MATHEMATICS OF COMPUTATION
https://doi.org/10.1090/mcom/3411
Article electronically published on January 23, 2019

EFFICIENT NONLINEAR ITERATION SCHEMES BASED ON ALGEBRAIC SPLITTING FOR THE INCOMPRESSIBLE NAVIER-STOKES EQUATIONS

LEO G. REBHOLZ, ALEX VIGUERIE, AND MENGYING XIAO

Abstract. We propose new, efficient, and simple nonlinear iteration methods for the incompressible Navier-Stokes equations. The methods are constructed by applying Yosida-type algebraic splitting to the linear systems that arise from grad-div stabilized finite element implementations of incremental Picard and Newton iterations. They are efficient because at each nonlinear iteration, the same symmetric positive definite Schur complement system needs to be solved, which allows for CG to be used for inner and outer solvers, simple preconditioning, and the reusing of preconditioners. For the proposed incremental Picard-Yosida and Newton-Yosida iterations, we prove under small data conditions that the methods converge to the solution of the discrete nonlinear problem. Numerical tests are performed which illustrate the effectiveness of the method on a variety of test problems.

1. Introduction

The development of effective solvers for the steady incompressible Navier-Stokes equations (NSE) is an open problem in the study of incompressible fluids [22]. A successful solver must have two fundamental properties: the nonlinear iteration scheme must converge in a low number of iterations, and the linear systems that arise at each iteration must be efficiently solvable. Newton and Picard iterations typically converge in a low number of nonlinear iterations [14], but create non-symmetric saddle point linear systems that can be difficult to solve. These linear systems resemble closely those that arise in time dependent NSE problems, but are significantly harder because they do not possess the mass matrix contribution of the time derivative in the velocity block, which is critical for the use of splitting methods (either before or after discretization). Other types of nonlinear iteration schemes for steady NSE exist which create easier linear system solves, such as iterated penalty (with small penalty) or Arrow-Hurwicz methods [15, 37], but they may require large numbers of nonlinear iterations to converge. Herein, we propose a modification of Picard- and Newton-type schemes which require linear system solves that are more easily solvable than those that arise for Picard and Newton, and which can have minimal effect on the convergence behavior of the nonlinear iteration.

Received by the editor August 9, 2017, and, in revised form, February 2, 2018, and June 4, 2018.

2010 *Mathematics Subject Classification.* Primary 65N22, 35Q30, 65N30.

The research of the first author was partially supported by NSF grant DMS1522191 and U.S. Army grant 65294-MA.

The research of the fourth author was partially supported by NSF grant DMS1522191.

We consider the steady incompressible NSE, which are given in a domain $\Omega \subset \mathbb{R}^d$ (d=2,3) by

$$u \cdot \nabla u + \nabla p - \nu \Delta u - \gamma \nabla(\nabla \cdot u) = f, \tag{1.1}$$

$$\nabla \cdot u = 0, \tag{1.2}$$

$$u|_{\partial\Omega} = 0, \tag{1.3}$$

where ν is the kinematic viscosity, f is a forcing, u and p represent velocity and pressure, and the parameter $\gamma \geq 0$ is the grad-div stabilization parameter. Although the term $-\gamma\nabla(\nabla \cdot u) = 0$ due to (1.2) in the system above, when discretized with the most common finite element schemes which only weakly enforce (1.2), this term acts to penalize the $L^2(\Omega)$ divergence error of numerical solutions, and reduce the effect of the pressure error on the velocity error [23]. It is shown in numerous recent papers how the use of grad-div stabilization with $\gamma = O(1)$ can dramatically reduce velocity error (see, e.g., [12, 19, 21, 23, 28] and the references therein), hence for the same accuracy, significantly coarser meshes can be used when grad-div stabilization is applied. Grad-div stabilization is also known to aid in preconditioning the Schur complement that arises in the associated linear sytems, although it is a tradeoff since it makes the velocity block solves harder [5, 6, 19]. How to efficiently solve the linear systems with $\gamma > O(1)$ is an open problem, although some work has been done in this direction [36]. Overall, a small grad-div parameter tends to be a good choice that balances these pros and cons, and therefore throughout this work, we will assume $\gamma = O(1)$.

For simplicity of our presentation and analysis, we consider homogeneous Dirichlet boundary conditions, but in our numerical tests we will also use nonhomogeneous Dirichlet and zero-traction boundary conditions. We remark also that all results and analysis herein can be extended, with the same analytic tools, to the case of the NSE with an added friction term of σu to (1.1), with parameter $\sigma > 0$, which can arise from a friction term but more commonly arises from an unsteady NSE temporal discretization, with $\sigma \sim \frac{1}{\Delta t}$ and Δt representing a time step size. The case of interest is σ small, since if σ is large, then practitioners are typically not interested in solving the nonlinear problem (1.1)–(1.3), but instead some linearization of it where $u \cdot \nabla u$ is replaced by $U \cdot \nabla u$, with U a known (e.g., from previous time step solutions) and good approximation to u. We consider herein only the case of $\sigma = 0$ to simplify the presentation and analysis.

The standard Picard iteration scheme for (1.1)–(1.3) takes the following form: Guess u^0, and for $k = 1, 2, \ldots$, find u^k, p^k satisfying

$$u^{k-1} \cdot \nabla u^k + \nabla p^k - \nu \Delta u^k - \gamma \nabla(\nabla \cdot u^k) = f, \tag{1.4}$$

$$\nabla \cdot u^k = 0, \tag{1.5}$$

$$u^k|_{\partial\Omega} = 0. \tag{1.6}$$

After equipping (1.4)–(1.6) with a finite element spatial discretization, one is left to solve a (sometimes very) large linear system at each iteration that takes the form

$$\begin{pmatrix} A_k & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} \hat{u}_k \\ \hat{p}_k \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix}, \tag{1.7}$$

where $\hat{u}_k$ and $\hat{p}_k$ represent coefficient vectors corresponding to functions u_k and p_k, respectively.

