 2004; Canuto et al., 2007).

Nevertheless, BDF schemes are not self-starting and have smaller stability regions at high orders (Shampine, 1994). In particular, methods of order higher than 3 are not A-stable, and methods of order higher than 6 are unstable (Hundsdorfer and Verwer, 2013). Although this is not necessarily an issue for all types of problems (Cubillos-Moraga, 2015) and other multi-step methods with better stability properties do exist (Hundsdorfer and Ruuth, 2007), multistage methods, such as Runge-Kutta schemes seem to offer a viable alternative. The study of their application to projection methods
has however been limited so far. A number of studies address the use of fully explicit schemes (Ren and Utnes, 1993; Kang et al., 2000; Löhner, 2004; Detrembleur et al., 2008). Second order RK-IMEX schemes were studied by Ni et al. (2003) and Ni and Abdou (2007), and higher order schemes by Zhou (2014) and Colomés and Badia (2016) for pressure-Poisson methods. A pressure-free scheme projection methods with periodic boundary conditions was proposed by Zhang (2014). To our knowledge, Ueckermann and Lermusiaux (2016) outlines the most general IMEX-RK schemes for pressure-correction methods.

We start by summarizing the most commonly used methods for solving the incompressible Navier-Stokes equation (chapter 2). We then analyze the convergence properties of our iterative projection method (chapter 3). In particular, we show that in order to achieve a certain accuracy, there is often an advantage in terms of computational effort in performing more iterations as opposed to using smaller time steps. We give an example of a dynamically adaptive stopping criterion that allows some control of the temporal convergence of a scheme and show how the method can be accelerated using an Aitken relaxation scheme (Macleod, 1986). Next, we illustrate how the method can practically be embedded in an existing pressure-correction scheme for both multi-step and multistage time integration schemes (chapter 4). We then present numerical results that validate the properties inferred in the earlier parts by applying the method to test cases for both the Stokes and Navier-Stokes equations (chapter 5). Finally, conclusions are drawn (chapter 6).
Chapter 2  

Method outline  

The incompressible Navier-Stokes equations can be written as:

\[
\begin{align*} 
\frac{\partial \mathbf{u}}{\partial t} &= \nu \nabla^2 \mathbf{u} - \mathbf{u} \cdot \nabla \mathbf{u} - \nabla p + f \\
\nabla \cdot \mathbf{u} &= 0 
\end{align*}
\]  

(2.1a)  

(2.1b)

where \( p \) is pressure divided by the density of water \( \rho_0 \), \( \nu \) the kinematic viscosity and \( f \) a forcing term. Over the years, a variety of techniques have been devised that strive to offer a reasonable trade-off between accuracy and computational cost. In this chapter, we present three common numerical methods for solving the Navier-Stokes, namely fully coupled schemes, pressure-Schur complement methods and pressure-correction projection methods, and we remind the reader that the projection method can essentially be interpreted as a pressure-Schur complement method.

**Fully coupled schemes**  
The fully discretized system can be generically represented as:

\[
\begin{bmatrix} 
\mathbf{A} & \mathbf{G} \\
\mathbf{D} & 0 
\end{bmatrix} 
\begin{bmatrix} 
\mathbf{u}^{n+1} \\
\mathbf{p}^{n+1} 
\end{bmatrix} = 
\begin{bmatrix} 
\mathbf{f}^{n+1} \\
0 
\end{bmatrix} + 
\begin{bmatrix} 
b c 
\end{bmatrix}
\]  

(2.2)

where \( \mathbf{G} \) and \( \mathbf{D} \) are respectively the gradient and diffusion operators, \( bc \) accounts for the boundary conditions prescribed for the velocity and \( \mathbf{A} \) is the implicit operator.
for the advection-diffusion part of the momentum equation (Chang et al., 2002). This system can be solved using a linear iterative solver, such as the conjugate gradient or the GMRES methods. Doing so is however usually computationally expensive.

**Pressure-Schur complement methods** Using the momentum equation (2.1a), the velocity can be expressed in terms of the pressure:

\[ u^{n+1} = A^{-1}(f^{n+1} + bc - Gp^{n+1}) \]  

(2.3)

The resulting expression can be plugged in the incompressibility equation (2.1b) to obtain an equation only in terms of the pressure:

\[ DA^{-1}Gp^{n+1} = DA^{-1}f^{n+1} \]  

(2.4)

It is thus possible to first solve for the pressure using (2.4) and recover the velocity with (2.3). This technique is called the Schur-complement (Zhang, 2006) and is commonly used for physical problems involving the coupling of a constitutive equation with a conservation equation (see for example nodal analysis in electrical engineering (Chua and Lin, 1975)).

In our case, \( DA^{-1}G \) is referred to as the pressure-Schur complement and (2.4) as the pressure-Schur complement equation (Turek, 1999). While the pressure-Schur complement is invertible (Mardal and Langtangen, 2003), it is dense, so typically a relaxation scheme needs to be used. Stationary iterative methods such as the Jacobi, Gauss-Seidel or the Richardson iterative schemes are a common choice. For better convergence, the Richardson iteration can be used with a preconditioner (Watkins, 2004). As an example, applied to the pressure-Schur complement equation with preconditioner \( P \), it can be written as:

\[ p^{n+1}_{k+1} = p^{n+1}_{k} - P^{-1}[DA^{-1}Gp^{n+1}_{k} - b^{n+1}] \]  

(2.5)

**Pressure-correction projection methods** One of the most commonly used techniques is the projection method of Chorin (1968) and Temam (1968). Details about
the scheme and its variations can be found in Guermond et al. (2006). Here, we only consider the incremental pressure-correction method. The scheme is well-known, and it is outlined here only to clarify concepts and notations that will be used in other parts of this document. Let us assume a backward Euler scheme is used for time integration:

\[
\begin{align*}
\frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} &= \nu \nabla^2 \tilde{\mathbf{u}}^{n+1} - \nabla \nabla \mathbf{u}^{n+1} - \nabla p_n^{n+1} + \mathbf{f}^{n+1} \\
\frac{\mathbf{u}^{n+1} - \tilde{\mathbf{u}}^{n+1}}{\Delta t} &= -\nabla \delta p^{n+1} \\
\nabla^2 \delta p^{n+1} &= \frac{\nabla \cdot \tilde{\mathbf{u}}^{n+1}}{\Delta t} \\
p^{n+1} &= p_*^{n+1} + \delta p^{n+1} \\
\mathbf{u}^{n+1} &= \tilde{\mathbf{u}}^{n+1} - \Delta t \nabla \delta p^{n+1}
\end{align*}
\] (2.6)

