Technical Report

TR-2017-002

Inexact Algebraic Factorization Methods for the Steady Incompressible Navier Stokes Equations at Moderate Reynolds Numbers

by

Alex Viguerie, Alessandro Veneziani

Mathematics and Computer Science

EMORY UNIVERSITY

Inexact Algebraic Factorization Methods for the Steady Incompressible Navier Stokes Equations at Moderate Reynolds Numbers

Alex Viguerie[a,*], Alessandro Veneziani[a]

[a] *Department of Mathematics and Computer Science, Emory University, Atlanta, GA 30322-0001 , USA*

Abstract

The unsteady incompressible Navier-Stokes equations in primitive variables are often numerically solved by segregating the computation of velocity and pressure, according to either functional analysis arguments following the pioneering work of A.J. Chorin and R. Temam (STD, Split-Then-Discretize paradigm) or linear algebra arguments based on the inexact block factorization of the discretized problem (DTS, Discretize-Then-Split paradigm). The presence of the time derivative allows for the calibration of an appropriate approximation of the pseudo-differential operator of the pressure problem and excellent results in terms of both accuracy and efficiency have been obtained as witnessed by the abundant literature. The extension of the same segregated approach to the steady Navier-Stokes equations is unclear, unless a pseudo-time advancing formulation is undertaken. In this paper we present a methodology for a segregated computation of the primitive variables in a genuinely steady formulation, so to avoid iterations to get to the steady limit. The approach is largely inspired by the algebraic factorization of the unsteady problem (DTS approach), yet we detail specific settings required by the absence of the velocity time-derivative. The basic idea relies on the introduction of some parameters in a modified Picard linearization. We discuss stability bounds and the convergence of the segregated method to the unsplit solution. Several numerical results on different test cases confirm the efficiency of the procedure.

1. Introduction

The steady incompressible Navier-Stokes equations are receiving increasing attention as they are used also in fields where traditionally unsteady models are considered appropriate. In fact, sometimes time average information can be properly retrieved by a suitable steady computation, with the advantage of reducing computational costs (see e.g. [1]). This is particularly of interest when computation time is a sensible aspect, as in medical applications of computational fluid dynamics with a strong clinical orientation.

*Corresponding author
Email addresses: aviguer@emory.edu (Alex Viguerie), ale@mathcs.emory.edu (Alessandro Veneziani)

Generally, steady problems are more troublesome to analyze and solve. Terms originating from time derivatives are numerically beneficial and time discretization offers different ways for managing possible nonlinearities. For this reason, sometimes steady problems are solved by embedding the solution process into pseudo-evolutionary dynamics where the solution of interest is regarded as the asymptotic limit of a time-dependent problem (albeit with time-independent boundary conditions). This however requires computational costs for completing the time loop. In incompressible fluid dynamics, unsteady Navier-Stokes equations are often solved by segregated approaches that efficiently split the velocity and pressure computation. These approaches can be based on functional principles (Ladhyzhenskaja theorem or Helmholtz decomposition principle) as well as algebraic arguments. The former approach (Split-Then-Discretize paradigm) has a more elegant and solid mathematical background in functional analysis allowing for a more complete error analysis for specific boundary value problems [2, 3, 4, 5]; the latter one (Discretize-Then-Split paradigm) has many practical advantages, specifically in the management of general boundary conditions of interest in engineering problems [6, 7, 8, 9, 10, 11].

These approaches are specifically devised for the unsteady problems and the extension to the steady case is not immediate since they rely on the presence of the discretized time derivative. In this paper we propose an algebraic factorization approach for steady problems intended to mimic the DTS paradigm in a genuinely steady setting (no pseudo-evolution), retaining the advantages of segregated computation appreciated in the unsteady solution. The core of the method is in the approximation of the pressure Schur complement of the steady incompressible Navier-Stokes matrix.

We introduce the method (Sect. 2.2) as a segregation of the generic iteration of an iterative solver, required by the nonlinear convective term. In particular, we refer to the popular Picard iteration, the extension to the Newton method being addressed too. An appropriate modification of the original Picard method needs to be introduced, and its convergence to the exact solution is investigated in Sect. 3. Then, we investigate the convergence properties of the splitting method (Sect. 4), pointing out how for moderate Reynolds numbers it can be a viable and effective approach for the solution of the steady problem. Numerical results of Sect. 5 confirm our findings.

2. The Steady Incompressible Navier-Stokes Problem and its numerical solution

Let $\Omega \in \mathbb{R}^d$ be a suitable domain, where $d = 2, 3$. The steady incompressible Navier-Stokes boundary problem (for a Newtonian fluid) reads

$$\begin{aligned} -\nu\nabla\cdot(\nabla\boldsymbol{u}+\nabla\boldsymbol{u}^T)+(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}+\nabla p &= \boldsymbol{f} \quad \text{in } \Omega \\ \nabla\cdot\boldsymbol{u} &= 0 \quad \text{in } \Omega \\ \mathcal{B}\boldsymbol{u} &= \boldsymbol{g} \quad \text{on } \partial\Omega, \end{aligned} \tag{2.1}$$

where ν is the kinematic viscosity, $\boldsymbol{u}$ is a d-vector representing the unknown velocity and the scalar function p is the pressure, both generally function of space. Here, $\mathcal{B}$ is an appropriate trace operator defining the different types of boundary conditions of interest. In particular, we consider standard options, namely

$$\text{Dirichlet conditions} : \boldsymbol{u}(\Gamma_D) = \boldsymbol{g}_D$$

and

$$\text{Neumann conditions} : \left(p\boldsymbol{n} - \nu\nabla\left(\boldsymbol{u} + \boldsymbol{u}^T\right)\cdot\boldsymbol{n}\right)(\Gamma_N) = \boldsymbol{g}_N$$

where $\partial\Omega = \Gamma_D \cup \Gamma_N$, $\Gamma_D \cap \Gamma_N = \emptyset$, $\boldsymbol{n}$ is the unit normal vector to the boundary and $\boldsymbol{g}_D$ and $\boldsymbol{g}_N$ are given. In the following, we will assume for simplicity data $\boldsymbol{g} = 0$ on both Dirichlet and Neumann boundaries.

The theory of this problem is addressed e.g. in [12], Sect. IV.2, [13]. The weak formulation is promptly obtained by standard arguments. Let V be the Sobolev space $\left(H^1_{\Gamma_D}\right)^d(\Omega)$ and $Q = L^2(\Omega)$ (in the case $\Gamma_N = \emptyset$, Q is the space $L^2_0(\Omega)$ of the null-average L^2 functions). Let us define

$$a(\cdot,\cdot) : V \times V \to \mathbb{R}, \text{s.t. } a(\boldsymbol{w},\boldsymbol{v}) = \nu\int_\Omega (\nabla\boldsymbol{w} + \nabla\boldsymbol{w}^T) : \nabla\boldsymbol{v}, \tag{2.2}$$

$$b(\cdot,\cdot) : V \times Q \to \mathbb{R}, \text{s.t. } b(\boldsymbol{w},q) = -\int_\Omega \nabla\cdot\boldsymbol{w}q, \tag{2.3}$$

$$c(\cdot,\cdot,\cdot) : V \times V \times V \to \mathbb{R}, \text{s.t. } c(\boldsymbol{w},\boldsymbol{v},\boldsymbol{z}) = \int_\Omega (\boldsymbol{w}\cdot\nabla)\boldsymbol{v}\cdot\boldsymbol{z}. \tag{2.4}$$

Then the weak formulation of the problem reads: find $\boldsymbol{u} \in V$ and $p \in Q$ s.t. for any $\boldsymbol{v} \in V, q \in Q$

$$a(\boldsymbol{u},\boldsymbol{v}) + c(\boldsymbol{u},\boldsymbol{u},\boldsymbol{v}) + b(\boldsymbol{v},p) + b(\boldsymbol{u},q) - (\boldsymbol{f},\boldsymbol{v}) = 0$$

where $(\cdot,\cdot)$ is the usual scalar product in L^2. See e.g. [12] Chap. IV or [14] Chaps. 9,10 for more details. In particular, we recall that the bilinear form $a(\cdot,\cdot)$ is continuous and coercive in $V \times V$ if $\Gamma_D \neq \emptyset$. In particular, this means that there exists a positive constant C_a s.t.[1]

$$a(\boldsymbol{v},\boldsymbol{v}) \geq C_a\|\boldsymbol{v}\|^2_V, \forall\boldsymbol{v} \in V.$$

Also, the trilinear form is continuous too, so that there exists a constant C_c such that

$$|c(\boldsymbol{w},\boldsymbol{v},\boldsymbol{u})| \leq C_c\|\boldsymbol{w}\|_V\|\boldsymbol{v}\|_V\|\boldsymbol{u}\|_V.$$

[1] We notice that this inequality is the standard Poincaré inequality in the usual writing of the diffusion operator for the Stokes problem, i.e. the Laplace operator - as popular in many references, e.g. [13]. For the complete writing as we use here, which is correct in the weak formulation, the same inequality follows from a prompt application of the Korn inequality.

Finally, for a purely Dirichlet boundary value problem, if we have divergence free vectors, we get the skew-symmetry of $c(\cdot,\cdot,\cdot)$, i.e.

$$c(\boldsymbol{w},\boldsymbol{v},\boldsymbol{u}) = -c(\boldsymbol{w},\boldsymbol{u},\boldsymbol{v}) \Rightarrow c(\boldsymbol{w},\boldsymbol{v},\boldsymbol{v}) = 0, \quad \forall \boldsymbol{v},\boldsymbol{u} \in H_0^1(\Omega).$$

For the numerical solution of the problem we need to both discretize the differential operators and linearize the nonlinear quadratic convective term. In particular, we refer to the Finite Element Method (FEM), however any discretization technique can be used in what follows. With FEM, we postulate the velocity-pressure solution to be piecewise polynomial over a reticulation $\mathcal{T}$ of the domain Ω featuring a representative mesh size h. To guarantee the discrete problem to be nonsingular, we assume that the polynomial degrees for the velocity and pressure approximation fulfill the so called the Babushka-Brezzi condition [14, 12]. For instance, we may resort to piecewise quadratic velocity and piecewise (continuous) linear pressure fields (Taylor-Hood pair). From now on, we denote by $\boldsymbol{u}_h$ and p_h the discrete velocity and pressure belonging to an inf-sup compatible finite dimensional space pair $V_h \times Q_h$. We denote by $N_u(N_p)$ the number of degrees of freedom for the discrete velocity (pressure). The discrete form of problem (2.1) is then given by the following *saddle-point system*

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{k+1} \\ \boldsymbol{p}^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix} \tag{2.5}$$

where $A = K + C(\boldsymbol{u})$ is a $N_u \times N_u$ matrix. K is commonly called the *stiffness matrix* and corresponds to the diffusive term in (2.1). For $\boldsymbol{\varphi}_i$ being the generic FEM quadratic basis function associated with the $i-$th degree of freedom, we have $[K]_{ij} = a(\boldsymbol{\varphi}_j, \boldsymbol{\varphi}_i)$. Similarly, $C(\boldsymbol{u})$ corresponds to the convective term in (2.1),

$$[C(\boldsymbol{u})]_{ij} = c(\boldsymbol{u}, \boldsymbol{\varphi}_j, \boldsymbol{\varphi}_i).$$

Finally, for ψ_i to be the generic basis (piecewise linear) function to represent the pressure, we have B to be a $N_p \times N_u$ matrix defined by

$$[B]_{ij} = b(\boldsymbol{\varphi}_j, \psi_i).$$

By standard arguments based on the coercivity of $a(\cdot,\cdot)$, K is readily proved to be symmetric positive-definite (s.p.d.) and C to be skew-symmetric for $\Gamma_N = \emptyset$.