Such "saddle point" linear systems for the steady NSE are well known to be difficult to solve, especially when inertial forces from convection are not dominated by the viscous forces, creating an A matrix with large nonsymmetric part. Direct solvers are not effective for these systems, except for tiny problems, and typical approaches use some decomposition of the saddle point matrix. The main difficulty in the decompositions is the need to perform a linear solve using the Schur complement matrix $S := B^T A_k^{-1} B$. The Schur complement matrix is revealed, for example, by performing a block LU decomposition of the above linear system (1.7) arising at step k from (1.4)–(1.6):

$$\begin{pmatrix} A_k & 0 \\ B^T & -B^T A_k^{-1} B \end{pmatrix} \begin{pmatrix} I & A_k^{-1} B \\ 0 & I \end{pmatrix} \begin{pmatrix} \hat{u}_k \\ \hat{p}_k \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix}. \tag{1.8}$$

Solving (1.8) can thus be done by two linear solves with coefficient matrix A_k, and one with the Schur complement matrix.

Solving a Schur complement linear system when the system size is large and A_k has large nonsymmetric part is well known to be very difficult and is an active area of research in CFD. It requires inner and outer iterative solvers for nonsymmetric matrices (e.g., GMRES) since forming A_k^{-1} is not feasible, and also good preconditioners for both solvers. There has been a large amount of recent work on preconditioning these systems, e.g., [4, 5, 10], however, obtaining robust solvers that work for a wide range of problems, parameters, and discretizations remains an important and difficult open problem.

To partially circumvent this issue, work has been done for "two-level" methods for approximating solutions to (1.1)–(1.3), where just one linear solve of (1.7) is required for a fine mesh, and multiple solves of (1.7) for a coarser mesh [17, 24, 26, 27]. These methods work reasonably well in terms of accuracy, as they generally lose only minimal accuracy compared to a "fine mesh only" solving procedure. However, these methods still require the practitioner to solve the linear system (1.7) that arises from a fine mesh—even if it is just once, this may still be very costly.

The first method we propose for efficient and accurate solving of (1.1)–(1.3) we call *incremental Picard-Yosida* (IPY). It is an alteration of the Picard iteration that uses an incremental formulation for the pressure, and an inexact LU factorization that yields an SPD Schur complement. In matrix form, each iteration can be written as

$$\begin{pmatrix} A_k & 0 \\ B^T & -B^T \tilde{A}^{-1} B \end{pmatrix} \begin{pmatrix} I & A_k^{-1} B \\ 0 & I \end{pmatrix} \begin{pmatrix} \hat{u}_k \\ \hat{\delta}_k^p \end{pmatrix} = \begin{pmatrix} F - B\hat{p}_{k-1} \\ 0 \end{pmatrix}, \tag{1.9}$$

where $\tilde{A}$ is the SPD part of A_k and $\hat{\delta}_k^p$ is the pressure update (see Algorithm 3.1 for the precise definition of IPY). The matrix $\tilde{A}$ is constructed by removing the nonlinear contributions from A_k, which makes $\tilde{A}$ independent of k and thus the Schur complement for IPY is the same at each iteration.

The only difference in the matrices that arise in IPY linear systems and Picard linear systems is the Schur complement. The IPY Schur complement is the Stokes-Schur complement with grad-div stabilization, and is both SPD and constant at each step of the iteration. A second key difference in IPY, which is critical for the iteration itself but does not affect the matrices, is that it relies on an incremental pressure formulation. For Picard, it is equivalent to use an incremental pressure formulation, however this equivalence is lost under Yosida approximations.

In particular, we note that without the incremental pressure formulation, a Yosida approximation applied to (1.8) need not converge to the correct solution (in particular, the limit of such a nonlinear iteration would not necessarily be discretely divergence-free; see Remark 3.2). As will be shown in our analysis and numerical tests, the use of grad-div stabilization is also critical for IPY to work. However, it is only required that $\gamma \geq \nu$ for good nonlinear convergence, which is a very mild restriction since typically $\nu < 1$, and γ would likely be chosen at least this large for accuracy purposes [21]. We note also the use of grad-div makes the system (1.9) related to approaches taken in [5, 6, 19] for preconditioning similar linear systems.

The proposed IPY method is very efficient because at each nonlinear iteration, it does two solves with A_k as the coefficient matrix, and one solve with a symmetric positive definite (SPD) Schur complement (that is, the same at each iteration) as the coefficient matrix. In particular, this means that only SPD Schur complement linear systems need solved, inner and outer iterative solvers can be conjugate gradient (CG), preconditioners can be reused, and are much easier to construct. Since grad-div stabilization is used, the outer Schur complement linear system solve is effectively preconditioned simply by the pressure mass matrix—this simple preconditioning of the outer solve is a key point, since even though other preconditioners may be better in the sense of iteration counts of the linear solvers (e.g., geometric and algebraic multigrid), they are typically harder to implement, can have higher setup costs, and can involve parameters.

We prove herein that under a small data condition and with grad-div parameter $\gamma \geq \nu$, the IPY nonlinear iteration will converge to the solution of the associated NSE finite element problem. Our numerical tests show that the number of nonlinear iterations required is about the same as when the Picard iteration is used. Hence, in short, the IPY method finds the same solution as Picard, and in about the same number of iterations, but with each iteration being both efficient and easy to implement.