\(p_*^{n+1}\) is a guess for \(p^{n+1}\), and \(\tilde{\mathbf{u}}^{n+1}\) is the predicted velocity. After \(\tilde{\mathbf{u}}^{n+1}\) is obtained from (2.6a), \(\delta p^{n+1} = p^{n+1} - p_*^{n+1}\), the difference between the actual pressure and its guess, can be obtained from (2.6b). However, the corrected velocity \(\mathbf{u}^{n+1}\) is unknown. To resolve this issue, we can appeal to the incompressibility condition that applies to the corrected velocity but not to the predicted velocity. Taking the divergence of (2.6b), one obtains the pressure Poisson equation (2.6c) that allows to solve for \(\delta p^{n+1}\). Once \(\delta p^{n+1}\) is known, the pressure can be obtained by performing the pressure-correction (2.6d) that lends its name to the method. We can then obtained the corrected velocity by performing the velocity correction (2.6e).

So far, boundary conditions have not been mentioned, and indeed they are a recurrent issue with projection methods. There is no accepted correct choice for either the velocity or the pressure — any imposed boundary condition is arbitrary (Liu, 1995) and results in an inconsistency in the boundary value of one or both variables. In fact, similar problems exist for other methods where a Laplacian operator is applied to the momentum equation (2.1a) (Shirokoff and Rosales, 2011).

Inconsistencies in the boundary conditions manifest themselves in a splitting error that limits the overall order of time convergence of the scheme, regardless of the order
of the time integration scheme used. To somewhat remedy this problem, it possible to add a term to the pressure correction, introduced by Timmermans et al. (1996) and popularized by Guermond and Shen (2004) under the name "rotational correction". The pressure-correction step now has an additional term:

\[ p^{n+1} = p^{n+1}_* + \delta p^{n+1} - \nu \nabla \cdot \mathbf{u}^{n+1} \quad (2.7) \]

To understand the origin of this term, one can consider the Stokes case and the unsplit-momentum equation that \( \mathbf{u}^{n+1} \) and \( p^{n+1} \) must satisfy:

\[ \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = \nu \nabla^2 \mathbf{u}^{n+1} - \nabla p_{n+1} + \mathbf{f}^{n+1} \quad (2.8) \]

By plugging (2.6e) and (2.6d) into (2.8), one obtains

\[ \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} = \nu \nabla^2 \mathbf{u}^{n+1} - \Delta t \nabla^2 \nabla \delta p^{n+1} - \nabla p_{n+1} + \mathbf{f}^{n+1} \quad (2.9) \]

which can be rearranged as

\[ \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} - \left( \nu \nabla^2 \mathbf{u}^{n+1} - \nabla p_{n+1} + \mathbf{f}^{n+1} \right) = -\Delta t \nabla^2 \nabla \delta p^{n+1} \quad (2.10) \]

From (2.6a) it is obvious that the left hand side is 0, which reveals an inconsistency error that scales with the time-step. One can check that if (2.7) is used instead of (2.6d), this error vanishes. A similar explanation is provided in Minev and Gresho (1998).

It can also be shown that when the rotational correction is applied, the splitting error only results in an inconsistent tangential boundary condition on \( \mathbf{u}^{n+1} \) and that \( \mathbf{u}^{n+1} \) is of order \( O(\Delta t^2) \) and \( p^{n+1} \) of order \( O(\Delta t^{3/2}) \) (Guermond and Shen, 2004). While in some special cases it is possible to achieve higher orders of temporal convergence (Zhang, 2014), those are the best convergence rates for a projection scheme in the most general cases.
The projection method as a pressure-Schur complement method  The above description of the projection method features diverse intermediate variables. It is important to realize that those variables have no meaning of themselves. Indeed, by performing the relevant substitutions, it is possible to condense the scheme into a set of two equations (see appendix for details):

Figure 2-1: analytic solution, along with the errors obtained using a fully coupled scheme, a regular pressure-correction scheme, and an iterative pressure-correction scheme after 12 iterations. The time-integration method is a BDF2 scheme with \( \Delta t = .1 \). The space resolution \( (N = 200, 4^{th} \text{ order discretization in space}) \) is chosen to be high enough so that the spatial errors are negligible compared to the temporal and splitting errors. This means that the error of the fully coupled scheme is essentially the error of the time integration scheme. As more iterations are performed, the error obtained with the iterative scheme is almost identical to that obtained with the fully coupled scheme.
\[
\begin{align*}
    p^{n+1} &= p^n - L^{-1}(DA^{-1}Gp^{n+1} - b^{n+1}) \\
    u^{n+1} &= A^{-1}(f^{n+1} + bc - Gp^{n+1})
\end{align*}
\] (2.11)

One can thus see that (2.11) is simply a special case of (2.5) where \(P^{-1} = L^{-1}\), \(k = 0\) and \(p_0^{n+1} = p^n\). In words, this means the following: the incremental pressure-correction method is a preconditioned Richardson iterative method for the pressure-Schur complement equation. The preconditioner is the Laplacian operator and the first guess is the value at the previous time step. Only one iteration is performed, and a bound is known for the error of the solution obtained after that one iteration.

A detailed explanation of this concept can be found in Turek (1999). Moreover, it can be shown that if the rotational correction is applied, the preconditioner is \(P^{-1} = L^{-1} - \nu \Delta t I\) (see appendix). In both cases, \(-P\) is positive semi-definite. Interestingly, according to Turek (1999), the use of this preconditioner was first introduced by Glowinski (1991), before the rotational correction was used by Timmermans et al. (1996).

In the next section, we assess the advantage of performing more than one iteration of the projection method within a time step as opposed to reducing the time step \(\Delta t\) in terms of computational in order to achieve the same accuracy.
Chapter 3

Iterative pressure-correction schemes

In this section, we show how iteration schemes can efficiently reduce the splitting error and improve the global temporal order of convergence. We compare the computational costs to the cost of decreasing the time-step. We also obtain and discuss varied computational properties of our reiteration schemes including their overall efficiency, stopping criteria, and acceleration schemes.