The nonlinear algebraic system can be linearized by standard techniques. Among the others, we mention Picard-type and Newton-type schemes (see e.g. [14] for more details). In the former case, the generic nonlinear iteration reads: given $\boldsymbol{u}^{(k)}$, solve for $k+1$ iteration

$$\begin{bmatrix} A_1 & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{(k+1)} \\ \boldsymbol{p}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix}, \tag{2.6}$$

with $A_1(\boldsymbol{u}^{(k)}) = K + C(\boldsymbol{u}^{(k)})$. In the Newton linearization, we need to introduce the matrix $\hat{C}(\boldsymbol{u}^{(k)})$ defined as

$$[\hat{C}(\boldsymbol{u})]_{ij} = c(\boldsymbol{\varphi}_j, \boldsymbol{u}, \boldsymbol{\varphi}_i)$$

and the generic iteration resorts to: given $\boldsymbol{u}^{(k)}$, solve

$$\begin{bmatrix} K + C(\boldsymbol{u}^{(k)}) + \hat{C}(\boldsymbol{u}^{(k)}) & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{(k+1)} \\ \boldsymbol{p}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} + C(\boldsymbol{u}^{(k)})\boldsymbol{u}^{(k)} \\ \boldsymbol{0} \end{bmatrix}. \tag{2.7}$$

The Newton approach is generally faster, with the error at the generic iteration $k+1$ scaling as the square of the error at the previous iteration, however it generally requires a good initial guess $\boldsymbol{u}^{(0)}$. This can be possibly identified by a continuation method [12]. The Picard approach converges more slowly (linear scaling of the error with the error at the previous iteration) but is generally less restrictive for the initial guess. Hereafter, we refer to the Picard approach for simplicity, with the algebraic arguments we introduce being applicable to the Newton system (2.7) as well.

The saddle-point system (2.6) admits the following block LU-factorization (see e.g. [15]):

$$\begin{bmatrix} A_1 & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{(k+1)} \\ \boldsymbol{p}^{(k+1)} \end{bmatrix} = \begin{bmatrix} A_1 & 0 \\ B & -BA_1^{-1}B^T \end{bmatrix} \begin{bmatrix} I & A_1^{-1}B^T \\ 0 & I \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{(k+1)} \\ \boldsymbol{p}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix} \tag{2.8}$$

Even if formally this block LU factorization yields a segregated computation of velocity and pressure, it is not practical. The system for the pressure requires in fact the solution of the so called (pressure) *Schur complement* $\Sigma_1 = -BA_1^{-1}B^T$. This matrix changes at each iteration and cannot be computed, as A_1^{-1} is neither available nor worth computing, for both computational and storage costs induced by the fill-in of the inversion. A possible approach is to solve the Schur complement by an iterative method so that when the matrix needs to be multiplied by a vector, we instead solve a system in A_1. The nested iterations obtained in this way (as A_1 is usually solved by an iterative method too) rapidly increase the computational costs and the advantage of segregation is easily lost.

In the unsteady problem, a similar situation is encountered at each step of the time discretization. However, in that case there is a natural way for circumventing the problem by introducing a suitable approximation of the Schur complement that does not affect the overall time accuracy, yet yields an efficient solution - as we address in the next section. The present work relies on the identification of an appropriate iterative linearization method combined with an approximation of the Schur complement that preserves the convergence of the nonlinear iteration with the advantage of cost reduction introduced by the segregation of velocity/pressure computation.

2.1. Velocity-pressure segregation techniques for the unsteady problem

The unsteady problem reads

$$\begin{aligned} \frac{\partial \boldsymbol{u}}{\partial t} - \nu\nabla\cdot(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T) + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nabla p = \boldsymbol{f} \quad & in\ \Omega\times[t_0,t_n] \\ \nabla\cdot\boldsymbol{u} = 0 \quad & in\ \Omega\times[t_0,t_n] \\ \mathcal{B}\boldsymbol{u} = \boldsymbol{g} \quad & on\ \partial\Omega\times[t_0,t_n] \\ \boldsymbol{u}(\boldsymbol{x},0) = \boldsymbol{u}_0 \quad & at \quad t = t_0 \end{aligned} \tag{2.9}$$

With the additional dependence on time, we now require the prescription of an appropriate initial condition $\boldsymbol{u}_0$. The functional environment of the time dependent problem requires the introduction of vector spaces. As the main focus of the paper is on the steady case, we omit these details, referring to the literature [12, 16]. For the projection method (STD approach) we refer additionally to e.g. [17, 18, 19, 20, 21, 22].

For the DTS approach, we discretize with the finite element method in space, and use a finite difference scheme for the time discretization. For simplicity, we use a simple first-order semi-implicit Euler scheme here, so at each time step we solve the system

$$\begin{bmatrix} A_t(\boldsymbol{u}^n) & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{n+1} \\ \boldsymbol{p}^{n+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} + \frac{1}{\Delta t} M \boldsymbol{u}^n \\ \boldsymbol{0} \end{bmatrix} \tag{2.10}$$

with $A_t(\boldsymbol{u}^n) = \frac{1}{\Delta t} M + A_1^n$ where Δt is the time step, $A_1^n \equiv K + C(\boldsymbol{u}^n)$ is as before and M (called the *mass matrix*) comes from the bilinear form $m(\boldsymbol{u}, \boldsymbol{v}) : V \times V \to \mathbb{R}$:

$$m(\boldsymbol{u}, \boldsymbol{v}) = \int_\Omega \boldsymbol{u} \cdot \boldsymbol{v} \tag{2.11}$$

If I denotes the identity, observe that:

$$\frac{1}{\Delta t} M + A_1^n = \frac{1}{\Delta t} M (I + \Delta t M^{-1} A_1^n) \Rightarrow (\frac{1}{\Delta t} M + A_1^n)^{-1} = (I + \Delta t M^{-1} A_1^n)^{-1} \Delta t M^{-1} \tag{2.12}$$

For Δt sufficiently small, we can ensure that $\rho(\Delta t M^{-1} A_1^n) < 1$ and exploit the well-known Neumann series identity for $(I + \Delta t M^{-1} A_1^n)^{-1}$ to observe that

$$(\frac{1}{\Delta t} M + A_1^n)^{-1} = \big(\sum_{j=0}^{\infty} (-1)^j (\Delta t M^{-1} A_1^n)^j\big) \Delta t M^{-1} \approx \Delta t M^{-1} \tag{2.13}$$

with a first order truncation of the Neumann expansion. Hence we can regard $\Delta t M^{-1}$ as a first-order approximation for A_t. With this approximation in mind we then consider a block factorization of (2.10) similar to (2.8) defined as:

$$\begin{bmatrix} A_t & B^T \\ B & 0 \end{bmatrix} = \begin{bmatrix} A_t & 0 \\ B & -B H_1 B^T \end{bmatrix} \begin{bmatrix} I & H_2 B^T \\ 0 & I \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{n+1} \\ \boldsymbol{p}^{n+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} + \frac{1}{\Delta t} M \boldsymbol{u}^n \\ \boldsymbol{0} \end{bmatrix} \tag{2.14}$$

For $H_1 = H_2 = A_t^{-1}$, we get the exact block factorization of the original system (2.10). The *Inexact Factorization methods* seek to replace H_1 and H_2 with approximations of A_t^{-1}, a natural choice being $\Delta t M^{-1}$. Replacing both H_1 and H_2 with $\Delta t M^{-1}$ gives the so called *Algebraic Chorin-Temam* method, as it was pointed out that there is a formal analogy with the original Chorin-Temam projection scheme

based on the Ladhyzhenskaja decomposition theorem [2, 3, 4, 23, 5]. Using $\Delta t M^{-1}$ for H_1 and A_t^{-1} for H_2 is another viable option leading to the so-called *Yosida* method. These methods are discussed and analyzed in detail in [7, 8].

The common denominator of this approach is the approximation of the Schur complement having the velocity mass matrix as a critical ingredient, with the parameter Δt modulating the accuracy of the approximation. Variants to the original scheme have been eventually introduced to guarantee higher order of time accuracy and to trigger time adaptivity [24, 25, 9, 10, 26, 11]. It is worth noting that the higher accuracy is not obtained by including more terms in the Neumann expansion, as that would cause some stability and practical problems [27]. In fact, the approximation of the Schur complement with A_t^{-1} replaced by the inverse of the mass matrix is particularly convenient, as the mass matrix is not only s.p.d., but also diagonal for finite difference, spectral and lumped finite elements. This allows for specific ad hoc solvers based on the sparse QR factorization of Σ [11].

The absence of the mass matrix in the stationary problem prevents an immediate extension of the approach to the steady case. Thus, the goal here is to find a way that accelerates the computation of the steady solution by splitting velocity and pressure with a convenient approximation of the Schur complement.

2.2. Velocity-pressure segregation techniques for the steady problem

Following the unsteady approach, for the block-factorized system (2.8), we approximate the Schur complement with the following arguments.

We first replace the Picard iteration with the following variant. Let $\alpha \in [0, 1]$ be a positive parameter . The *modified Picard scheme reads*: given the guess $({\boldsymbol{u}_h}^{(k)}, {p_h}^{(k)}) \in V_h \times Q_h$, find $({\boldsymbol{u}_h}^{(k)}, {p_h}^{(k)})$ in the same space pair s.t.

$$\begin{aligned} &a({\boldsymbol{u}_h}^{(k+1)}, \boldsymbol{v}_h) + \alpha c({\boldsymbol{u}_h}^{(k)}, {\boldsymbol{u}_h}^{(k+1)}, \boldsymbol{v}_h) + b({p_h}^{(k+1)}, \boldsymbol{v}_h) = (\boldsymbol{f}, \boldsymbol{v}_h) - (1-\alpha) c({\boldsymbol{u}_h}^{(k)}, {\boldsymbol{u}_h}^{(k)}, \boldsymbol{v}_h) \\ &b(q_h, {\boldsymbol{u}_h}^{(k+1)}) = 0 \end{aligned} \tag{2.15}$$

for any $\boldsymbol{v}_h \in V_h$ and $q_h \in Q_h$.