We note that algebraic splitting methods are quite common for the time dependent NSE and related flow problems [3, 7, 13, 29–32], and they work by introducing $O(\Delta t^k)$ error (where Δt is the time step size) into the system to create linear systems of the form (1.9). These methods are known to be quite efficient for the same reasons IPY is efficient. However, since their use relies on approximations dependent on the time step size, extending them to be used on steady problems was not considered until the two recent papers [33, 38]. In [33], it was shown for a single linear solve that if grad-div stabilization is used in the finite element method that creates the saddle point linear system, then the Yosida-type splitting can be effective. The work of [38] showed that certain types of nonlinear iterations can also be effective for the steady NSE. The nonlinear iteration scheme we propose above is a careful combination of ideas from [33] and [38], and our proofs of convergence of the method utilize techniques from analysis of an incremental Yosida-type method for time dependent problems in [34].

This paper is arranged as follows. In section 2, we give mathematical preliminaries and notation, to allow a smooth analysis to follow. Section 3 analyzes and tests the incremental Picard-Yosida method. Here, we rigorously prove convergence, and test the method on several benchmark and application problems. In section 4, we extend the ideas to the Newton iteration, complete with development, analysis, and

testing of the incremental Newton-Yosida method. Finally, conclusions and future directions are given in section 5.

2. Mathematical preliminaries

We consider a domain $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$ that is open, connected, and with Lipschitz boundary $\partial\Omega$. The $L^2(\Omega)$ norm and inner product will be denoted by $\|\cdot\|$ and $(\cdot,\cdot)$, and $L^2_0(\Omega)$ denotes the zero mean subspace of $L^2(\Omega)$. Throughout this paper, it is understood by context whether a particular space is scalar or vector valued, and so we do not distinguish notation.

For the natural velocity and pressure spaces for Stokes and NSE, we use the notation

$$X := H^1_0(\Omega), \qquad Q := L^2_0(\Omega).$$

In the space X, the Poincaré inequality is known to hold: There exists $\lambda > 0$, dependent only on the size of Ω, such that for every $v \in X$,

$$\|v\| \le \lambda \|\nabla v\|.$$

The dual space of X will be denoted by X', with norm $\|\cdot\|_{-1}$.

Let τ_h be a conforming, shape-regular, and simplicial triangulation of Ω with h_T denoting the maximum element diameter. We denote with P_k the space of degree k globally continuous piecewise polynomials with respect to τ_h, and P_k^{disc} the space of degree k piecewise polynomials that can be discontinuous across elements.

Throughout the paper, we consider only discrete velocity-pressure spaces (X_h, Q_h) $\subset (X, Q)$ that satisfy the LBB condition: there exists a constant β, independent of h, satisfying

$$\inf_{q\in Q_h} \sup_{v\in X_h} \frac{(\nabla\cdot v, q)}{\|q\|\|\nabla v\|} \ge \beta > 0. \tag{2.1}$$

Common examples of such elements are (P_2, P_1) Taylor-Hood elements, and (P_k, P_{k-1}^{disc}) Scott-Vogelius (SV) elements on meshes with particular structure [2,40]; see also [11,16].

Define the discretely divergence-free velocity space by

$$V_h := \{v \in X_h,\ (\nabla\cdot v, q) = 0\ \forall q \in Q_h\}.$$

Define the skew-symmetric, trilinear operator $b^* : X \times X \times X \to \mathbb{R}$ by

$$b^*(u, v, w) := \frac{1}{2}(u\cdot\nabla v, w) - \frac{1}{2}(u\cdot\nabla w, v),$$

and recall from, e.g., [14], that there exists M depending only on Ω such that

$$|b^*(u, v, w)| \le M\|\nabla u\|\|\nabla v\|\|\nabla w\| \tag{2.2}$$

for every $u, v, w \in X$. As with most finite element NSE analyses, the use of the skew-symmetric form of the nonlinearity is critical in the analysis, although does not seem to make a difference in any of our computations if the convective formulation was used instead (results omitted).

2.1. **Discrete NSE.** The discrete NSE take the following form: find $(u,p) \in (X_h, Q_h)$ satisfying for all $(v,q) \in (X_h, Q_h)$,

$$b^*(u,u,v) - (p, \nabla \cdot v) + \nu(\nabla u, \nabla v) + \gamma(\nabla \cdot u, \nabla \cdot v) = (f,v), \tag{2.3}$$

$$(\nabla \cdot u, q) = 0. \tag{2.4}$$

Define the data-dependent constant $\alpha := M\nu^{-2}\|f\|_{-1}$, and we will refer to the following as a *small data condition*:

$$\alpha < 1. \tag{2.5}$$

Recall that if (2.5) holds, then (2.3)-(2.4) is well-posed and $\|\nabla u\| \leq \nu^{-1}\|f\|_{-1}$ [25,37].

Two common fixed point iterations for solving the nonlinear problem (2.3)–(2.4) are the Picard and Newton iterations, which are stated below. These iterations are classical, and are known to work well on many problems. However, they suffer from the problem we discuss above, which is that the linear systems that arise at each iteration can be very difficult to solve. Sections 3 and 4 present our proposed alternatives, which attempt to address some of the difficulties with these algorithms.

Algorithm 2.1. *The Usual Picard iteration for* (2.3)–(2.4) *takes the form*

Step 1*: Guess* $u_0 \in X_h$.
Step k*: Find* $(u_k, p_k) \in (X_h, Q_h)$ *satisfying for all* $(v,q) \in (X_h, Q_h)$,

$$b^*(u_{k-1}, u_k, v) - (p_k, \nabla \cdot v) + \nu(\nabla u_k, \nabla v) + \gamma(\nabla \cdot u_k, \nabla \cdot v) = (f,v),$$
$$(\nabla \cdot u_k, q) = 0.$$

Algorithm 2.2. *The Usual Newton iteration for* (2.3)–(2.4) *takes the form*

Step 1*: Guess* $u_0 \in X_h$.
Step k*: Find* $(u_k, p_k) \in (X_h, Q_h)$ *satisfying for all* $(v,q) \in (X_h, Q_h)$,

$$b^*(u_{k-1}, u_k, v) + b^*(u_k, u_{k-1}, v) - b^*(u_{k-1}, u_{k-1}, v)$$
$$-(p_k, \nabla \cdot v) + \nu(\nabla u_k, \nabla v) + \gamma(\nabla \cdot u_k, \nabla \cdot v) = (f,v),$$
$$(\nabla \cdot u_k, q) = 0.$$

Both of these iterations are known to converge (under small data conditions), and Newton is known to converge quadratically [14]. For completeness, and to motivate the IPY convergence proof in the next section, we provide next a proof for convergence of the Picard iteration. The proof of quadratic convergence of Newton is more technical, and we refer the interested reader to Theorem 6.3 in [14].