3.1 General scheme

3.1.1 Scheme

The iterative scheme we analyze in this document can be written as:

\[
\begin{align*}
    p_{k+1}^{n+1} &= p_k^{n+1} - P^{-1}(D A^{-1} G p_k^{n+1} - b^{n+1}) \\
    u^{n+1} &= A^{-1}(f^{n+1} + bc - G p^{n+1}) \\
    p_0^{n+1} &= p^n_{k_n}
\end{align*}
\]  

(3.1)

where \( P^{-1} = L^{-1} \) for a regular pressure-correction scheme and \( P^{-1} = L^{-1} - \frac{\Delta t}{\alpha} I \) for a scheme with rotational correction, with \( \alpha \) a parameter that depends on the time integration scheme used. Here, the number of iterations \( k_n \) is prescribed for simplicity. Of course, a tolerance on the error can also be used. The choice of the stopping criterion will be discussed later. It is known that for this scheme, the
numerical solution of the velocity and the pressure after one iteration are respectively of order \( O(\Delta t^2) \) and \( O(\Delta^3) \) (Guermond and Shen, 2004).

Figure 3-1: evolution of the shape of the pressure error as more iterations are performed for the Stokes problem using a BDF3 scheme. After 25 iterations, the final shape is already distinguishable, while after 35 iterations, the splitting error is negligible compared to the space error and the shape of the total error is virtually identical to that obtained with a fully coupled scheme. The space resolution (\( N = 400 \), 4th order discretization in space) is chosen to be high enough so that the spatial errors are negligible compared to the temporal and splitting errors.

### 3.1.2 Error types

When dealing with projection methods, one is confronted to three types of error:

- the spatial error, which scales with \( h^p \), with \( h \) a parameter representative of the space resolution and \( p \) the order of the space discretization
• the temporal error which scales with $\Delta t^q$, with $q$ the order of the time integration scheme

• the splitting error which scales with $\Delta t^s$

The global truncation error of the scheme is composed of the sum of those three errors:

$$e_{global} = e_{space} + e_{splitting} + e_{time} = O(h^p) + O(\Delta t^q) + O(\Delta t^s)$$  \hspace{1cm} (3.2)

Within each time iteration, the value of the pressure becomes closer to the values that would be obtained if the pressure-Schur complement equation were solved directly, without Richardson relaxation. This means that in addition to decreasing with smaller time steps, the splitting error decreases after each projection iteration. In this section, we assume that the spatial and time-scheme error are negligible compared to the splitting error.

### 3.1.3 Definitions

For clarity of exposition, we define the following terms:

• **iteration**: one pressure-correction iteration from $k$ to $k + 1$ of scheme (3.1)

• **global temporal convergence**: convergence rate of the error of the scheme as the time-step varies

• **time scheme convergence**: intrinsic rate of convergence of the time integration scheme used

### 3.2 Convergence properties of implicit-explicit schemes

**Convergence within one time-step**

The iteration matrix of scheme (3.1) is $M = I - P^{-1}(DA^{-1}G)$. For schemes in which advection is treated explicitly, $A = I - \frac{\nu \Delta t}{\alpha} L$. The Richardson iterative methods is a stationary iterative method, like Jacobi and Gauss-Seidel iterations. For such
methods, the rate of convergence $R$ is bounded by $\rho(M)$, the radius of convergence of the iteration matrix (Watkins, 2004).

We can thus immediately see that since the values corresponding to the boundary conditions, the variables at the previous time level and forcing terms are all contained in vector $b$, they do not affect $\rho(M)$ and therefore have no effect on the convergence of the algorithm. Only the choice of space discretization and time-step do.

For common space discretization methods such as finite volumes, finite elements and finite differences, the negative discrete Laplacian operator is symmetric positive definite (Meyer, 2000; Elman et al., 2014). Therefore, all the eigenvalues of $-L$ are real and positive. If $\lambda_{\min}$ is the smallest eigenvalue of $-L$, the largest eigenvalue of $(I - \frac{\nu \Delta t}{\alpha} L)^{-1}$ is

$$\lambda_{\min}(A^{-1}) = \frac{1}{1 + \frac{\nu \Delta t}{\alpha} \lambda_{\max}(-L)}$$

(3.3)

The discrete gradient and divergence operators verify $D = -G^T$ (Turek, 1999; Chang et al., 2002). Since $A$ is a sum of SPD matrices, so is the pressure-Schur complement

$$S = -DA^{-1}G = G^TA^{-1}G$$

(3.4)

Let us call $x_0$ the eigenvector that corresponds to $\lambda_{\min}(S)$.

$$\|x_0^T G^T A^{-1} G x_0\| = \|x_0^T \lambda_{\min}(S) x_0\| = \lambda_{\min}(S) \|x_0\|^2$$

(3.5)

Now using the Rayleigh quotient, for any vector $y = Gx$,

$$\lambda_{\min}(A^{-1}) < \frac{\|y^T A^{-1} y\|}{\|y^T y\|}$$

(3.6)

so in particular,

$$\lambda_{\min}(A^{-1}) \|x_0^T G^T G x_0\| < \|x_0^T G^T A^{-1} G x_0\|$$

(3.7)
Similarly, using the Rayleigh quotient, we can show that, since $G^T G$ is symmetric positive semi-definite:

$$0 < \lambda_{\min}(G^T G) \| \tilde{x}_0 \|^2 < \| \tilde{x}_0^T G^T G \tilde{x}_0 \|$$

(3.8)

and combining with (3.7), that

$$\lambda_{\min}(A^{-1}) \lambda_{\min}(G^T G) \| \tilde{x}_0 \|^2 < \| \tilde{x}_0^T G^T A^{-1} G \tilde{x}_0 \|$$

(3.9)

Finally, combining (3.3), (3.5) and (3.9) we obtain a lower bound for the eigenvalues of $S$:

$$0 < \frac{\lambda_{\min}(G^T G)}{1 + \frac{\nu \Delta t}{\alpha} \lambda_{\max}(-L)} < \lambda_{\min}(S)$$

(3.10)

We can write $M$ as $M = I - (-P^{-1})S$. Since $-P$ and $S$ are both symmetric positive definite, their eigenvalues are positive, and we have

$$0 < \lambda_{\max}((-P^{-1}S)^{-1}) < \lambda_{\max}(-L) \lambda_{\max}(S^{-1})$$

(3.11)

and thus

$$0 < \lambda_{\min}(-L^{-1}) \lambda_{\min}(S) < \lambda_{\min}(-L^{-1}S)$$

(3.12)

The convergence rate of the scheme is bounded by $\rho(M)$, the spectral radius of $M$:

$$R \leq \rho(M) \leq \left| 1 - \lambda_{\min}(-P^{-1}) \frac{\lambda_{\min}(G^T G)}{1 + \frac{\nu \Delta t}{\alpha} \lambda_{\max}(-L)} \right|$$

(3.13)