At the algebraic level, for $A_\alpha \equiv K + \alpha C(\boldsymbol{u}^{(k)})$ (we do not emphasize the dependence on k for easiness of notation), the modified Picard iteration reads

$$\begin{bmatrix} A_\alpha & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{k+1} \\ \boldsymbol{p}^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} - (1-\alpha) C(\boldsymbol{u}^{(k)}) \boldsymbol{u}^{(k)} \\ \boldsymbol{0} \end{bmatrix} \tag{2.16}$$

For $\alpha = 1$ we clearly have the standard Picard scheme, while for $\alpha = 0$ we have a full explicit treatment of the nonlinearity. While the convergence rate is linear for the original scheme[13], when α approaches 0,

we expect the pool of possible initial good guesses for the convergence to shrink. This will be confirmed in the analysis below.

Notice that[2]

$$A_\alpha = K(I + \alpha K^{-1} C) \tag{2.17}$$

By a virtually identical argument as for the unsteady case (with Δt), if α is chosen such that $\rho(\alpha K^{-1} C) < 1$, then it is justified to set $(K + \alpha C)^{-1} \approx K^{-1}$ as a first order truncation of the Neumann expansion. With the LU factorization approximated in this way, we propose the following algorithm for the solution of the single iteration of (2.16):

$$\begin{aligned} &A_\alpha \boldsymbol{u}^* = \boldsymbol{f} - (1-\alpha) C \boldsymbol{u}^{(k)} \\ &B K^{-1} B^T p^{(k+1)} = B \boldsymbol{u}^* \\ &K \boldsymbol{u}^{(k+1)} = K \boldsymbol{u}^* - B^T p^{(k+1)} = \boldsymbol{f} - (1-\alpha) C \boldsymbol{u}^{(k)} - \alpha C \boldsymbol{u}^* - B^T p^{(k+1)} \end{aligned} \tag{2.18}$$

Since K is not dependent on the current iteration, we do not need to reassemble any associated preconditioner for the approximated Schur complement at each step. Additionally, our approximated Schur complement is guaranteed to be s.p.d., allowing for the use of efficient iterative methods.

The sequence of systems (2.18) provides the backbone of our method. It corresponds to what we previously called *Algebraic Chorin-Temam* scheme in the unsteady case, as we use the same approximation of A^{-1} in both its occurrences of the factorization. However, a substantial difference must be noted between the steady and unsteady schemes. In the (algebraic) Chorin-Temam method, the final operator recovering the end-of-step velocity is a projection. In the algebraic scheme, this corresponds to the velocity mass matrix. A drawback of this is that there is virtually no control on the tangential boundary conditions for the end-of-step velocity. For this reason, some authors preferred the intermediate velocity to be qualified as the physical field, despite it not being divergence free [17, 18]. On the contrary, in the Yosida approach, for $H_2 = A_1^{-1}$, the end-of-step velocity is solved by a second order operator and the control of the end-of-step velocity boundary conditions is retained.

In the steady case, also following the strategy *á la* Chorin-Temam, the final step is obtained by solving a second order (diffusion) operator, so the enforcement of the boundary conditions to the end-of-step velocity is fulfilled. Because of this, we find that the Yosida-like strategy in the steady case is not justified and we do not pursue it any longer.

If $\mathcal{A}_\alpha$ denotes the matrix of the modified Picard scheme

$$\mathcal{A}_{\alpha,u} \equiv \begin{bmatrix} A_\alpha & B^T \\ B & 0 \end{bmatrix},$$

[2]The dependence of C on the iterate $\boldsymbol{u}^{(k)}$ is understood.

the matrix resulting from the splitting scheme reads

$$\mathcal{A}_{\alpha,s} \equiv \begin{bmatrix} A_\alpha & 0 \\ B & -BK^{-1}B^T \end{bmatrix} \begin{bmatrix} I & K^{-1}B^T \\ 0 & I \end{bmatrix} = \begin{bmatrix} A_\alpha & A_\alpha K^{-1}B^T \\ B & 0 \end{bmatrix} = \mathcal{A}_{\alpha,u} + \begin{bmatrix} 0 & \boxed{\alpha C K^{-1}B^T} \\ 0 & 0 \end{bmatrix}$$

As the boxed term points out, the splitting error affects only the block (1,2) of the original system. This means that the approximated end-of-step velocity is truly (discretely) divergence free. In addition, the error term scales with α so, as expected, vanishes for $\alpha \to 0$.

To enhance the convergence rate and the robustness of the method we introduce below two variants that eventually we combine in the final formulation of the scheme.

Remark

At the continuous level, the first step of our splitting scheme reads: given $\boldsymbol{u}^{(k)}$

$$-\nabla \cdot \left(\nu \nabla \hat{\boldsymbol{u}}^{(k+1)} + \nabla^T \hat{\boldsymbol{u}}^{(k+1)}\right) + \alpha \left(\boldsymbol{u}^{(k)} \cdot \nabla\right) \hat{\boldsymbol{u}}^{(k+1)} = \boldsymbol{f} + (\alpha - 1) \left(\boldsymbol{u}^{(k)} \cdot\right) \nabla \boldsymbol{u}^{(k)} \tag{2.19}$$

This formulation sheds light on the role of α and consequently its selection. The Péclet number of this problem reads

$$\mathbb{P} = \frac{\alpha ||\boldsymbol{u}^{(k)}||_\infty h}{2\nu}$$

so we argue that the numerical approximation with standard finite elements is convective stable when

$$\alpha h < \frac{2\nu}{||\boldsymbol{u}^{(k)}||_\infty}.$$

Since α is selected by the user, this inequality leads to two conclusions:

1. α has a stabilizing effect, so that it can mitigate the need of a fine mesh required by the original Picard scheme;
2. an adaptive strategy for the automatic selection of this parameter, that is then assumed to depend on k, may be based on this inequality, and is an interesting follow up of the present work.

Remark

A similar approach can be applied to the Newton method. Weighting the linearized term of the Newton iteration by the parameter α and evaluating the residual term weighted by $1-\alpha$ at the previous iteration, the modified Newton scheme reads given the guess $({\boldsymbol{u}_h}^{(k)}, {p_h}^{(k)}) \in V_h \times Q_h$, find $({\boldsymbol{u}_h}^{(k)}, {p_h}^{(k)})$ in the same space pair s.t.

$$\begin{aligned} &a({\boldsymbol{u}_h}^{(k+1)}, \boldsymbol{v}_h) + \alpha \left(c({\boldsymbol{u}_h}^{(k)}, {\boldsymbol{u}_h}^{(k+1)}, \boldsymbol{v}_h) + c({\boldsymbol{u}_h}^{(k+1)}, {\boldsymbol{u}_h}^{(k)}, \boldsymbol{v}_h)\right) + b({p_h}^{(k+1)}, \boldsymbol{v}_h) = \\ &\qquad (\boldsymbol{f}, \boldsymbol{v}_h) + (2\alpha - 1) c({\boldsymbol{u}_h}^{(k)}, {\boldsymbol{u}_h}^{(k)}, \boldsymbol{v}_h) \end{aligned} \tag{2.20}$$

$$b(q_h, {\boldsymbol{u}_h}^{(k+1)}) = 0$$

for any $\boldsymbol{v}_h \in V_h$ and $q_h \in Q_h$.

This is consistent with the original problem for any value of α (as promptly realized by setting ${\boldsymbol{u}_h}^{(k+1)} = {\boldsymbol{u}_h}^{(k)}$).

As the Newton method requires an initial guess good enough, we expect the choice of α - that affects the pool of good guesses, as we will see later - to be even more delicate with this approach. On the other hand, the Newton method is faster and the actual effect of this improved convergence rate on the splitting algorithm will be investigated elsewhere.

2.2.1. The incremental variant

By mimicking a popular approach for the unsteady problem [25, 28], we consider the following *incremental pressure* form of the problem, letting $\delta\boldsymbol{p}^{k+1} = \boldsymbol{p}^{k+1} - \boldsymbol{p}^k$, given $\boldsymbol{u}^{(k)}$ and $p^{(k)}$, solve

$$\begin{bmatrix} K + \alpha C & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{k+1} \\ \delta\boldsymbol{p}^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} - (1-\alpha)C\boldsymbol{u}^k - B^T\boldsymbol{p}^k \\ \boldsymbol{0} \end{bmatrix} \tag{2.21}$$

The relevance of the incremental approach is promptly realized by noting that the consistency error affects only the block (1,2) of the matrix of the system. Should the method be convergent, the splitting error of the nonincremental scheme will gather on the momentum equation as $\alpha C K^{-1} B^T p$ where p is the convergence pressure. This vanishes for $\alpha \to 0$. In the case of the incremental scheme, the solution $\delta p \to 0$ when approaching convergence. This means that the convergence solution solves the exact problem for all the values of α. The incremental scheme is therefore "strongly consistent", as the convergence solution is eaxct (up to numerical discretization errors) regardless the value of the parameter.

2.2.2. Relaxation

The parameter α is introduced to modulate the weight of the convective component and to make the Schur complement approximation acceptable. While for $\alpha = 0$ the splitting error is null, the original Picard convergent (and robust) iteration is obtained for $\alpha = 1$. The proper selection of this parameter is therefore critical for the overall performance of the method. To add some flexibility and enhancement, we introduce a second parameter for the purpose of enhancing the convergence. Denoting by $\hat{\boldsymbol{u}}^{k+1}$ the velocity field obtained by the basic unrelaxed (either incremental or not) scheme, the relaxed scheme computes

$$\boldsymbol{u}^{k+1} = \gamma\hat{\boldsymbol{u}}^{k+1} + (1-\gamma)\boldsymbol{u}^k$$

with $\gamma \in (0, 1]$. The actual bound on the relaxation parameter γ that guarantees convergence will be discussed in the next section.

The final splitting incremental relaxed scheme then reads as in Algorithm 1.

Remark

Algorithm 1 Incremental relaxed segregated scheme for the steady Navier-Stokes equations.

while Convergence criterion not met: **do**
  solve $A_\alpha \hat{\tilde{\boldsymbol{u}}}^{k+1} = \boldsymbol{f} + (\alpha - 1)C\boldsymbol{u}^k - B^T\boldsymbol{p}^k$
  solve $BK^{-1}B^T\delta\boldsymbol{p}^{k+1} = B\hat{\tilde{\boldsymbol{u}}}^{k+1}$
  solve $K\hat{\boldsymbol{u}}^{k+1} = K\hat{\tilde{\boldsymbol{u}}}^{k+1} - B^T\delta\boldsymbol{p}^{k+1}$
  update the pressure: $\boldsymbol{p}^{k+1} = \delta\boldsymbol{p}^{k+1} + \boldsymbol{p}^k$
  relax the velocity: $\boldsymbol{u}^{k+1} = \gamma\hat{\boldsymbol{u}}^{k+1} + (1-\gamma)\boldsymbol{u}^k$
  check tolerance
end while

The expected bottleneck in the algorithm is represented by the approximated Schur complement $BK^{-1}B^T$. For K to be the pure diffusion operator (i.e. for $\nabla \cdot (\nabla + \nabla^T)$ reduced to the Laplace operator), we may notice that this matrix is spectrally equivalent to a pressure mass matrix [14]. In principle, we could replace this matrix with a lumped pressure mass matrix so as to minimmize the cost of the solution of the system. Nevertheless, in our numerical experiments (not shown) we found that this choice is overall detrimental for the performance of the entire splitting scheme. While we will further investigate this option and its smart implementation, we notice however that the pressure mass matrix is a natural and excellent preconditioner for the approximate Schur complement. It will be used in the entire section on Numerical Results.