Lemma 2.1. *The Usual Picard iteration, Algorithm* 2.1*, converges to the unique solution of* (2.3)–(2.4)*, provided the small data condition* (2.5) *is satisfied.*

Proof. We first note that $\|\nabla u_k\| \leq \nu^{-1}\|f\|_{-1}$ for all $k = 1, 2, \ldots$, since taking $v = u_k$ in Algorithm 2.1 causes the nonlinear and pressure terms to vanish, leaving

$$\nu\|\nabla u_k\|^2 = (f, u_k) \leq \|f\|_{-1}\|\nabla u_k\|.$$

Since the iteration is linear, bounded, and finite dimensional, we have that each step of the iteration is well-posed.

To prove the iteration converges, we begin by setting $\delta_k = u_k - u_{k-1}$ and $\delta_k^p = p_k - p_{k-1}$ and consider the result of subtracting the iteration equations at Step k and $k+1$:

$$b^*(\delta_k, u_k, v) + b^*(u_k, \delta_{k+1}, v) - (\delta_{k+1}^p, \nabla \cdot v) + \nu(\nabla \delta_{k+1}, \nabla v) + \gamma(\nabla \cdot \delta_{k+1}, \nabla \cdot v) = 0.$$

Taking $v = \delta_{k+1}$ (note $\delta_{k+1} \in V_h$) causes the second nonlinear term and the pressure term to vanish, leaving

$$\begin{aligned} \gamma\|\nabla \cdot \delta_{k+1}\|^2 + \nu\|\nabla \delta_{k+1}\|^2 &= -b^*(\delta_{k+1}, u_k, \delta_k) \\ &\leq M\|\nabla \delta_{k+1}\|\|\nabla u_k\|\|\nabla \delta_k\| \\ &\leq M\nu^{-1}\|f\|_{-1}\|\nabla \delta_{k+1}\|\|\nabla \delta_k\|, \end{aligned}$$

with the last step utilizing the stability bound for u_k. This bound can be reduced to

$$\|\nabla \delta_{k+1}\| \leq \alpha \|\nabla \delta_k\|,$$

and thus since $\alpha < 1$ is assumed, the sequence $\{\delta_k\}_{k=1}^{\infty}$ is contractive, and therefore the Picard iteration converges, say to $(\tilde{u}, \tilde{p}) \in (X_h, Q_h)$. Since $u_k \in X_h$ and $p_k \in Q_h$ (due to the inf-sup condition), we have that $(\tilde{u}, \tilde{p})$ satisfies (2.3). Moreover, since $u_k \in V_h$ for all k, the fact that $\|\nabla(u_k - \tilde{u})\| \to 0$ implies that $\tilde{u} \in V_h$. But now $(\tilde{u}, \tilde{p})$ satisfies (2.3)–(2.4), which thus must be the system's unique solution. □

3. The incremental Picard-Yosida iteration

We now present and analyze our new incremental Picard-Yosida (IPY) method for solving the discrete steady NSE.

Algorithm 3.1. *The IPY for the steady NSE is defined by:*

Step 1*: Guess* $u_0 \in X_h$ *and* $p_0 \in Q_h$*.*

Step k *consists of the following* 3 *steps:*

k.1 *Find* $z_k \in X_h$ *satisfying for all* $v \in X_h$*,*

$$\gamma(\nabla \cdot z_k, \nabla \cdot v) + b^*(u_{k-1}, z_k, v) + \nu(\nabla z_k, \nabla v) = (f, v) + (p_{k-1}, \nabla \cdot v).$$

k.2 *Find* $(w_k, \delta_k^p) \in (X_h, Q_h)$ *satisfying for all* $(v, q) \in (X_h, Q_h)$*,*

$$\begin{aligned} \gamma(\nabla \cdot w_k, \nabla \cdot v) - (\delta_k^p, \nabla \cdot v) + \nu(\nabla w_k, \nabla v) &= 0, \\ (\nabla \cdot w_k, q) &= -(\nabla \cdot z_k, q). \end{aligned}$$

k.3 *Set* $p_k := p_{k-1} + \delta_k^p$ *and then find* $u_k \in X_h$ *satisfying for all* $v \in X_h$*,*

$$\gamma(\nabla \cdot u_k, \nabla \cdot v) + b^*(u_{k-1}, u_k, v) + \nu(\nabla u_k, \nabla v) = (f, v) + (p_k, \nabla \cdot v).$$

After discussing its implementation, we prove that this iteration converges to the discrete NSE solution (2.3)–(2.4), provided a small data condition holds.

Remark 3.2. A nonincremental version of Algorithm 3.1 can be shown to converge, but need not converge to the steady NSE solution (2.3)–(2.4). The limit will satisfy (2.3) but not necessarily (2.4) (theory can be shown with techniques similar to below, but is omitted).