For the case without rotational correction we have

$$R \leq \left| 1 - \left( \frac{1}{\lambda_{\max}(-L)} \right) \frac{\lambda_{\min}(G^T G)}{1 + \frac{\nu \Delta t}{\alpha} \lambda_{\max}(-L)} \right|$$

(3.14)

For the scheme with rotational correction, the bound is
Using the Rayleigh quotient, we find that for any \( \hat{x} \)

\[
0 < \frac{\lambda_{\min}(G^T G)}{\lambda_{\max}(-L)} \leq \frac{\|\hat{x}^T G^T G \hat{x}\|}{\|\hat{x}^T (-L) \hat{x}\|}
\]

It was shown in Cai (2015) that \( \frac{\|\hat{x}^T G^T G \hat{x}\|}{\|\hat{x}^T (-L) \hat{x}\|} \leq 1 \). Combining this result with (3.16), we can see that \( R < 1 \). We conclude that with the rotational correction, the scheme convergences and that the convergence rate is bounded by a constant that is independent of the time-step. When the rotational correction is not applied, the convergence rate is bounded by a smaller constant with smaller time steps and asymptotically, it is bounded by the same constant as with the rotational correction case. Intuitively, this result is sensible, since as shown in the previous section, the rotational correction accounts for an error that scales with the time-step, and asymptotically, the two schemes are the same.

### 3.2.1 Number of iterations needed to achieve a certain error

If only one iteration is performed, as for the regular pressure-correction method, the splitting error \( \|e_0\| \) on the pressure is such that:

\[
\|e_0\| < C_0 \Delta t^{s_0}
\]

Let us assume that at each time-step, we want to perform enough iterations so that the temporal order of the splitting error, \( \|e_1\| \), is \( s_1 > s_0 \) and

\[
\|e_1\| < C_1 \Delta t^{s_1}
\]
Let $r$ be the number of iterations required to achieve this. We then have

$$\|e_1\| = \|e_0\| R^r < C_1 \Delta t^{s_1} \quad (3.20)$$

and thus

$$\|e_0\| < \frac{C_1 \Delta t^{s_1}}{R^r} \quad (3.21)$$

Using (3.18) we obtain the equality

$$R^r = \frac{C_1}{C_0} \Delta t^{s_1 - s_0} \quad (3.22)$$

so the number of iterations required is

$$r = (s_1 - s_0) \frac{\ln(C_1 \Delta t)}{\ln R} \quad (3.23)$$

Since as $\Delta t$ tends to 0, $R$ is bounded by a constant, $r$ tends to infinity. This means that as the time step is smaller, we need to perform more iterations to achieve the same order of temporal convergence of the splitting error. However, we will prove in the next section that this is not disadvantageous.

### 3.2.2 Comparison of cost of regular projection vs iterated projection

Let the splitting error for a scheme with no reiterations with temporal convergence $s_0$ be:

$$e_0 = C_0 \Delta t^{s_0} + O(\Delta t^{s_0}) \quad (3.24)$$

and the error for a scheme for which enough iterations are performed so that the scheme is of order $s_1 > s_0$ be:

$$e_1 = C_1 \Delta t^{s_1} + O(\Delta t^{s_1}) \quad (3.25)$$
Let us assume that if the time-step used for the first scheme is \( \Delta t_0 \), the time step we need to use with the second scheme for \( e_0 \) and \( e_1 \) to be equal is \( \Delta t_1 \). Neglecting higher order terms, we then have:

\[
\Delta t_0 = \left( \frac{C_1}{C_0} \right)^{\frac{1}{\alpha_0}} \Delta t_1^{\frac{\alpha_1}{\alpha_0}} \quad (3.26)
\]

The costs in terms of number of projections for the two methods are respectively:

\[
c_0 = \frac{T}{\Delta t_0} = \frac{T}{\left( \frac{C_1}{C_0} \right)^{\frac{1}{\alpha_0}} \Delta t_1^{\frac{\alpha_1}{\alpha_0}}} \quad (3.27)
\]

\[
c_1 = (s_1 - s_0) \frac{\ln\left( \frac{C_1}{C_0} \Delta t_1 \right)}{\ln R} \frac{T}{\Delta t_1} \quad (3.28)
\]

The ratio of the cost is:

\[
\frac{c_1}{c_0} = (s_1 - s_0) \frac{\ln\left( \frac{C_1}{C_0} \Delta t_1 \right)}{\ln R} \frac{T}{\Delta t_1} \frac{\left( \frac{C_1}{C_0} \right)^{\frac{1}{\alpha_0}} \Delta t_1^{\frac{\alpha_1}{\alpha_0}}}{T} = C \frac{\ln\left( \frac{C_1}{C_0} \Delta t_1 \right)}{\Delta t_1^{\frac{1-\alpha_1}{\alpha_0}}} \quad (3.29)
\]

The limit of this ratio as \( \Delta t \) tends to 0 is 0, which shows that asymptotically, the iterative method is computationally cheaper than reducing the time step.

### 3.3 Stopping criterion

As with any iterative method, one difficulty is to decide when to stop iterating, and indeed choosing a stopping criterion is an art in itself. Stopping criteria often involve the value of the residual and, for the best ones, knowledge of the iteration matrix (Barrett et al., 1994), neither of which are readily available in our method. In any case, it is necessary to choose a tolerance for the error, and this choice, if possible, should not be arbitrary.

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

Let us assume that for a certain projection scheme, the splitting error is of order $s_0$ and let $p_{ex}^{n+1}$ be the exact solution:

$$\|p_1^{n+1} - p_{ex}^{n+1}\| < C_0 \Delta t^{s_0} \tag{3.30}$$

Since $p_{ex}^{n+1}$ is unknown, this bound is of little practical value. However, we do know the value of the second pressure correction $\|p_1^{n+1} - p_2^{n+1}\|$ for which we can derive a bound using (3.30):

$$\|p_1^{n+1} - p_2^{n+1}\| = \|p_1^{n+1} - p_{ex}^{n+1} + p_{ex}^{n+1} - p_2^{n+1}\| < \|p_1^{n+1} - p_{ex}^{n+1}\| + \|p_2^{n+1} - p_{ex}^{n+1}\| \tag{3.31}$$

Since $p_2^{n+1}$ is a better approximation of $p_{ex}^{n+1}$ than $p_1^{n+1}$, we also have:

$$\|p_2^{n+1} - p_{ex}^{n+1}\| < C_0 \Delta t^{s_0} \tag{3.32}$$

so that:

$$\|p_1^{n+1} - p_2^{n+1}\| < 2C_0 \Delta t^{s_0} \tag{3.33}$$

We now want to use this result to devise a stopping criterion. Let us assume that we want the scheme to converge at order $s_1$, so we want to stop at iteration $k$ where:

$$\|p_k^{n+1} - p_{ex}^{n+1}\| < C_1 \Delta t^{s_1} \tag{3.34}$$

We will now show that if we choose the following stopping criterion:

$$\|p_{k-1}^{n+1} - p_k^{n+1}\| < \|p_1^{n+1} - p_2^{n+1}\| \Delta t^{s_1-s_0} \tag{3.35}$$

then (3.34) is satisfied.