2.2.3. The Yosida-like variant

In principle, following [7], we can think of a variant to the splitting scheme where the third step is not approximated:

Algorithm 2 Steady Yosida Method.

while Convergence criterion not met: **do**
  solve $A_\alpha \hat{\tilde{\boldsymbol{u}}}^{k+1} = \boldsymbol{f} + (\alpha - 1)C\boldsymbol{u}^k - B^T\boldsymbol{p}^k$
  solve $BK^{-1}B^T\delta\boldsymbol{p}^{k+1} = B\hat{\tilde{\boldsymbol{u}}}^{k+1}$
  solve $A_\alpha\hat{\boldsymbol{u}}^{k+1} = \boldsymbol{f} + (\alpha - 1)C\boldsymbol{u}^k - B^T\boldsymbol{p}^k - B^T\delta\boldsymbol{p}^{k+1}$
  update the pressure: $\boldsymbol{p}^{k+1} = \delta\boldsymbol{p}^{k+1} + \boldsymbol{p}^k$
  relax the velocity: $\boldsymbol{u}^{k+1} = \gamma\hat{\boldsymbol{u}}^{k+1} + (1-\gamma)\boldsymbol{u}^k$
  check tolerance
end while

In this case, the splitting error is gathered in the mass conservation equation as opposed to the momentum equation as in our proposed scheme. In the unsteady splitting method, this leads to the Yosida scheme. As pointed out previously, the Yosida approach has the advantage of enforcing the

entire boundary conditions to the end-of-step velocity. In the present case, our method also enforces the boundary conditions, and is therefore more justified. In particular, the third step of our method requires the solution of a s.p.d. system, which can be solved more efficiently than the corresponding system in Algorithm 2. For this reason we do not analyze in detail this variant.

3. Analysis of the modified Picard scheme

We provide a complete convergence analysis of the modified Picard scheme, the original Picard iteration being analyzed in [13]. We follow similar guidelines for the proof.

We assume in this case for simplicity that the boundary conditions are of (homogeneous) Dirichlet type over the entire boundary $\partial\Omega$, so that $V = H_0^1(\Omega)$ and $Q = L^2 \setminus \mathbb{R}$. As a consequence, as mentioned earlier the trilinear form is skew-symmetric, i.e.

$$c(\boldsymbol{v}, \boldsymbol{u}, \boldsymbol{u}) = 0$$

for any $\boldsymbol{v} \in V$ and divergence free velocity $\boldsymbol{u} \in V$.

For a full Dirichlet problem, it is possible to prove with standard arguments that the following bound holds for the finite element solution $\boldsymbol{u}_h$

$$\|\boldsymbol{u}_h\| \leq \frac{\|\boldsymbol{f}\|_{-1}}{\nu C_a} \tag{3.1}$$

where $\|\cdot\|_{-1}$ denotes the norm of the dual space of $H_0^1(\Omega)$.

Also, we postulate that the Reynolds number is moderate, and in particular we assume that [13]

$$\delta \equiv \nu^2 C_a^2 - C_b \|\boldsymbol{f}\|_{-1} > 0. \tag{3.2}$$

Under this assumption, the solution to the Navier-Stokes steady problem is unique. In particular, this assumption implies that the coefficient $\bar{\delta}$ defined hereafter is

$$\bar{\delta} \equiv \frac{\nu^2 C_a^2}{C_b \|\boldsymbol{f}\|_{-1}} > 1. \tag{3.3}$$

As an additional hypothesis, we assume that

$$\bar{\delta} > 2. \tag{3.4}$$

Theorem 3.1. *Let $u_h^{(0)} \times p_h^{(0)}$ be an initial guess for the modified Picard scheme, belonging to $V_h \times Q_h$. Under the assumptions* (3.2), (3.4)*, for $\alpha \in [0, 1]$, and $k > 0$ the sequence $u_h^{(k)} \times p_h^{(k)}$ computed by the modified Picard scheme converges to the (unique) solution of the steady Navier-Stokes problem if the initial guess is close enough to the exact solution. For $\alpha \to 1$, the convergence holds unconditionally, regardless the initial guess and the assumption* (3.4) *can be relaxed to* (3.3)*.*

PROOF

The proof extends the one of [13], that we follow closely. Let[3] $\boldsymbol{e}^{(k)} \equiv \boldsymbol{u}_h{}^{(k)} - \boldsymbol{u}_h$ and $\varepsilon^{(k)} \equiv p_h{}^{(k)} - p_h$ be the errors associated with the modified Picard scheme at iteration k.

By direct subtraction of the (weak) Navier-Stokes problem from the modified Picard iteration we obtain the error equation, for any $\boldsymbol{v}_h \in V_h, q_h \in Q_h$:

$$\begin{aligned} a(\boldsymbol{e}^{(k+1)}, \boldsymbol{v}_h) + b(\varepsilon^{(k+1)}, \boldsymbol{v}_h) - b(q_h, \boldsymbol{e}^{(k+1)}) = -\alpha \left(c(\boldsymbol{u}_h{}^{(k)}, \boldsymbol{u}_h{}^{(k+1)}, \boldsymbol{v}_h) - c(\boldsymbol{u}_h, \boldsymbol{u}_h, \boldsymbol{v}_h) \right) \\ -(1-\alpha) \left(c(\boldsymbol{u}_h{}^{(k)}, \boldsymbol{u}_h{}^{(k)}, \boldsymbol{v}_h) - c(\boldsymbol{u}_h, \boldsymbol{u}_h, \boldsymbol{v}_h) \right). \end{aligned} \tag{3.5}$$

In particular, we select $\boldsymbol{v}_h = \boldsymbol{e}^{(k+1)}$ and $q_h = \varepsilon^{(k+1)}$ so that the the left hand side reduces to $a(\boldsymbol{e}^{(k+1)}, \boldsymbol{e}^{(k+1)})$. Let us focus on the two terms within parentheses on the right hand side.

$$\left(c(\boldsymbol{u}_h{}^{(k)}, \boldsymbol{u}_h{}^{(k+1)}, \boldsymbol{e}^{(k+1)}) - c(\boldsymbol{u}_h, \boldsymbol{u}_h, \boldsymbol{e}^{(k+1)}) \right) = \left(c(\boldsymbol{u}_h{}^{(k)}, \boldsymbol{e}^{(k+1)}, \boldsymbol{e}^{(k+1)}) + c(\boldsymbol{e}^{(k)}, \boldsymbol{u}_h, \boldsymbol{e}^{(k+1)}) \right) =$$

$$c(\boldsymbol{e}^{(k)}, \boldsymbol{u}_h, \boldsymbol{e}^{(k+1)}).$$

Also,

$$\left(c(\boldsymbol{u}_h{}^{(k)}, \boldsymbol{u}_h{}^{(k)}, \boldsymbol{e}^{(k+1)}) - c(\boldsymbol{u}_h, \boldsymbol{u}_h, \boldsymbol{e}^{(k+1)}) \right) = \left(c(\boldsymbol{u}_h{}^{(k)}, \boldsymbol{e}^{(k)}, \boldsymbol{e}^{(k+1)}) + c(\boldsymbol{e}^{(k)}, \boldsymbol{u}_h, \boldsymbol{e}^{(k+1)}) \right) =$$

$$\left(c(\boldsymbol{e}^{(k)}, \boldsymbol{e}^{(k)}, \boldsymbol{e}^{(k+1)}) + c(\boldsymbol{u}_h, \boldsymbol{e}^{(k)}, \boldsymbol{e}^{(k+1)}) + c(\boldsymbol{e}^{(k)}, \boldsymbol{u}_h, \boldsymbol{e}^{(k+1)}) \right)$$

By collecting all these results, for the coercivity of the bilnear form $a(\cdot,\cdot)$ we obtain

$$\begin{aligned} \nu C_a \|\boldsymbol{e}^{(k+1)}\|_V^2 \leq C_b \left(\|\boldsymbol{u}_h\|_V \|\boldsymbol{e}^{(k)}\|_V \|\boldsymbol{e}^{(k+1)}\|_V + (1-\alpha)\|\boldsymbol{u}_h\|_V \|\boldsymbol{e}^{(k)}\|_V \|\boldsymbol{e}^{(k+1)}\|_V \right. \\ \left. +(1-\alpha)\|\boldsymbol{e}^{(k)}\|_V^2 \|\boldsymbol{e}^{(k+1)}\|_V \right) \end{aligned} \tag{3.6}$$

After trivial manipulation, and recalling (3.1) we find

$$\|\boldsymbol{e}^{(k+1)}\|_V \leq (2-\alpha) \frac{C_b \|\boldsymbol{f}\|_{-1}}{\nu^2 C_a^2} \|\boldsymbol{e}^{(k)}\|_V + \frac{(1-\alpha) C_b}{\nu C_a} \|\boldsymbol{e}^{(k)}\|_V^2. \tag{3.7}$$

if we set $\rho \equiv \|\boldsymbol{e}^{(k)}\|_V$ and $\sigma \equiv \|\boldsymbol{e}^{(k+1)}\|_V$, we have the inequality

$$\chi \rho^2 + \xi \rho > \sigma$$

with

$$\chi = \frac{(1-\alpha) C_b}{\nu C_a}, \quad \xi = (2-\alpha) \frac{C_b \|\boldsymbol{f}\|_{-1}}{\nu^2 C_a^2} = \frac{2-\alpha}{\bar{\delta}}.$$

Now, if $\rho > \chi\rho^2 + \xi\rho$ we have that $\sigma = \beta\rho$ with $0 < \beta < 1$ proving that the modified Picard method is convergent with dumping factor β. For $\alpha = 1$ this is trivially true, as expected by the original Picard scheme, since we find $\chi = 0, \xi = \bar{\delta}^{-1} < 1$.

[3] We do not add the subscript h to the error symbols. Even if appropriate, we feel it makes the notation much heavier.

In general, since $\alpha \in [0,1]$, the inequality $\rho > \chi\rho^2 + \xi\rho$ leads to

$$\rho < \frac{1-\xi}{\chi} = \frac{\nu C_a}{(1-\alpha)C_b}\frac{\delta - \bar{2} + \alpha}{\bar{\delta}}.$$

Notice that for $\alpha \in [0,1]$ under the assumption (3.4), the right hand side is positive and the bound can be fulfilled.

If we assume that the velocity initial guess is such that $\|\boldsymbol{u}_h{}^{(0)}\|_V < \frac{\nu C_a}{(1-\alpha)C_b}\frac{\bar{\delta}-2}{\bar{\delta}}$, then by induction the entire sequence is convergent. Notice, however, that for $\alpha \to 1$ this restriction relaxes, and (3.3) suffices for the right hand side to be positive and the the unconditional convergence of the original Picard method is recovered.