3.1. **Implementation.** At iteration k, Algorithm (3.1) yields the following sequence of discrete problems:

$$(\nu K + C(\hat{u}_{k-1}) + \gamma D)\,\hat{z}_k = F - B^T \hat{p}_{k-1}, \tag{3.1}$$

$$\begin{bmatrix} \nu K + \gamma D & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} \hat{w}_k \\ \hat{\delta}_k^p \end{bmatrix} = \begin{bmatrix} 0 \\ -B\hat{z}_k \end{bmatrix}, \tag{3.2}$$

$$\hat{p}_k = \hat{\delta}_k + \hat{p}_{k-1}, \tag{3.3}$$

$$(\nu K + C(\hat{u}_{k-1}) + \gamma D)\,\hat{u}_k = F - B^T \hat{p}_k. \tag{3.4}$$

Here the *hat* notation represents the vector representation of the corresponding finite element function, K corresponds to the diffusive term matrix, $C(\hat{u}_{k-1})$ to the nonlinear term matrix, D to the grad-div stabilization term matrix, B^T to the gradient operator on the space Q_h, and B to the divergence operator on the space X_h. As we do not require w_k in the iteration, $\hat{w}_k$ can be eliminated from the system, and we need only solve

$$B\,(\nu K + \gamma D)^{-1}\,B^T \hat{\delta}_k = -B\hat{z}_k,$$

instead of (3.2).

This sequence of problems has several advantageous numerical properties. The computation of velocity and pressure is formally decoupled. This greatly reduces both the size and condition number of our problem [35]. We also observe that the Schur complement $B\,(\nu K + \gamma D)^{-1}\,B^T$ in the equation above is SPD and spectrally equivalent to the pressure mass matrix [5,19], allowing for easy preconditioning and efficient computation using Krylov methods using short recurrences (such as CG). Additionally, the Schur complement does not depend on the current iteration k and hence the associated preconditioner need only be assembled once. This is a marked advantage when compared to Usual Picard and Newton schemes, in which the Schur complement is not SPD in general, and preconditioners often require updating [10].

We observe that the discrete problems (3.1)–(3.4) can be recombined into a block linear system: writing $A_k = (\nu K + C(\hat{u}_{k-1}) + \gamma D)$ and $\tilde{A} = (\nu K + \gamma D)$, and setting $\hat{p}_k = \hat{p}_{k-1} + \hat{\delta}_k^p$, yields precisely (1.9).

3.2. **Convergence.** We now prove convergence of the IPY iteration. The proof requires a small data condition, a sufficiently good initial guess, and a grad-div stabilization parameter $\gamma \geq \nu$. As discussed above, since ν is generally small for problems of interest where fine meshes are required, and $\gamma = O(1)$ is a good choice for smaller error, this is a very weak assumption on γ in practice. The small data condition we require for IPY convergence is

$$\alpha < \min\{1, (8(5\nu + 2))^{-1}\beta^2, (16\beta^{-2} + 2)^{-1}\}, \tag{3.5}$$

where β is the inf-sup constant. Comparing to the Usual Picard iteration, here we observe a greater restriction on α, which seems unavoidable in the analysis due to the approximation on the pressure, which must be handled in the analysis and brings the inf-sup constant into the analysis. We note in our numerical tests that there was no lack of robustness or in convergence of IPY compared to Usual Picard.

Theorem 3.1. *Assume the small data condition* (3.5) *is satisfied,* $\gamma \geq \nu$, *and* $\nu\|\nabla(u - u_0)\|^2 + \gamma^{-1}\|p - p_0\|^2 \leq \|\nabla u\|^2$, *where* (u, p) *is the solution of* (2.3)–(2.4)

and $(u_0, p_0) \in (X_h, Q_h)$ *is the initial guess. For* $(u_k, p_k) \in (X_h, Q_h)$ *the Step* k *solution of Algorithm* 3.1*, we have that:*

$$\gamma^{-1}\|p - p_k\|^2 + \nu\|\nabla(u - u_k)\|^2 \quad \leq \quad \alpha\left(\gamma^{-1}\|p - p_{k-1}\|^2 + \nu\|\nabla(u - u_{k-1})\|^2\right).$$

Remark 3.3. If we additionally assume $\left(8(5\nu + 2)\beta^{-2}\alpha^2 + (16\beta^{-2} + 1)\right)\alpha^\epsilon \leq 1$ for $\epsilon > 0$, then from (3.13) in the proof below, we can obtain

$$\gamma^{-1}\|p - p_k\|^2 + \nu\|\nabla(u - u_k)\|^2 \quad \leq \quad \alpha^{2-\epsilon}\left(\gamma^{-1}\|p - p_{k-1}\|^2 + \nu\|\nabla(u - u_{k-1})\|^2\right).$$

This implies a contraction ratio of "almost α", which suggests under this data condition that IPY should converge at approximately the same rate as Usual Picard.

Proof. Denote $e_k^u := u - u_k$ and $e_k^z := u - z_k$. Our proof will assume $\nu\|\nabla e_{k-1}^u\|^2 + \gamma^{-1}\|p - p_{k-1}\|^2 \leq \|\nabla u\|^2$, and by proving that the sequence defined by $\nu\|\nabla e_{k-1}^u\|^2 + \gamma^{-1}\|p - p_{k-1}\|^2$ is decreasing, this will imply the condition at the next iteration.