If $R < 1$ is the convergence rate of the scheme, at each iteration, the norm of the splitting error at each iteration is multiplied by $R$:
\[ \|p_{k+1}^{n+1} - p_{ex}^{n+1}\| = R\|p_k^{n+1} - p_{ex}^{n+1}\| \] (3.36)

Thus, the error at iteration \( k \) can be bounded by \( \|p_{k-1}^{n+1} - p_k^{n+1}\| \) the pressure correction at iteration \( k \):

\[
\|p_{k-1}^{n+1} - p_{ex}^{n+1}\| < \|p_{k-1}^{n+1} - p_k^{n+1}\| + \|p_k^{n+1} - p_{ex}^{n+1}\| \\
= \|p_{k-1}^{n+1} - p_k^{n+1}\| + R\|p_k^{n+1} - p_{ex}^{n+1}\| 
n\] (3.37)

so

\[
\|p_{k-1}^{n+1} - p_{ex}^{n+1}\| = R\|p_k^{n+1} - p_{ex}^{n+1}\| < \frac{R}{1-R}\|p_k^{n+1} - p_{ex}^{n+1}\| 
n\] (3.38)

If we now choose (3.35) as our stopping criterion, using (3.38), we see that:

\[
\|p_{k-1}^{n+1} - p_{ex}^{n+1}\| < \frac{R}{1-R}\|p_k^{n+1} - p_{ex}^{n+1}\| \Delta t^{s_1-s_0} 
n\] (3.39)

Now, using (3.39) and (3.33), we have:

\[
\|p_k^{n+1} - p_{ex}^{n+1}\| < \frac{C_0 R}{1-R}\Delta t^{s_0} \Delta t^{s_1-s_0} = C_1 \Delta t^{s_1} 
n\] (3.40)

The result is valid in any norm. In this study, we implement the criterion with the \( L^\infty \)-norm.

### 3.4 Convergence acceleration

Stationary iterative methods can converge slowly and one might desire to accelerate the convergence of the algorithm. It is of course possible to use better preconditioners for the pressure-Schur complement, or event to use other types of solvers. However, such techniques are beyond the scope of the present study. Instead, we prefer to devise a method that can be simply embedded in an existing code with minimal changes.

It is known that Richardson iterative methods converges linearly to the correct
solution for any arbitrary guess. A good candidate to accelerate the convergence of linear methods is the Aitken relaxation method. There exists many variants of the scheme, which differ in the computation themselves and in the choice of initial parameters, and we only provide an example of one possible implementation. In particular, while there is only one version of the method for the scalar case, multiple methods exist for the vector case (Macleod, 1986). We used the method devised by Anderson (1965) which was also used by Baek and Karniadakis (2011) for an iterative velocity correction scheme. If a relaxation algorithm can be summarized in the following form:

$$x_{k+1} = F(x_k)$$ (3.41)

where $x_k$ is a vector, then the Aitken relaxation scheme is:

$$x_{k+1} = F(x_k) + \omega_k (x_k - F(x_k))$$ (3.42)

where $\omega_k$ a dynamic relaxation parameter, computed as follows:

$$\omega_k = \omega_{k-1} + (\omega_{k-1} - 1) \frac{d_k \cdot (d_k - d_{k-1})}{\|d_k - d_{k-1}\|^2}$$ (3.43)

and $d_k$ is defined by $d_k = x_k - F(x_k)$.

For our implementation, we use the method devised by Balda (2006) which is a modified version of that of Anderson (1965) that includes a stabilization parameter. This method was also used for a velocity-correction reiteration scheme in Baek and Karniadakis (2011). $\omega_0$ can be chosen to be a large number, such as done by Balda (2006) or the value at the previous time-step, such as done by Baek and Karniadakis (2011). The stopping criterion of the unaccelerated method can be used as:

$$\|d_{k-1}^{n+1}\| < \|d_0^{n+1}\| \Delta t^{s_1-s_0}$$ (3.44)
Chapter 4

Implementation with common time-integration schemes

4.1 Detailed scheme

The scheme is now rewritten in a form more familiar to coders of projection methods. We begin with a backward Euler scheme to illustrate the basic algorithm. If a fully implicit scheme is used:

\[
\begin{align*}
  p_{0}^{n+1} &= p_{0}^{n} \\
  \frac{\bar{u}_{k+1}^{n+1} - u^{n}}{\Delta t} &= \nu \nabla^{2} \bar{u}_{k+1}^{n+1} - \bar{u}_{k+1}^{n+1} \cdot \nabla \bar{u}_{k+1}^{n+1} - \nabla p_{k}^{n+1} + f_{n+1} \\
  \frac{u_{k+1}^{n+1} - \bar{u}_{k+1}^{n+1}}{\Delta t} &= \nabla p_{k+1}^{n+1} - \nabla p_{k}^{n+1} \\
  \nabla^{2} p_{k+1}^{n+1} &= \nabla^{2} p_{k}^{n+1} - \frac{\nabla \cdot \bar{u}_{k+1}^{n+1}}{\Delta t}
\end{align*}
\]  

(4.1)

If the advective term is treated explicitly:
Again, for simplicity, we assume that the algorithm stop at a prescribed \( k = k_n \).

For the scheme outlined here, the velocity is only a by-product of the algorithm. In other words, it is an intermediate variable that can ultimately be expressed in term of the pressure-guess only and is only needed after the stopping criterion is fulfilled, and the algorithm is ready to move to the next time level. This statement is true regardless of whether the advection term is treated explicitly or implicitly. If the advection is treated explicitly, it is only dependent on the values of the velocity at the previous time level. If it is treated implicitly, velocity is solved for in the first fractional step (4.1). The velocity is nowhere to be "plugged back in". There exist methods for which the iteration is done for the linearized fully implicit equation and where the velocity needs to be updated. Those methods are called Galerkin schemes by Turek and are different from projection type schemes (Turek, 1999).