We finally notice that the assumption (3.4) is sufficient for the Theorem. It can be replaced by an assumption on α close enough to 1, precisely $\alpha > 2 - \bar{\delta}$ (for $\bar{\delta} < 2$).

Remark

The previous analysis suggests a possible practical approach to obtain a good initial guess. In practice, we can initiate the iterative process with $\alpha = 1$ (original Picard) and eventually switch to $\alpha < 1$ so as to apply the splitting scheme. This approach is robust, even if it requires one to solve the problem with a mesh fine enough to stand the physical Reynolds number. In a previous Remark, we suggested using α to mitigate the mesh requirements for a given (moderate) Reynolds number. A smart dynamic fine tuning of α can clearly attain the purpose of both robustness and efficiency. This dynamic strategy will be investigated elsewhere. For the problems presented herafter, we did not find any problem for the initial guess, so we opted for "small" constant values of α to guarantee computational efficiency.

4. Analysis of the splitting method

A complete convergence analysis of the splitting scheme is not the purpose of the present work, and it requires a deep analysis of the error equations governing the behavior of the difference between the solution of the modified Picard method and the segregated solution. A similar analysis for the unsteady splitting Yosida scheme was carried out in [8]. Here, we limit ourselves to the consistency analysis, showing that the splitting error locally (i.e. at each modified Picard iteration) vanishes with α. Then, we discuss the empirical selection of the relaxation parameter related to boundness of the generic guess $\boldsymbol{u}_{h,s}^{(k)}$. Here s denotes the solution of the split scheme as opposed to u that we use for the unsplit modified Picard iterate.

We resort to algebraic arguments. To this aim, we will extensively use the following formula, for two generic invertible matrices R and T,

$$R^{-1} - T^{-1} = -R^{-1}(R-T)T^{-1}. \tag{4.1}$$

4.1. Consistency

As mentioned earlier, the splitting scheme is an inexact block LU factorization of the original saddle point system. Notice that the (1,2) block of the splitting matrix reads

$$A_\alpha K^{-1} B^T = B^T + \alpha C K^{-1} B^T.$$

The consistency of the splitting error block with the unsplit method for $\alpha \to 0$ follows promptly as K, C and B are independent of α.

For the sake of notation, we denote

$$S \equiv B K^{-1} B^T \qquad \text{and} \quad \Sigma_\alpha \equiv B A_\alpha^{-1} B^T.$$

By direct inspection we find that

$$\mathcal{A}_{\alpha,u}^{-1} = \begin{bmatrix} (I - A_\alpha^{-1} B^T \Sigma_\alpha^{-1} B) A_\alpha^{-1} & A_\alpha^{-1} B^T \Sigma_\alpha^{-1} \\ \\ \Sigma_\alpha^{-1} B A_\alpha^{-1} & -\Sigma_\alpha^{-1} \end{bmatrix}, \quad \mathcal{A}_{\alpha,s}^{-1} = \begin{bmatrix} (I - K^{-1} B^T S^{-1} B) A_\alpha^{-1} & K^{-1} B^T S^{-1} \\ \\ S^{-1} B A_\alpha^{-1} & -S^{-1} \end{bmatrix}.$$

To analyze the error introduced at each iteration, let us assume to perform one iteration of the unsplit as well as of the split method starting from the same guess $\boldsymbol{u}_h^{(k)}, p_h^{(k)}$. Denoting

$$\boldsymbol{f}_\alpha \equiv \boldsymbol{f} - (1-\alpha) C \boldsymbol{u}_h^{(k)},$$

we have then the two systems

$$\mathcal{A}_{\alpha,u} \begin{bmatrix} {\boldsymbol{u}_{h,u}}^{(k+1)} \\ \\ {p_{h,u}}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_\alpha \\ \\ 0 \end{bmatrix}, \qquad \mathcal{A}_{\alpha,s} \begin{bmatrix} {\boldsymbol{u}_{h,s}}^{(k+1)} \\ \\ {p_{h,s}}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_\alpha \\ \\ 0 \end{bmatrix}.$$

Let ${\boldsymbol{e}_s}^{(k+1)} \equiv {\boldsymbol{u}_{h,u}}^{(k+1)} - {\boldsymbol{u}_{h,s}}^{(k+1)}$ and ${\varepsilon_s}^{(k+1)} \equiv {p_{h,u}}^{(k+1)} - {p_{h,s}}^{(k+1)}$ denote the splitting error for velocity and pressure respectively. By direct computation, the local splitting error reads

$$\begin{bmatrix} {\boldsymbol{e}_s}^{(k+1)} \\ \\ {\varepsilon_s}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \left(K^{-1} B^T S^{-1} B A_\alpha^{-1} - A_\alpha^{-1} B^T \Sigma_\alpha^{-1} B A_\alpha^{-1}\right) \boldsymbol{f}_\alpha \\ \\ \left(S^{-1} - \Sigma_\alpha^{-1}\right) B A_\alpha^{-1} \boldsymbol{f}_\alpha \end{bmatrix}. \tag{4.2}$$

From (4.1), we obtain

$$K^{-1} - A_\alpha^{-1} = \alpha K^{-1} C A_\alpha^{-1},$$

$$S^{-1} - \Sigma_\alpha^{-1} = -S^{-1} \left(S - \Sigma_\alpha\right) \Sigma_\alpha^{-1} = -S^{-1} B \left(K^{-1} - A_\alpha^{-1}\right) B^T \Sigma_\alpha^{-1} = -\alpha S^{-1} B K^{-1} C B^T \Sigma_\alpha^{-1}$$

Also, set

$$Z_\alpha = K^{-1} C A_\alpha^{-1} B^T \Sigma_\alpha^{-1} B A_\alpha^{-1}$$

and since K is s.p.d. we obtain

$$K^{-1}B^TS^{-1}BA_\alpha^{-1} - A_\alpha^{-1}B^T\Sigma_\alpha^{-1}BA_\alpha^{-1} = K^{-1}B^T\left(S^{-1} - \Sigma_\alpha^{-1}\right)BA_\alpha^{-1} + \left(K^{-1} - A_\alpha^{-1}\right)B^T\Sigma_\alpha^{-1}BA_\alpha^{-1} =$$

$$\alpha\left(I - K^{-1}B^TS^{-1}B\right)Z_\alpha = \alpha K^{-1/2}\left(I - K^{-1/2}B^TS^{-1}BK^{-1/2}\right)K^{1/2}Z_\alpha.$$

After introducing the QR factorization of the matrix $K^{-1/2}B^T$ by standard arguments the previous matrix reads

$$\alpha K^{-1/2}Q^T\begin{bmatrix}0 & 0\\ & \\ 0 & I_{N_p}\end{bmatrix}QK^{1/2}Z_\alpha.$$

For the assumption on the Reynolds number, we postulate that the matrix A_α is invertible for any value of $\alpha \in [0,1]$ and its spectrum depends continuously on α and it is bounded accordingly. The same holds for Σ_α and Z_α. Consequently,

$$\begin{bmatrix}\boldsymbol{e}_s{}^{(k+1)}\\ \\ \varepsilon_s{}^{(k+1)}\end{bmatrix} = \alpha\begin{bmatrix}WZ_\alpha\boldsymbol{f}_\alpha\\ \\ -S^{-1}BK^{-1}CB^T\Sigma_\alpha^{-1}BA_\alpha^{-1}\boldsymbol{f}_\alpha\end{bmatrix} \tag{4.3}$$

where

$$W \equiv I - K^{-1}B^TS^{-1}B = K^{-1/2}Q^T\begin{bmatrix}0 & 0\\ & \\ 0 & I_{N_p}\end{bmatrix}QK^{1/2}$$

has N_u null eigenvalues and N_p eigenvalues $= 1$. This points out that the splitting error reads $\alpha\boldsymbol{w}_\alpha$ where the vector $\boldsymbol{w}_\alpha$ has a norm bounded with $\alpha \in [0,1]$. This proves the local consistency of the splitting error at each with $\alpha \to 0$.

4.2. Boundedness and empirical selection of the relaxation parameter

As proven in the analysis of the modified Picard scheme, the introduction of an explicit component in the iteration can in general cause some stability issues which can be mitigated through the introduction of the relaxation step with parameter γ. While, in practice, the selection of an appropriate parameter can require some trial and error, we nonetheless can demonstrate some heuristic guidelines for its selection. Here we consider the splitting method applied to a simplified problem with no forcing term, as for any stability analysis, so that $\boldsymbol{f}_\alpha$ reduces to $(\alpha-1)C\boldsymbol{u}^k$ (we keep working on the non incremental scheme).

By direct inspection, the unrelaxed velocity, hereafter denoted by $\hat{\boldsymbol{u}}^{(k+1)}$ computed by the splitting scheme reads

$$\hat{\boldsymbol{u}}^{(k+1)} = (A_\alpha^{-1} - K^{-1}B^TS^{-1}BA_\alpha^{-1})\boldsymbol{f}_\alpha = (1-\alpha)WA_\alpha^{-1}C\boldsymbol{u}^{(k)}.$$

Notice that from (4.1), we have

$$A_\alpha^{-1} = K^{-1} - \alpha K^{-1} C A_\alpha^{-1}$$

so that

$$(1-\alpha) W A_\alpha^{-1} C = (1-\alpha) W K^{-1} C \left(I - \alpha A_\alpha^{-1} C\right) = (1-\alpha) W K^{-1} C \left(I - \alpha K^{-1} C\right) + \mathcal{O}(\alpha^2)$$

Hereafter, we approximate the matrix up to terms of order α,

$$(1-\alpha) W A_\alpha^{-1} C \approx (1-\alpha) W K^{-1} C.$$

Fix $\epsilon > 0$. Since $\rho(W) = 1$ as noted in the previous Section, there must exist a matrix norm $\|\cdot\|$ such that $\|W\| < (1+\epsilon)$.

Now, for the relaxed velocity, we have

$$\boldsymbol{u}^{(k+1)} = \gamma \hat{\boldsymbol{u}}^{(k+1)} + (1-\gamma)\boldsymbol{u}^{(k)} = \left[(1-\alpha)\gamma W K^{-1} C + (1-\gamma)\right] \boldsymbol{u}^{(k)}$$

so that for the norm introduced before and assuming that $\alpha \|K^{-1} C\| < 1$

$$\|\boldsymbol{u}^{(k+1)}\| \leq \left(\frac{(1-\alpha)}{\alpha} \gamma (1+\epsilon) + (1-\gamma) \right) \|\boldsymbol{u}^{(k)}\|.$$

Now,

$$\left| \frac{(1-\alpha)}{\alpha} \gamma (1+\epsilon) + (1-\gamma) \right| \leq 1$$

leads to

$$\gamma \leq 2\alpha + \mathcal{O}(\epsilon)$$

Therefore, we empirically follow the bound

$$\gamma < 2\alpha$$

in Sect. 5, as this resulted to be effective.