Subtracting Step k.1 from the unique steady solution equation (2.3), we obtain for all $v \in X_h$,

$$\gamma(\nabla \cdot e_k^z, \nabla \cdot v) + \nu(\nabla e_k^z, \nabla v) = (p - p_{k-1}, \nabla \cdot v) - b^*(e_{k-1}^u, u, v) - b^*(u_{k-1}, e_k^z, v).$$

Choose $v = e_k^z$. This causes the last nonlinear term to vanish, and then applying Cauchy-Schwarz and Young's inequalities to the pressure term, and applying (2.2) to the nonlinear term, we produce the bound

$$\begin{aligned}\gamma\|\nabla \cdot e_k^z\|^2 + \nu\|\nabla e_k^z\|^2 &= (p - p_{k-1}, \nabla \cdot e_k^z) - b^*(e_{k-1}^u, u, e_k^z) \\ &\leq \frac{\gamma}{2}\|\nabla \cdot e_k^z\|^2 + \frac{1}{2\gamma}\|p - p_{k-1}\|^2 + M\|\nabla e_{k-1}^u\|\|\nabla u\|\|\nabla e_k^z\|.\end{aligned}$$

Using the a priori bound $\|\nabla u\| \leq \nu^{-1}\|f\|_{-1}$, Young's inequality, and the definition of α on the last term, we arrive at

$$\gamma\|\nabla \cdot e_k^z\|^2 + \nu\|\nabla e_k^z\|^2 \leq \gamma^{-1}\|p - p_{k-1}\|^2 + \nu\alpha^2\|\nabla e_{k-1}^u\|^2. \tag{3.6}$$

Similarly, for Step k.3, we obtain

$$\gamma\|\nabla \cdot e_k^u\|^2 + \nu\|\nabla e_k^u\|^2 \leq \gamma^{-1}\|p - p_k\|^2 + \nu\alpha^2\|\nabla e_{k-1}^u\|^2. \tag{3.7}$$

Next, we bound $\|p - p_k\|$. Begin by adding Steps k.1 and k.2, which gives for all $v \in X_h$,

$$\gamma(\nabla \cdot (w_k + z_k), \nabla \cdot v) + \nu(\nabla(w_k + z_k), \nabla v) = (p_k, \nabla \cdot v) - b^*(u_{k-1}, z_k, v) + (f, v),$$

and we note that $(w_k + z_k) \in V_h$. Subtracting the unique steady solution equation (2.3) from this, we obtain the error equation

$$\begin{aligned}&\gamma(\nabla \cdot (w_k + z_k - u), \nabla \cdot v) + \nu(\nabla(w_k + z_k - u), \nabla v) \\ &\quad = (p_k - p, \nabla \cdot v) - b^*(e_{k-1}^u, e_k^z, v) - b^*(u, e_k^z, v) - b^*(e_{k-1}^u, u, v) \ \forall v \in X_h.\end{aligned} \tag{3.8}$$

Choosing $v = (w_k + z_k - u) \in V_h$ causes the pressure term to vanish, and yields the bound

$$\begin{aligned}&\gamma\|\nabla \cdot (w_k + z_k - u)\|^2 + \nu\|\nabla(w_k + z_k - u)\|^2 \\ &\qquad \leq 2M^2\nu^{-1}\|\nabla e_{k-1}^u\|^2\|\nabla e_k^z\|^2 + \nu\alpha^2\|\nabla e_k^z\|^2 + \nu\alpha^2\|\nabla e_{k-1}^u\|^2,\end{aligned} \tag{3.9}$$

thanks to (2.2), Young's inequality, and the bound on u. Using the assumption that $\nu\|\nabla e_{k-1}^u\|^2 \leq \|\nabla u\|^2$, this reduces to

$$\gamma\|\nabla \cdot (w_k + z_k - u)\|^2 + \nu\|\nabla(w_k + z_k - u)\|^2 \leq (\nu + 2)\alpha^2\|\nabla e_k^z\|^2 + \nu\alpha^2\|\nabla e_{k-1}^u\|^2. \tag{3.10}$$

We now use this bound to bound the pressure error, after applying inf-sup to (3.8) to find

$$\begin{aligned}\beta\|p-p_k\| &\le \gamma\|\nabla\cdot(w_k+z_k-u)\| + \nu\|\nabla(w_k+z_k-u)\| \\ &\quad + 2M\|\nabla u\|\|\nabla e_k^z\| + M\|\nabla u\|\|\nabla e_{k-1}^u\| \\ &\le \gamma\|\nabla\cdot(w_k+z_k-u)\| + \nu\|\nabla(w_k+z_k-u)\| \\ &\quad + 2\alpha\nu\|\nabla e_k^z\| + \alpha\nu\|\nabla e_{k-1}^u\|.\end{aligned}$$

Squaring both sides, using that $\gamma \ge \nu$, and reducing yields

$$\begin{aligned}\beta^2\|p-p_k\|^2 \le 4\gamma(\gamma\|\nabla\cdot(w_k+z_k-u)\|^2 + \nu\|\nabla(w_k+z_k-u)\|^2 \\ + 4\alpha^2\nu\|\nabla e_k^z\|^2 + \alpha^2\nu\|\nabla e_{k-1}^u\|^2).\end{aligned}$$

Using the bound (3.10) reduces this estimate to

$$\|p-p_k\|^2 \quad\le\quad 4\gamma\beta^{-2}\left((5\nu+2)\alpha^2\|\nabla e_k^z\|^2 + 2\nu\alpha^2\|\nabla e_{k-1}^u\|^2\right). \tag{3.11}$$

Combining (3.11) with (3.6) and multiplying both sides by γ^{-1} yields

(3.12)

$$\begin{aligned}&\gamma^{-1}\|p-p_k\|^2 \\ &\quad\le 4\beta^{-2}\left((5\nu+2)\alpha^2\left(\gamma^{-1}\|p-p_{k-1}\|^2 + \nu\alpha^2\|\nabla e_{k-1}^u\|^2\right) + 2\nu\alpha^2\|\nabla e_{k-1}^u\|^2\right) \\ &\quad\le 4\beta^{-2}\left((5\nu+2)\alpha^2\gamma^{-1}\|p-p_{k-1}\|^2 + (5\nu+2)\nu\alpha^4\|\nabla e_{k-1}^u\|^2 + 2\nu\alpha^2\|\nabla e_{k-1}^u\|^2\right).\end{aligned}$$