### 4.2 Generalization for IMEX schemes

**BDFq-IMEX schemes** The general scheme for the \( q \)th order BDFq-IMEX is

\[
\begin{align*}
    p_0^{n+1} &= p_{kn}^n \\
    \tilde{u}_k^{n+1} - u^n \frac{\Delta t}{\Delta t} &= \nu \nabla^2 \tilde{u}_k^{n+1} - \tilde{u}_k^{n+1} \cdot \nabla \tilde{u}_k^{n+1} - \nabla p_k^{n+1} + f^{n+1} \\
    u_k^{n+1} - \tilde{u}_k^{n+1} \frac{\Delta t}{\Delta t} &= \nabla P_k^{n+1} - \nabla p_k^{n+1} \\
    \nabla^2 p_k^{n+1} &= \nabla^2 p_k^{n+1} - \frac{\nabla \cdot \tilde{u}_k^{n+1}}{\Delta t}
\end{align*}
\]  

(4.2)

\[
\begin{align*}
    \alpha_k \tilde{u}_k^{n+1} - \sum_{j=0}^{q-1} \alpha_j u_k^{n-j} \frac{\Delta t}{\Delta t} &= \nu \nabla^2 \tilde{u}_k^{n+1} - \sum_{j=0}^{q-1} \beta_j \tilde{u}_k^{n-j} \cdot \nabla \tilde{u}_k^{n-j} - \nabla p_k^{n+1} + f^{n+1} \\
    \alpha_k u_k^{n+1} - \alpha_0 \tilde{u}_k^{n+1} \frac{\Delta t}{\Delta t} &= \nabla P_k^{n+1} - \nabla p_k^{n+1} \\
    \nabla^2 p_k^{n+1} &= \nabla^2 p_k^{n+1} - \alpha_q \frac{\nabla \cdot \tilde{u}_k^{n+1}}{\Delta t}
\end{align*}
\]  

(4.3)

36
Implicit-Explicit Runge-Kutta scheme  We follow the procedure outlined in Ueckermann and Lermusiaux (2016), with the difference that the forcing term is treated implicitly. For an IMEX scheme represented by the following Butcher tableaus

\[
\begin{array}{c|ccc}
0 & 0 & \ldots & 0 \\
c_1 & a_{s1}^{ex} & 0 & \ldots & 0 \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
1 & a_{s1}^{ex} & \ldots & a_{s(s-1)}^{ex} & 0 \\
\hline
b_1^{ex} & \ldots & b_{s-1}^{ex} & b_s^{ex} \\
\end{array}
\quad
\begin{array}{c|ccc}
0 & 0 & 0 & \ldots & 0 \\
c_1 & a_{s2}^{im} & a & 0 & \ldots & 0 \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\
1 & a_{s1}^{im} & \ldots & a_{s(s-1)}^{im} & a_s^{im} \\
\hline
b_1^{im} & \ldots & b_{s-1}^{im} & b_s^{im} \\
\end{array}
\]

(4.4)

The pressure-correction iteration unfolds as follows:

\[
\begin{align*}
    p_0^j &= p_{k_n}^{j-1} \\
    \frac{\bar{u}_{k+1}^i - u_{k_n}^i}{\Delta t} &= a_{ii}^{im} \left( \nu \nabla^2 \bar{u}_{k+1}^i - \nabla p_k^i + f^i \right) + \sum_{j=1}^{i-1} a_{ij}^{im} F_j^{im} + \sum_{j=1}^{i-1} a_{ij}^{ex} F_j^{ex} \\
    \frac{u_{k+1}^i - \bar{u}_{k+1}^i}{a_{ii}^{im} \Delta t} &= \nabla p_{k+1}^i - \nabla p_k^j \\
    \nabla^2 p_{k+1}^i &= \nabla^2 p_k^i - \frac{\nabla \cdot \bar{u}_{k+1}^i}{a_{ii}^{im} \Delta t}
\end{align*}
\]

(4.5)

where

\[
    f^i = f(t^n + c_i \Delta t) \\
    F_j^{im} = \nu \nabla^2 \bar{u}_{k_n}^j - \nabla \bar{p}_{k_n}^j + f^j \\
    F_j^{ex} = -\bar{u}_{k_n}^j \cdot \nabla \bar{u}_{k_n}^j
\]

(4.6)

The recombination stage is:
\[
\begin{align*}
\mathbf{u}^{n+1}_k &= \mathbf{u}^n_k + \Delta t \sum_{j=1}^{s} a_{ij}^{im} \mathbf{F}_j^{im} + \Delta t \sum_{j=1}^{s} a_{ij}^{ex} \mathbf{F}_j^{ex} \\
\mathbf{u}^{n+1}_k - \mathbf{\bar{u}}^{n+1}_k &= \nabla p^{n+1}_k - \nabla p^{s}_k \\
\nabla^2 p^{n+1}_k &= \nabla^2 p^{s}_k - \frac{\nabla \cdot \mathbf{u}^{n+1}_k}{b^{im}_s \Delta t}
\end{align*}
\] (4.7)

Note that the number of projection iterations within a Runge-Kutta stage does not need to be the same for every stage, in particular when a tolerance is chosen instead of a prescribed number of iterations.

As outlined by Ueckermann and Lermusiaux (2016), instead of computing \( \mathbf{F}_{i-1}^{im} \) after each stage, it is computationally more efficient to solve for it:

\[
\mathbf{F}_{i}^{im} = \frac{\mathbf{u}^{i+1}_k - \mathbf{u}^n_k}{a^{im}_{ii} \Delta t} - \frac{1}{a^{im}_{ii}} \sum_{j=1}^{i-1} a^{im}_{ij} \mathbf{F}_j^{im} + \sum_{j=1}^{i-1} a^{ex}_{ij} \mathbf{F}_j^{ex}
\] (4.8)

One additional advantage of solving for \( \mathbf{F}_{i-1}^{im} \) instead of computing it, is that it does not involve the value of the Laplacian of \( \mathbf{u}^i_k \), whose error near the boundary can be exacerbated when a finite-difference or finite-volume space discretization is involved. If this error is large, more iterations may be required to achieve the same accuracy.

For an example of the equations solved for the purpose of this work and more details on a possible implementation, see appendix.
Chapter 5

Test Cases

In this section we present numerical results illustrating the concepts outlined in the previous sections and supporting the theoretical results. We use a finite volume staggered grid for the space discretization for the Stokes problem and a QUICK advective scheme (Wesseling, 2009) when solving the Navier-Stokes problem.