5. Numerical results

5.1. Analytical Solution Test

We devise a test problem with a known analytic solution to verify the convergence of the method. We consider problem (2.1) defined on $\Omega = [0,1]^2$ with $\nu = .01$. We define our forcing term as:

$$\boldsymbol{f} = [-2\nu \sin(x)\cos(y) + \frac{1}{2}\sin(2x) + \cos(x), -2\nu \cos(x)\sin(y) + \frac{1}{2}\sin(2y) + \cos(y)]',$$

with the corresponding solution

$$\boldsymbol{u}_{ex} = [-\sin(x)\cos(y), \cos(x)\sin(y)]'; \boldsymbol{p}_{ex} = \sin(x) + \sin(y)$$

We simulated the problem with both a coarse (800 elements) and a fine (5000 elements) mesh for several different values of α and γ. As convergence criterion we compared the L^2 norm of velocity between each iteration for tolerance levels of 1e-3, 1e-4, and 1e-5. These computations were performed in FreeFem++ on a 2013 MacBook Pro. We solved the linear systems with UMFPACK, except for the Schur Complement step which was solved with PCG (using the simple Stokes pressure mass preconditioner, see e.g. [29]) with a stopping tolerance of 1e-6. The results are reported in Tab. 1 and 2.

Coarse Mesh, 800 Elements					
Parameters	Tolerance	$\|\boldsymbol{u}-\boldsymbol{u}_{ex}\|_{H1}$	$\|\boldsymbol{u}-\boldsymbol{u}_{ex}\|_{L2}$	$\|\boldsymbol{p}-\boldsymbol{p}_{ex}\|_{L2}$	Number of Iterations
$\alpha = .175$, $\gamma = .3$	1e-3	.0313	.00197	.00185	19
$\alpha = .175$, $\gamma = .3$	1e-4	.00325	.000161	.000345	25
$\alpha = .175$, $\gamma = .3$	1e-5	.00072	1.86e-5	.00022	32
$\alpha = .2$, $\gamma = .35$	1e-3	.02061	.00129	.00128	17
$\alpha = .2$, $\gamma = .35$	1e-4	.00210	7.75e-5	.000253	23
$\alpha = .2$, $\gamma = .35$	1e-5	.000671	1.57e-5	.000206	32
$\alpha = .25$, $\gamma = .4$	1e-3	.01259	.00179	.00207	18
$\alpha = .25$, $\gamma = .4$	1e-4	.00158	.00021	.000371	31
$\alpha = .25$, $\gamma = .4$	1e-5	.000674	1.748e-5	.000212	47

Table 1: Performance of the segregated scheme on the Analytical Solution Test. Coarse Mesh.

Fine Mesh, 5000 Elements					
Parameters	Tolerance	$\|\boldsymbol{u}-\boldsymbol{u}_{ex}\|_{H1}$	$\|\boldsymbol{u}-\boldsymbol{u}_{ex}\|_{L2}$	$\|\boldsymbol{p}-\boldsymbol{p}_{ex}\|_{L2}$	Number of Iterations
$\alpha = .175$, $\gamma = .3$	1e-3	.0313	.00197	.00169	18
$\alpha = .175$, $\gamma = .3$	1e-4	.00320	.000163	.000174	25
$\alpha = .175$, $\gamma = .3$	1e-5	.00033	1.61e-5	4.46e-5	32
$\alpha = .2$, $\gamma = .35$	1e-3	02055	.00127	.00114	17
$\alpha = .2$, $\gamma = .35$	1e-4	00202	7.28e-5	.000109	23
$\alpha = .2$, $\gamma = .35$	1e-5	000153	8.16e-6	3.32e-5	32
$\alpha = .25$, $\gamma = .4$	1e-3	.01251	.00179	.00204	18
$\alpha = .25$, $\gamma = .4$	1e-4	.00144	.000211	.000264	31
$\alpha = .25$, $\gamma = .4$	1e-5	.000171	1.40e-5	4.04e-5	47

Table 2: Performance of the segregated scheme on the Analytical Solution Test. Fine mesh.

These results confirm the convergence to the desired solution for appropriate values of α and γ and that convergence rate is independent of mesh size. Convergence was most rapid for $\alpha = .2$ and $\gamma = .35$.

The stabilizing effect of α is evident in the coarse mesh case. In fact, for $\alpha = 1$ we have $h_{coarse} \approx 0.03$

and a Péclet number for the corresponding problem ($\alpha = 1$) of

$$\mathbb{P} = \frac{\|\boldsymbol{u}\|_\infty h}{2\nu} \approx \frac{\sqrt{2} \times 0.03}{2 \times 0.01} \approx 2.$$

With the selection of $\alpha < 0.5$ the associated Péclet number is < 1 and any convective instability is avoided even for the coarse mesh.

We also observe that, as expected, refining the mesh results in better convergence to the exact solution, particularly for the pressure. This confirms that the splitting error does not dominate the discretization error after refining the mesh, despite little change in the convergence rate.

5.2. Schaeffer/Turek Flow Past a Cylinder

5.2.1. Problem Setting

We now test the method on the classic benchmark problem of flow past a cylinder in both the 2D and 3D settings. We model our problem setting as the one given in [30]. We consider the domain reported in Fig. 2.

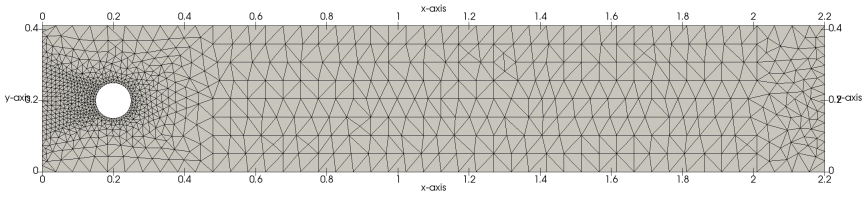

Figure 1: Domain for the Shaeffer/Turek test case.

We prescribe at the inflow the Dirichlet boundary condition:

$$\begin{aligned} \boldsymbol{u}(0, y) &= \frac{1}{.41^2}(1.2y(0.41 - y), 0), \quad 0 \leq y \leq .41 \ \text{ in 2D} \\ \boldsymbol{u}(0, y, z) &= \frac{1}{.41^4}(6.56yz(.41 - y)(.41 - z), 0, 0), \quad 0 \leq y, z \leq .41 \ \text{ in 3D} \end{aligned} \tag{5.1}$$

We prescribe no-slip boundary conditions along the cylinder and the top and bottom of the domain and traction-free boundary conditions at the outflow. The benchmark parameters of interest are the lift and drag coefficients at the cylinder c_l and c_d respectively and the difference in pressure between the front

and back of the cylinder defined as:

$$\begin{aligned} \Delta p &= p(.15,.2) - p(.25,.2) \quad \text{in 2D} \\ \Delta p &= p(.45,.2,.205) - p(.55,.2,.205) \quad \text{in 3D} \end{aligned} \tag{5.2}$$

The drag and lift coefficients c_d and c_l are defined classically as

$$c_d = \frac{2}{U^2 L}\int_{\Gamma_{cyl}} \nu \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} \boldsymbol{n}_y - p\boldsymbol{n}_x \, \partial\Gamma_{cyl}, \;\; c_l = -\frac{2}{U^2 L}\int_{\Gamma_{cyl}} \nu \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} \boldsymbol{n}_y + p\boldsymbol{n}_y \, \partial\Gamma_{cyl} \quad \text{in 2D} \tag{5.3}$$

and

$$c_d = \frac{2}{U^2 LH}\int_{\Gamma_{cyl}} \nu \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} \boldsymbol{n}_y - p\boldsymbol{n}_x \, \partial\Gamma_{cyl}, \quad c_l = -\frac{2}{U^2 LH}\int_{\Gamma_{cyl}} \nu \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{n}} \boldsymbol{n}_y + p\boldsymbol{n}_y \, \partial\Gamma_{cyl} \quad \text{in 3D} \tag{5.4}$$

where $\boldsymbol{n}$ is the unit normal vector and U, L and H are the characteristic velocity, length and height respectively. Here the kinematic viscosity ν is set to .001. As shown in [31], the computation of c_d and c_l based on this formulation may yield inaccurate numerical results. Accordingly, we employ a technique used in [32] to rewrite (5.3) as a volume integral. Define $\boldsymbol{v}_d$ as $(\boldsymbol{v}_d)|_{\Gamma_{cyl}} = (1,0)^T$ and vanishing at all other boundaries. Following the argument given in [32] we see that for two dimensions we obtain the alternate formulation

$$c_d = -500\int_\Omega \nu \nabla \boldsymbol{u} : \nabla \boldsymbol{v}_d + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\cdot\boldsymbol{v}_d - p(\nabla\cdot\boldsymbol{v}_d)\,\partial\Omega \tag{5.5}$$

For three dimensions, we define $\boldsymbol{v}_d$ as $(\boldsymbol{v}_d)|_{\Gamma_{cyl}} = (1,0,0)^T$, vanishing at all other boundaries and obtain:

$$c_d = -\frac{500}{.41}\int_\Omega \nu \nabla \boldsymbol{u} : \nabla \boldsymbol{v}_d + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\cdot\boldsymbol{v}_d - p(\nabla\cdot\boldsymbol{v}_d)\,\partial\Omega \tag{5.6}$$

Similarly, defining $\boldsymbol{v}_l$ as $(\boldsymbol{v}_l)|_{\Gamma_{cyl}} = (0,1)^T$ and vanishing at all other boundaries for two dimensions we find:

$$c_l = -500\int_\Omega \nu \nabla \boldsymbol{u} : \nabla \boldsymbol{v}_l + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\cdot\boldsymbol{v}_l - p(\nabla\cdot\boldsymbol{v}_l)\,\partial\Omega \tag{5.7}$$

For three dimensions, let $\boldsymbol{v}_l$ as $(\boldsymbol{v}_l)|_{\Gamma_{cyl}} = (0,1,0)^T$ and vanishing at all other boundaries:

$$c_l = -\frac{500}{.41}\int_\Omega \nu \nabla \boldsymbol{u} : \nabla \boldsymbol{v}_l + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u}\cdot\boldsymbol{v}_l - p(\nabla\cdot\boldsymbol{v}_l)\,\partial\Omega \tag{5.8}$$

For our finite element discretization, we use standard Taylor-Hood P2/P1 for velocity and pressure. We test convergence by comparing the $L2$ norm of the current iteration with the previous iteration and stop iterating at a tolerance level of 1e-3. For accuracy, we compare our values to those computed in [33], which were computed by high-order spectral element methods and are given for the 2D case by:

$$c_d = 5.57953523384, \quad c_l = 0.010618948146, \quad \Delta p = 0.11752016697 \tag{5.9}$$

and for the 3D case:

$$c_d = 6.1853267, \quad c_l = .0094009839, \quad \Delta p = 0.1708 \tag{5.10}$$

Coarse Mesh, 831 Elements				
Parameters	c_d/c_d Error	c_l/c_l Error	$\Delta p/\Delta p$ Error	Number of Iterations
$\alpha = .175$, $\gamma = .3$	5.5599/.00350	.010422/.0187	.11666/.00731	26
$\alpha = .2$, $\gamma = .35$	5.5606/.00339	.01008/.0500	.11666/.00726	23
$\alpha = .25$, $\gamma = .4$	5.5632/.00291	.00970/.0862	.116714/.00685	31

Moderate Mesh, 1257 Elements				
Parameters	c_d/c_d Error	c_l/c_l Error	$\Delta p/\Delta p$ Error	Number of Iterations
$\alpha = .175$, $\gamma = .3$	5.5657/.00247	.01148/.0813	.11573/.01518	26
$\alpha = .2$, $\gamma = .35$	5.5663/.00236	.01164/.05132	.115741/.01513	23
$\alpha = .25$, $\gamma = .4$	5.5700/.00169	.01106/.04211	.11579/.014662	30

Fine Mesh, 3365 Elements				
Parameters	c_d/c_d Error	c_l/c_l Error	$\Delta p/\Delta p$ Error	Number of Iterations
$\alpha = .175$, $\gamma = .3$	5.5737/.00103	.01089/.02529	.11745/.00061	26
$\alpha = .2$, $\gamma = .35$	5.5742/.00093	.01056/.00512	.117454/.0006	23
$\alpha = .25$, $\gamma = .4$	5.5781/.00025	.01036/.0239	.11751/7.3e-05	30

Table 3: Numerical Results on the 2D Schaefer/Turek Flow Past a Cylinder Test Case.