Next, use (3.12) in (3.7) to find that

$$\begin{aligned}&\gamma\|\nabla\cdot e_k^u\|^2 + \nu\|\nabla e_k^u\|^2 \\ &\quad\le 4(5\nu+2)\beta^{-2}\alpha^2\gamma^{-1}\|p-p_{k-1}\|^2 \\ &\qquad + \left(4(5\nu+2)\beta^{-2}\nu\alpha^4 + (8\beta^{-2}+1)\nu\alpha^2\right)\|\nabla e_{k-1}^u\|^2 \\ &\quad= \alpha^2\left(4(5\nu+2)\beta^{-2}\gamma^{-1}\|p-p_{k-1}\|^2\right. \\ &\qquad \left. + \left(4(5\nu+2)\beta^{-2}\nu\alpha^2 + (8\beta^{-2}+1)\nu\right)\|\nabla e_{k-1}^u\|^2\right).\end{aligned}$$

Add this bound with (3.12) to obtain

$$\begin{aligned}&\gamma^{-1}\|p-p_k\|^2 + \gamma\|\nabla\cdot e_k^u\|^2 + \nu\|\nabla e_k^u\|^2 \\ &\quad\le 8(5\nu+2)\beta^{-2}\alpha^2\gamma^{-1}\|p-p_{k-1}\|^2 \\ &\qquad + \left(8(5\nu+2)\beta^{-2}\nu\alpha^4 + (16\beta^{-2}+1)\nu\alpha^2\right)\|\nabla e_{k-1}^u\|^2.\end{aligned} \tag{3.13}$$

Now using the small data condition (3.5), we find that

$$\begin{aligned}&\gamma^{-1}\|p-p_k\|^2 + \nu\|\nabla e_k^u\|^2 \\ &\quad\le \alpha\left(\gamma^{-1}\|p-p_{k-1}\|^2 + \alpha\left(\nu\alpha + (16\beta^{-2}+1)\nu\right)\|\nabla e_{k-1}^u\|^2\right) \\ &\quad\le \alpha\left(\gamma^{-1}\|p-p_{k-1}\|^2 + \alpha(16\beta^{-2}+2)\nu\|\nabla e_{k-1}^u\|^2\right) \\ &\quad\le \alpha\left(\gamma^{-1}\|p-p_{k-1}\|^2 + \nu\|\nabla e_{k-1}^u\|^2\right).\end{aligned}$$

We have thus proven that $\gamma^{-1}\|p-p_k\|^2 + \nu\|\nabla e_k^u\|^2$ is a contractive sequence in k, and thus converges. Since the solution of the finite dimensional problem (2.3)–(2.4) is unique and bounded by the data (since $\alpha < 1$), we have that the limit of the IPY iteration converges to the solution of (2.3)–(2.4). □

3.3. **Numerical tests for IPY.** We now test Algorithm 3.1, the IPY iteration, on several benchmark problems. For a convergence criteria of the nonlinear solvers, we use the L^2 norm of successive iterates being below a tolerance, $\|u_k - u_{k-1}\| < tol$ (since the problems we consider below have O(1) velocities, this is equivalent to using a relative tolerance). Due to the theory above, this is equivalent to monitoring nonlinear residuals (which we did also, as a sanity check in our codes). For the 2D tests and the Aneurisk 32 test case, Freefem++ software [18] was used, and for 3D tests, the authors' Matlab codes were used. Unless otherwise noted, initial conditions were taken to be zero. There are several properties of IPY that we test in these experiments, including effectiveness as a nonlinear solver, the effect of the various parameters on convergence, and also efficiency by comparing IPY to Usual Picard. For clarity of exposition, for each numerical test, we investigate a subset of these properties. Tests were run on the authors' workstations, but all comparisons were made on the same machine for each respective test.

We refer below to "outer solvers" and "inner solvers" for systems containing the Schur complement matrices $S = B^T A^{-1} B$ (or replace A with $\tilde{A}$). By "outer solver" we refer to a linear solve routine for $Sx = b$. Since we cannot explicitly form S, these solves are performed with an iterative method (e.g., CG or BICGSTAB), for which at each iteration a matrix-vector multiply of S to a vector y is done. But to calculate Sy, one has to apply the action of A^{-1} to By, which results in the need to solve a linear system of the form $Az = By$. In this sense, the solver used to solve this $Az = By$ system is nested inside each iteration of the outer solver, and is thus referred to as the inner solver.

Due to the use of grad-div stabilization, care must be taken when using iterative linear solvers on systems with coefficient matrices A and $\tilde{A}$. Results from the literature, e.g., [5, 19] suggest that GMRES or BICGSTAB with certain multigrid preconditioners can be quite effective. While better solvers for these matrices exist when $\gamma = 0$, the use of $\gamma = O(1)$ improves error so that coarser meshes / smaller matrices are often possible (for the same error tolerance of the finite element approximation) and improves solves with the Schur complement. Hence the trade-off of having these "more difficult" A-block (or $\tilde{A}$-block) solves due to grad-div seems well worth it. However, how to best precondition and solve linear systems containing these matrices when grad-div is used remains an open problem, and should get more attention in the near future due to the more prevalent recent use of grad-div in incompressible flow simulations.

3.3.1. *Analytic solution test.* We first test IPY on a problem with an analytic solution, in order to analyze its convergence and its dependence on the mesh level h.

We consider here the steady NSE defined on $\Omega = [0, 1]^2$ with $\nu = 0.01$ and forcing

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

It can be verified that the problem yields the following exact solution:

$$u_{ex} = [-\sin(x)\cos(y), \cos(x)\sin(y)]'; \quad p_{ex} = \sin(x) + \sin(y).$$

This test illustrates linear convergence of IPY to the discrete NSE solution. It also demonstrates how convergence is affected by the mesh level, in terms of both accuracy and convergence rate. We run the iteration on three different mesh levels, h=1/20, 1/40, and 1/80, with $\gamma = 1$, and compare our computed solutions with the

discrete NSE solution on the corresponding mesh; see Figure 1. The results are as anticipated in the sense that linear convergence to the solution is observed. Moreover, no deterioration of the convergence is observed as h decreases. Interestingly, on the coarser meshes, an inflection point appears around the tenth iteration, after which the convergence is slowed, but still linear. This suggests a more complicated analysis might be possible, where the contraction ratio can change in different regions, based on problem parameters. However, this phenomenon is not significant on the finest mesh, which allows for improved convergence in this case.