5.1 Guermond and Shen test case

We start with a test case originally proposed by Guermond and Shen (2004) for the Stokes problem. If we denote \( \mathbf{u} = (u, v) \), we can prescribe the shape of the velocity and the pressure to be

\[
\begin{align*}
    u &= \pi \sin(t) \sin(2\pi y) \sin^2(\pi x) \\
    v &= -\pi \sin(t) \sin(2\pi x) \sin^2(\pi y) \\
    p &= \sin(t) \cos(\pi x) \sin(\pi y)
\end{align*}
\]  

The test case is well known, and has been extended to the Navier-Stokes problem for different methods (Ueckermann and Lermusiaux, 2016; Zhou, 2014; Celledoni et al., 2015; Colomés and Badia, 2016).
5.1.1 **Time step reduction vs. iterations**

One of the objectives of the study is to determine whether it is better to do more iterations or reduce the time step. We showed that asymptotically, if enough iterations are performed so that the global order of time convergence is increased, it is more advantageous to do more iterations. In some cases, the advantage is retained even if the order of convergence does not significantly increase.

![Figure 5-1](image)

*Figure 5-1: pressure error against the total number of iterations, using a BDF3 time integration solver. For each curve, the number of iteration during one time step is prescribed and fixed, while the time step varies. The first point of each curve represents the case where $\Delta t = .0625$. As we can see in Fig. 5-1, if enough iterations are performed, the iterative scheme offers better accuracy for the same cost. The number of iterations after which iterating is more competitive is not necessarily the same for different variables and different*
norms. In the case present, it is always cheaper to do more iterations for the $L^2$-norm of the velocity, whereas for the $L^\infty$ of the pressure there is only an advantage after 16 iterations.

5.1.2 Increased order of temporal convergence

If enough iterations are performed so that the splitting error becomes negligible compared to the error intrinsic to the time integration scheme, then only the latter is observed.

![Figure 5-2: Stokes problem. The order of convergence of the time-integration scheme is recovered when enough iterations are performed to eliminate the splitting error.](image)

In particular, the global temporal order of convergence is now that of the time integration scheme.
Figure 5-3: Navier-Stokes problem. The order of convergence of the time-integration scheme is recovered when enough iterations are performed to eliminate the splitting error.

5.1.3 Aitken acceleration

Aitken acceleration makes the error of the iterative scheme converge faster to the error of the pressure-Schur correction scheme with no relaxation.

Figure 5-4: convergence of the iterative pressure-correction scheme with and without Aitken acceleration. A BDF2 with time step scheme $\Delta t = .1$ at time $T = .1$ was used
In the example of Fig. 5-4, the scheme with Aitken acceleration reaches an error of $10^{-10}$ for the $L^\infty$-norm and of $10^{-12}$ for the $L^2$-norm in approximately 3 times fewer iterations.

5.1.4 Effects of the rotational correction

As we can see in Fig. 5-5, convergence of the iterative scheme is much faster with the rotational correction. As expected, when the time-step is reduced, the error of the scheme without rotational correction is closer to that of the scheme with rotational correction. Also, this result supports our conclusion that the scheme with rotational correction converges, as the values start to decrease without oscillation.
5.2 Modified Colomés and Badia test case

As can be seen in Fig. 5-1c and Fig. 5-1d, for small time-steps, the temporal and splitting errors for the velocity are negligible compared to the spatial error. On the one hand, this might prevent us from observing the correct order of temporal convergence such as in Fig. 5-3a, for the velocity. On the other hand, one must beware of spatial errors, because as explain in Guermond et al. (2006), if spatial resolution is not high enough, a false higher temporal convergence might be observed.

![Spatial error shapes obtained with discretizations of different orders](image)

Moreover, in some cases, the shape of the spatial error can be deceivingly similar to that of the temporal error. For example, in Fig. 5-6, we can see that the spatial error with our 4th order in space finite-volume discretization is used is similar to the temporal error in Fig. 2-1 and Fig. 3-1, while the shape of the error with a 2nd space discretization is closer to the shape of the solution.

The manufactured solution proposed by Colomés and Badia (2016) thus seems advantageous because the spatial error is greatly reduced. However, since the pressure does not depend on space at all, there is no observable splitting error when the problem is solved using a projection method. We thus modify the solution for the pressure to include some spatial dependence.
\[ u = x \sin \left( \frac{\pi t}{10} \right) \exp \left( \frac{t}{25} \right) \]
\[ v = -y \sin \left( \frac{\pi t}{10} \right) \exp \left( \frac{t}{25} \right) \]
\[ p = 1000(x + y) \exp \left( \frac{t}{25} \right) \] (5.2)

5.2.1 Stopping criterion

The stopping criterion indeed performs as expected. In particular, if the splitting error is much larger than the error due to the time integration scheme, 3\textsuperscript{rd} order in our case, it is possible to observe orders of convergence higher than 3, as in Fig. 5-7a.

Figure 5-7: temporal convergence of the pressure with a BDF3 for the Stokes problem

However, if the splitting error is completely eliminated, only the convergence of the time-integration scheme is observed, as we can see in Fig. 5-7b.
Fig. 5-8 shows typical examples of how the number of iterations required to achieve a certain order of temporal convergence using the stopping criterion changes with the time-step. As expected, higher order criteria require more iterations. We can clearly observe the logarithmic growth at the beginning. However, it seems like our prediction for very small steps is pessimistic, as the number actually appears to be decreasing.

### 5.2.2 Runge-Kutta schemes

This section shows results of the Runge-Kutta scheme applied to both the Stokes and Navier-Stokes problem. For the Stokes problem, a BDF3 and the $A$-stable third order DIRK (Alexander, 1977) schemes are compared, while for the Navier-Stokes problem BDF3-IMEX and the third order, four-stage $L$-stable RK-IMEX (Kennedy and Carpenter, 2003) schemes are compared.
In both cases, the stopping criterion performs well and we can observe the prescribed order of global temporal convergences.
Figure 5-10: Navier-Stokes problem with 3rd order stopping criterion
Chapter 6

Conclusion and future work

In this thesis, we approach the incremental pressure-correction as an iterative Richardson relaxation scheme, where the preconditioner is the Laplacian operator. For classical projection schemes, only one iteration is performed. For a given spatial discretization, the time step can be reduced in order to obtain more accurate results. Alternatively, the time-step can be fixed and more iterations of the relaxation scheme can be performed. We compare the computational cost of performing more than one iteration of the scheme as opposed to reducing the time-step in order to achieve the same accuracy.