5.2.2. 2D Results

For this problem our experimental design was as follows: we considered three different meshes: coarse, moderate, and fine. We also looked at the accuracy of our segregated scheme for differing values of α and γ. For each mesh size we considered $\alpha = .175$ and $\gamma = .3$, $\alpha = .2$ and $\gamma = .35$ and $\alpha = .25$ and $\gamma = .4$. We also solve for each case an "exact" version of the problem that does not use any approximations or relaxation. These computations were performed in FreeFem++ on a 2013 MacBook Pro. We solved the linear systems with UMFPACK, except for the Schur Complement step which was solved with PCG (using the simple Stokes pressure mass preconditioner, see e.g. [29]) with a stopping criterion of 1e-6. To check the convergence, we computed the difference in L^2 of the current and previous velocity at each step and terminated the iteration at for a value below 1e-4. In Tab. 3 we report the accuracy for each parameter of interest, number of iterations required to reach convergence, and the average time of each iteration.

The lift coefficient c_l is sensitive to α and γ, more than to the fineness of the mesh. Values $\alpha = .2$ and $\gamma = .4$ gave generally the most accurate result as well as the fastest convergence. $\alpha = .25$ gave less accurate results for all of the parameters of interest. This is somehow expected, as our splitting error vanishes for $\alpha \to 0$. Overall, this test showcases our method ability to capture numerically delicate quantities efficiently and without a significant loss of accuracy.

5.3. Numerical Tests: 3D

In three dimensions we tested the algorithm on a moderate mesh with 7973 elements for the same values of α and γ. As the systems became too large for UMFPACK, we used GMRES (stopping criterion of 1e-6) except for the Schur Complement, where we used PCG with the M_p preconditioner for the approximate Schur complement S (again with a stopping criterion of 1e-6). We also solved the problem using the exact factorized solver for the purposes of comparison. In this case we solved the full Schur Complement using GMRES (stopping criterion 1e-6) using the well known Caouet Chabard preconditioner $\nu M_p^{-1} + \frac{1}{\nu} K_p^{-1}$. We computed the difference in L^2 of the current and previous velocity at each step and terminated the iteration for a value below 1e-3. We report the results in Tab. 4

3D Mesh, 7973 Elements						
Parameters	c_d/c_d Error	c_l/c_l Error	$\Delta p/\Delta p$ Error	Iterations	Avg time/Iter	Total time
Unsplit	6.1481/.00602	.00998/.0618	.169313/.00871	7	831.91s	5823.4s
$\alpha = .175$, $\gamma = .3$	6.1398/.00735	.00930/.01041	.1688/.01171	19	153.03s	2907.6s
$\alpha = .2$, $\gamma = .35$	6.1391/.00773	.00982/.0442	.16883/.01149	17	157.2s	2672.4s
$\alpha = .25$, $\gamma = .4$	6.1273/.00938	.01058/.12564	.16876/.01214	15	166.58s	2498.7s

Table 4: Numerical Results on the 3D Schaefer/Turek Flow Past a Cylinder Test Case.

We clearly see a marked advantage of our method vs the exact factorization, as for all levels of α and γ the method converged in roughly one half the overall time as compared to the solving the full system, with no major loss in accuracy. Although our method requires substantially more iterations, the savings at each iteration are so significant that our method drastically outperforms the exact approach.

5.4. Backward Facing Step

In this section we demonstrate the competitiveness of our method by solving the steady Navier-Stokes Equations in a backward-facing step geometry following the approach given in [34]. We consider the problem at Reynolds numbers of 200 and 400 and demonstrate that our method results in considerable savings in computational time compared to solving the full problem.

For our problem domain, we have an inlet height of 1 and a step height of .9423 in accordance with the setup in [34]. We extend the flow channel to $50h$ in order to allow the flow to develop as much as possible. We prescribe a parabolic inflow velocity at the inlet, homogenous Neumann boundary conditions at the outlet and no-slip conditions at each wall. To reduce the impact of the mesh on convergence, we use a uniformly meshed domain. We compute the solution on two meshes: a moderate mesh with 3292 elements and a fine mesh with 5073 elements. For the Re=200 case we set $\alpha = .165$ and $\gamma = .35$, and for the Re=400 case we used $\alpha = .1$ and $\gamma = .165$. These values were chosen somewhat arbitrarily. We report the results in Tab. 5.

Backward Facing Step, Re=200								
Mesh	Splitting solution			Unsplit solution			Splitting Error	
Elements	Iterations	Time/Iter	Total Time	Iterations	Time/Iter	Total Time	$\|\boldsymbol{u}\|_{H1}$	$\|p\|_{L2}$
3292	36	1.819s	65.48s	14	5.85s	81.9s	.00472	.00057
5073	32	2.99s	95.68s	14	12.019s	168.26s	.00419	.00042

Backward Facing Step, Re=400								
Mesh	Splitting solution			Unsplit solution			Splitting Error	
Elements	Iterations	Time/Iter	Total Time	Iterations	Time/Iter	Total Time	$\|\boldsymbol{u}\|_{H1}$	$\|p\|_{L2}$
3292	82	1.839s	150.798s	30	6.21s	186.3s	.006859	.000217
5073	95	2.758s	262.01s	30	9.919s	297.57s	.00703	.00237

Table 5:

We note that for this test the segregated method provides a noticeable computational advantage compared to solving the full system. Although more iterations are required to solve the problem, the cost of each iteration is much cheaper than the cost associated with solving the full problem. This is due to the better spectral properties of our approximated Schur Complement, which does not change and it can be cheaply and effectively preconditioned. Solving the full Schur Complement represents a bottleneck for the standard approach, as poor spectral properties can result in very slow convergence for each solve. The advantage becomes more pronounced for the $Re = 400$ case with the fine mesh case.

5.5. Heywood-Rannacher-Turek Bifurcation

Similarly to the last section, here we solve the problem on a bifurcated domain (as seen in [35]) at Re=400, Re=600 and Re=800 at three different mesh levels to demonstrate the advantages of our method over solving the full system. The geometry is the following:

Along the sides we prescribed no-slip boundary conditions. At the inlet we prescribed a standard parabolic outflow profile and at the outlets homogeneous Neumann conditions. For Re=400, we used $\alpha = .135$, $\gamma = .285$, for Re=600 we used $\alpha = .105$, $\gamma = .215$ and for Re=800 we used $\alpha = .075$, $\gamma = .15$. We used a standard Picard style iteration and stopped the iteration when the difference in velocity $L2$ norm compared to the previous iteration was below 1e-3.

We again observe a clear advantage in computational time using inexact factorization with a small splitting error, particularly for fine mesh levels. Additionally, we see that the convergence and error incurred by the inexact factorization method does not change significantly with mesh refinement, except in the case Re=800. This is expected by the entire construction of the method, which is supposed to work efficiently for moderate Reynolds numbers.

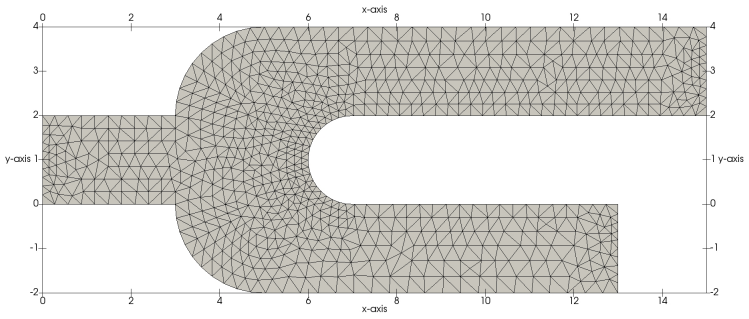

Figure 2: Domain for the Heywood-Rannacher-Turek Bifurcation test case.

Bifurcation, Re=400								
Mesh	Splitting solution			Unsplit solution			Splitting Error	
Elements	Iterations	Time/Iter	Total Time	Iterations	Time/Iter	Total Time	$\|\boldsymbol{u}\|_{H1}$	$\|p\|_{L2}$
1632	24	.741s	17.78s	12	1.642s	19.7s	.00552	.01353
3791	24	1.798s	43.15s	12	4.219s	50.62s	.00526	.01353
9739	24	3.21s	77.04s	12	9.076	108.91s	.00542	.01383

Bifurcation, Re=600								
Mesh	Splitting solution			Unsplit solution			Splitting Error	
Elements	Iterations	Time/Iter	Total Time	Iterations	Time/Iter	Total Time	$\|\boldsymbol{u}\|_{H1}$	$\|p\|_{L2}$
1632	37	.766s	28.34s	17	2.09s	35.53s	.00602	.01021
3791	41	1.76s	72.16s	17	5.03s	85.51s	.0067	.01020
9739	41	3.463s	141.98s	20	12.063	241.26s	.00698	.01023

Bifurcation, Re=800								
Mesh	Splitting solution			Unsplit solution			Splitting Error	
Elements	Iterations	Time/Iter	Total Time	Iterations	Time/Iter	Total Time	$\|\boldsymbol{u}\|_{H1}$	$\|p\|_{L2}$
1632	57	.7048s	40.174s	20	2.746s	54.92s	.00935	.01352
3791	64	1.85s	118.4s	20	6.76s	135.2s	.00752	.00857
9739	66	3.479s	229.61s	20	16.28s	325.6s	.00718	.00766

Table 6: Results for the Heywood-Rannacher-Turek 2D bifurcation.

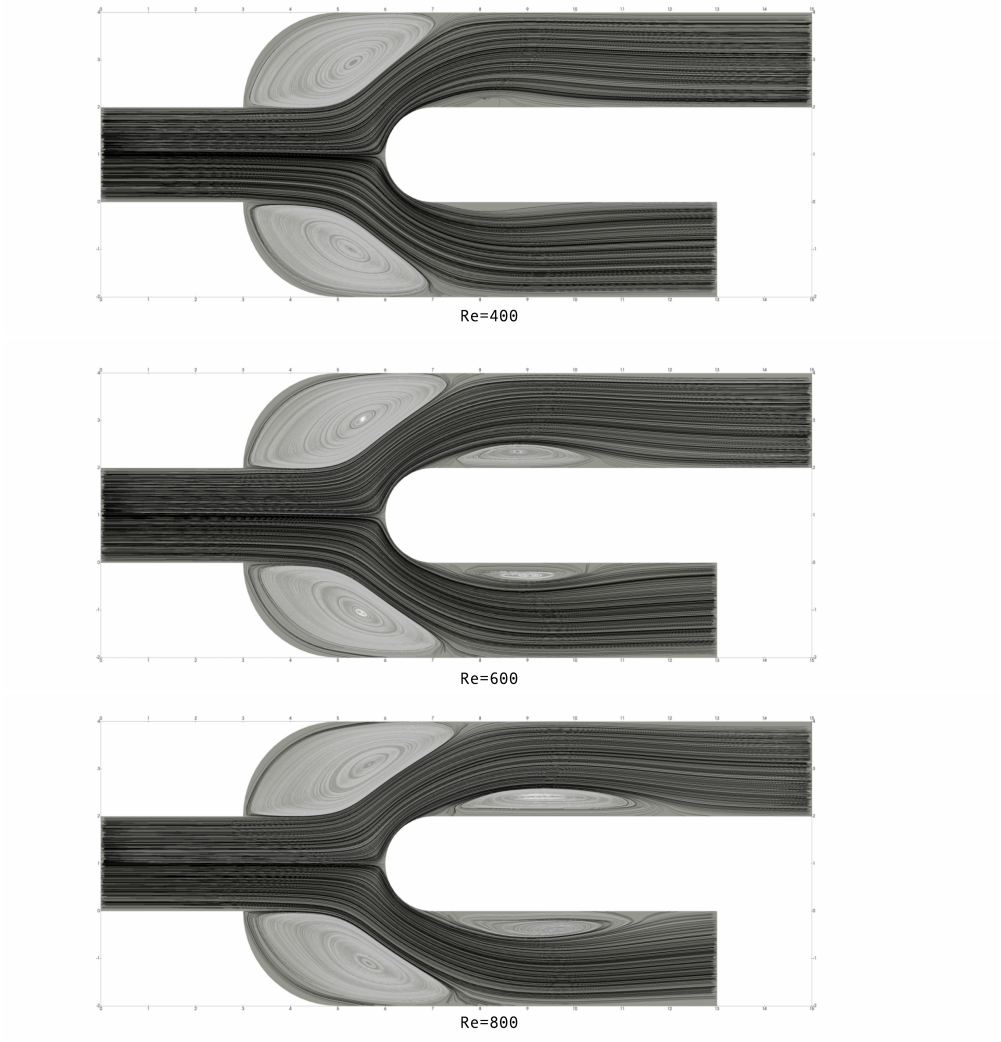

Figure 3: Flow in a bifurcation for different Reynolds numbers.

6. Conclusions

The efficient solution of the steady incompressible Navier-Stokes equations is emerging as a critical step also in many applications traditionally categorized as unsteady problems. For instance, in computational hemodynamics the unsteady solution for clinical purposes is typically averaged in time. The approximation of the time averaged unsteady solution with a a steady solve may clearly reduce the computational costs in an extremely time-sensitive framework.

Segregation approaches resorting to a separate computation of velocity and pressure are extremely popular in the unsteady setting, either as an application of decomposition principles (Ladhyzhenskaja theorem aka Helmholtz principle) in the STD paradigm [2, 3], or as the result of an algebraic argument with an inexact block LU factorization of the matrix of the discretized problem [6**?**] in the DTS

approach. The presence of the time derivative provides the natural approximation to the pseudo-differential operator for the pressure in the STD as well as for the pressure Schur complement in the DTS. The extension of this approach to the steady case is therefore not immediate. In this paper, we present a possible approach that works for moderate Reynolds numbers. We first introduce a modification of the Picard linearization approach with a parameter weighting the linearized term. Based on this parameter, it is possible to approximate the steady Schur complement with a matrix that is spectrally equivalent to a pressure mass. With a DTS perspective, we then introduce a segregated approach that resembles the so called "algebraic Chorin-Temam scheme" for unsteady problems. We provide a complete convergence analysis of the modified Picard linearization and a local analysis of the splitting error. By means of an appropriate relaxation step, we can guarantee stability of the computed solution, while the parameter of the modified Picard can play the role of a convective stabilization.

Numerical results on several and diverse test cases show that the method for Reynolds numbers below 500 can actually lead to efficient and accurate solutions, even in prototype implementations. In the follow up of this work, we will implement the method in our Finite Element library LifeV to work on problems of computational coronary hemodynamics (where Reynolds is of the order of 200). We aim at providing a complete splitting error analysis too.

Acknowledgements

The authors would like to acknowledge Simona Perotto, Luca Bertagna, Leo Rebholz, Adrien Lefieux, Nicola Parolini, Dallas Albritton and Steve Pittard for their helpful input and contributions.

[1] J.-M. Zhang, L. Zhong, T. Luo, Y. Huo, S. Y. Tan, A. S. L. Wong, B. Su, M. Wan, X. Zhao, G. S. Kassab, et al., Numerical simulation and clinical implications of stenosis in coronary blood flow, BioMed research international 2014.

[2] A. J. Chorin, A numerical method for solving incompressible viscous flow problems, Journal of computational physics 2 (1) (1967) 12–26.

[3] R. Temam, Sur l'approximation de la solution des équations de navier-stokes par la méthode des pas fractionnaires (ii), Archive for Rational Mechanics and Analysis 33 (5) (1969) 377–385.

[4] R. Rannacher, On chorin's projection method for the incompressible navier-stokes equations, Springer Berlin Heidelberg, Berlin, Heidelberg, 1992, pp. 167–183.

[5] A. Prohl, Projection and quasi-compressibility methods for solving the incompressible Navier-Stokes equations, Springer, 1997.

[6] J. B. Perot, An analysis of the fractional step method, Journal of Computational Physics 108 (1) (1993) 51–58.

[7] A. Quarteroni, F. Saleri, A. Veneziani, Factorization methods for the numerical approximation of navier–stokes equations, Computer methods in applied mechanics and engineering 188 (1) (2000) 505–526.

[8] A. Quarteroni, F. Saleri, A. Veneziani, Analysis of the yosida method for the incompressible navier–stokes equations, Journal de mathématiques pures et appliquées 78 (5) (1999) 473–503.

[9] P. Gervasio, F. Saleri, A. Veneziani, Algebraic fractional-step schemes with spectral methods for the incompressible navier–stokes equations, Journal of Computational Physics 214 (1) (2006) 347–365.

[10] A. Veneziani, Block factorized preconditioners for high-order accurate in time approximation of the navier-stokes equations, Numerical Methods for Partial Differential Equations 19 (4) (2003) 487–510.

[11] A. Veneziani, U. Villa, Aladins: An algebraic splitting time adaptive solver for the incompressible navier–stokes equations, Journal of Computational Physics 238 (2013) 359–375.

[12] V. Girault, P.-A. Raviart, Finite element methods for Navier-Stokes equations: theory and algorithms, Vol. 5, Springer Science & Business Media, 2012.

[13] O. A. Karakashian, On a galerkin-lagrange multiplier method for the stationary navier-stokes equations, SIAM Journal on Numerical Analysis 19 (5) (1982) 909–923.

[14] A. Quarteroni, A. Valli, Numerical approximation of partial differential equations, Vol. 23, Springer Science & Business Media, 2008.

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

[16] H. C. Elman, D. J. Silvester, A. J. Wathen, Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics, Oxford University Press (UK), 2014.

[17] J.-L. Guermond, L. Quartapelle, On the approximation of the unsteady navier–stokes equations by finite element projection methods, Numerische mathematik 80 (2) (1998) 207–238.

[18] J.-L. Guermond, L. Quartapelle, Calculation of incompressible viscous flows by an unconditionally stable projection fem, Journal of Computational Physics 132 (1) (1997) 12–33.

[19] J. K. Dukowicz, A. S. Dvinsky, Approximate factorization as a high order splitting for the implicit incompressible flow equations, Journal of Computational Physics 102 (2) (1992) 336–347.

[20] D. L. Brown, R. Cortez, M. L. Minion, Accurate projection methods for the incompressible navier–stokes equations, Journal of computational physics 168 (2) (2001) 464–499.

[21] J. Guermond, J. Shen, A new class of truly consistent splitting schemes for incompressible flows, Journal of computational physics 192 (1) (2003) 262–276.

[22] Guermond, Jean-Luc, Un resultat de convergence d'ordre deux en temps pour l'approximation des equations de navier stokes par une technique de projection incrementale, ESAIM: M2AN 33 (1) (1999) 169–189.

[23] L. Quartapelle, Numerical solution of the incompressible Navier-Stokes equations, Vol. 113, Birkhäuser, 2013.

[24] A. Gauthier, F. Saleri, A. Veneziani, A fast preconditioner for the incompressible navier stokes equations, Computing and Visualization in science 6 (2-3) (2004) 105–112.

[25] F. Saleri, A. Veneziani, Pressure correction algebraic splitting methods for the incompressible navier–stokes equations, SIAM journal on numerical analysis 43 (1) (2005) 174–194.

[26] A. Veneziani, A note on the consistency and stability properties of yosida fractional step schemes for the unsteady stokes equations, SIAM Journal on Numerical Analysis 47 (4) (2009) 2838–2843.

[27] W. Couzy, Spectral element discretization of the unsteady navier-stokes equations and its iterative solution on parallel computers.

[28] M. O. Henriksen, J. Holmen, Algebraic splitting for incompressible navier–stokes equations, Journal of Computational Physics 175 (2) (2002) 438–453.

[29] S. Turek, Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approache, Vol. 6, Springer Science & Business Media, 1999.

[30] M. Schäfer, S. Turek, F. Durst, E. Krause, R. Rannacher, Benchmark computations of laminar flow around a cylinder, Springer, 1996.

[31] R. Becker, Weighted error estimators for finite element approximations of the incompressible Navier-Stokes equations, IWR, 1998.

[32] V. John, G. Matthies, Higher-order finite element discretizations in a benchmark problem for incompressible flows, International Journal for Numerical Methods in Fluids 37 (8) (2001) 885–903.

[33] G. Nabh, On high order methods for the stationary incompressible navier-stokes equations.

[34] G. Biswas, M. Breuer, F. Durst, Backward-facing step flows for various expansion ratios at low and moderate reynolds numbers, Journal of fluids engineering 126 (3) (2004) 362–374.

[35] J. G. Heywood, R. Rannacher, S. Turek, Artificial boundaries and flux and pressure conditions for the incompressible navier-stokes equations, International Journal for Numerical Methods in Fluids 22 (5) (1996) 325–352.