We start by analyzing the convergence properties of the iterative scheme, and show that there is often an advantage in terms of computational effort in performing more iterations as opposed to using smaller time steps. We further confirm the benefit of the rotational correction, in our case in terms of convergence speed of the iterative scheme. We show how a dynamically adaptive stopping criterion allows some control of the temporal convergence of the scheme. In order to further reduce the computational cost of the scheme, we show how the method can be accelerated using an Aitken relaxation scheme. We also give examples of how the method can be conveniently embedded in an existing pressure-correction scheme for both multi-step and multistage time integration schemes.

We provide numerical results to support the theoretical results that we derive. We show that as more iterations are performed, the numerical boundary layer that
characterizes projection methods vanishes. Also, if enough iterations are performed, the error becomes virtually identical to that obtained with a fully coupled scheme or a pressure-Schur scheme, with a direct solve, that is with no relaxation. It is then possible to observe higher orders of temporal convergence, for both IMEX multi-stage and multi-step time integration schemes. We also show cases where the spatial error can be mistaken for the temporal error in some cases and that a high enough spatial resolution must be used in order to obtain sensible results.

In a quest for evermore accurate and computationally efficient methods, future work includes testing the iterative pressure-correction scheme with various space discretization scheme. The finite element Hybrid Discontinuous Galerkin (HDG) method in particular has proved a viable option for ocean modelling. The HDG method has been combined with projection methods and IMEX Runge-Kutta schemes (Ueckermann and Lermusiaux, 2016).

Accuracy is important not only for common CFD fields, but also in areas like path planning (Lolla et al., 2014; Subramani and Lermusiaux, 2016) where limited numerical diffusion is needed, data assimilation (Sondergaard and Lermusiaux, 2013a,b) and uncertainty prediction methods (Ueckermann et al., 2013) such as dynamically orthogonal methods, where numerical errors can grow to become spurious uncertainty modes. The techniques employed involved the computation of the flow field, for which accuracy is important.
Appendix

The 2.29 framework is a MATLAB structured grid Finite Volume solver capable of solving the governing two-dimensional incompressible Navier-Stokes equations. It is capable of solving problems involving tracer transport such as temperature and salinity, under the Boussinesq approximation. For the purpose of this work, a number of additions were made to the code, of which the most important is an inner loop to the (now) outer time-integration loop in order to perform iterations of the pressure-correction. As a complement to this inner loop, a stopping criterion and an option to use Aitken acceleration were added. Other additions include 4th order matrix operators and 3rd order IMEX-BDF, DIRK and IMEX Runge-Kutta time integrations schemes.

In this framework, the following equations are solved:

\[
\begin{align*}
\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0, \\
\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} &= -\frac{\partial P}{\partial x} + \nu_1 \frac{\partial^2 u}{\partial x^2} + \nu_2 \frac{\partial^2 u}{\partial y^2} + f(x, y, t)v + F_u(x, y, t) \\
\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} &= -\frac{\partial P}{\partial y} + \nu_1 \frac{\partial^2 v}{\partial x^2} + \nu_2 \frac{\partial^2 v}{\partial y^2} - \rho g - f(x, y, t)u + F_v(x, y, t) \\
\frac{\partial \rho}{\partial t} + u \frac{\partial \rho}{\partial x} + v \frac{\partial \rho}{\partial y} &= \kappa_1 \frac{\partial^2 \rho}{\partial x^2} + \kappa_2 \frac{\partial^2 \rho}{\partial y^2} + F_\rho(\rho, x, y, t)
\end{align*}
\]

Where \(u\) and \(v\) are the horizontal and vertical velocity field respectively, and are also used as subscripts to denote the corresponding direction. \(P\) is the pressure field, \(f\) the Coriolis force, while \(F\) is a generic forcing term imposed by the user. \(\nu\) and \(\kappa\) denote viscosity respectively.

The code accepts the parameters and variables through the following inputs:

\[
\begin{align*}
\text{app.nu} &= \nu_1, & \text{app.nu2} &= \nu_2, & \text{app.kappa} &= \kappa_1, & \text{app.kappa2} &= \kappa_2, \\
\text{app.g} &= g, & \text{app.Fcor} &= f(x, y, t), \\
u &= u, & v &= v, & P &= P, & \rho &= \rho.
\end{align*}
\]
A number of advection schemes are available, namely central, upwind, QUICK and TVD schemes. Second and fourth order matrix gradient, divergence and Laplacian operators are available for the various operations needed for projection schemes.

The pressure-correction method used to solve the Stokes problem is implemented in a three-step process:

\[
\begin{align*}
\left[ \frac{I}{\Delta t} - \nu \nabla^2 \right] \tilde{u}^{k+1} &= \frac{\mathbf{u}^k}{\Delta t} + \mathbf{F}^{k+1} \\
\nabla^2 P^{k+1} &= \frac{1}{\Delta t} \nabla \cdot \tilde{u}^{k+1} \\
\mathbf{u}^{k+1} &= \tilde{u}^{k+1} - \Delta t \nabla P^{k+1}
\end{align*}
\] (3)

where \( P \) in this case is a "pseudo-pressure". An improved version of the algorithm is the incremental pressure-correction scheme, used along with the rotational correction:

\[
\begin{align*}
\left[ \frac{I}{\Delta t} - \nu \nabla^2 \right] \tilde{u}^{k+1} &= \frac{\mathbf{u}^k}{\Delta t} - \nabla P^k + \mathbf{F}^{k+1} \\
\nabla^2 (q^{k+1}) &= \frac{1}{\Delta t} \nabla \cdot \tilde{u}^{k+1} \\
\mathbf{u}^{k+1} &= \tilde{u}^{k+1} - \Delta t \nabla q^{k+1} \\
P^{k+1} &= q^{k+1} + P^k - \nu \nabla \cdot \tilde{u}^{k+1}
\end{align*}
\] (6)

For the Navier-Stokes equations, the non-linear terms explicitly are treated explicitly: \( \mathbf{F}^{k+1} \approx -\mathbf{u}^k \cdot \nabla \mathbf{u}^k \). They are then scaled by the relevant IMEX coefficient.

The framework allows to work on complex domains thanks to a masking function. Possible boundary conditions include open, Dirichlet and Neumann boundary conditions that can vary in time and in space. To implement different manufactured solution, one simply needs to impose the corresponding forcing term.

For a more in depth outline of the capabilities of the framework, the reader is referred to the user manual Uckermann and Lermusiaux (2009).
Bibliography