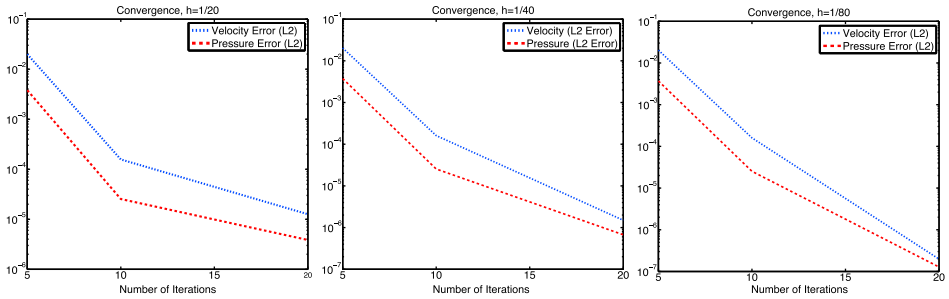

FIGURE 1. The error after each iteration of the IPY scheme, comparing to the discrete NSE solution on the corresponding mesh, for varying mesh widths.

3.3.2. *3D lid-driven cavity.* Our next test problem is the 3D lid-driven cavity benchmark. The domain is $(0,1)^3$ and the problem has no forcing $f = 0$, homogeneous Dirichlet boundary conditions on the bottom and side walls, and the Dirichlet condition $u_{\text{lid}} = \langle 1, 0, 0 \rangle^T$ on the lid ($z = 1$). We consider the case of $Re = \nu^{-1} = 100$, and refer to [39] and the references therein for a more detailed problem description. The purpose of this test is to investigate the effectiveness of IPY on a larger and more difficult test problem, to test IPY using varying element choices, and to compare the ability of IPY to solve the nonlinear problem to that of Usual Picard, under the assumption of ideal linear solvers.

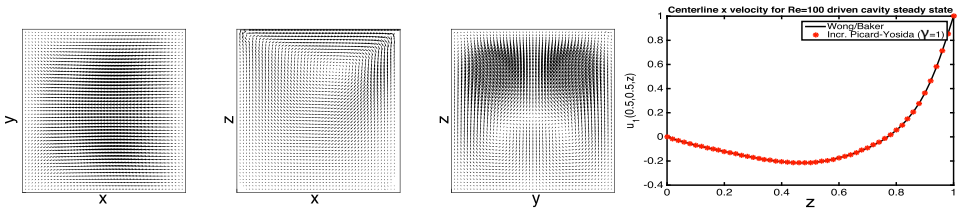

FIGURE 2. Centerplane slices of the IPY velocity field, and the centerline x-velocity for the IPY solution (as red dots) along with the solution from [39], using Re=100, Scott-Vogelius elements, and 796,722 total degrees of freedom.

We test the IPY and Usual Picard methods using different element choices and various meshes, grad-div parameter $\gamma = 1$, an initial velocity that is zero in the

interior but satisfies the boundary conditions, and an initial pressure $p_0 = 0$. A nonlinear solver tolerance of 10^{-6} in the $L^2(\Omega)$ norm was used for both methods. Uniform tetrahedral meshes were used, and for the Scott-Vogelius tests, a barycenter refinement of the uniform mesh was used in order to guarantee inf-sup stability [40]. We note that grad-div has no effect on solutions found with Scott-Vogelius elements, but it is still necessary for IPY convergence; it does, however, have the effect that it allows the IPY Schur complement to be effectively preconditioned with the pressure mass matrix [5, 19].

Since the purpose of this test is to compare nonlinear solvers, we solve Steps k.1 and k.3 of IPY with a direct solver, and use a direct solver for the inner solves of Step k.2 (via an LUPQ factorization that gets created just once for IPY, and once at each nonlinear iteration for Usual Picard). For the IPY Step 2 outer solver, we use CG with tolerance 10^{-8} (here and below, iterative linear solver tolerances refer to the relative residual), and precondition with the pressure mass matrix. For simplicity, we solve the linear system of Usual Picard in a very similar way, by treating the linear systems as in (1.8) and performing a three step solve process. Steps 1 and 3 are the same as for IPY, and again we use direct solvers. For Usual Picard Step 2, we use BICGSTAB with tolerance 10^{-8}, as well as GMRES with tolerance 10^{-8} (for comparison), precondition with the pressure mass matrix, and use a direct solver for the inner solves.

Table 1 shows statistics from these computations. The key observation is that in all cases, the number of nonlinear iterations needed by IPY is the same as for Usual Picard. This is important for IPY to be useful, since even though the linear solves of IPY have such nice properties, if too many nonlinear iterations are needed, then IPY will not give overall improvement compared to Usual Picard. Our theory shows that IPY may have a larger contraction ratio than Usual Picard, however, from Remark 3.3 we know that under a data restriction, the contraction ratios of IPY and Usual Picard are about the same, which is what we observe in this test.

Although timings and efficiency are solver / preconditioner dependent, Table 1 also shows average iteration counts and timings for the Schur complement system solves (not including the LUPQ factorizations). We observe the number of outer iterations are slightly lower for IPY compared to Usual Picard using BICGSTAB, but significantly lower than for Usual Picard using GMRES. For timings, we observe IPY to be significantly faster than Usual Picard with either BICGSTAB and GMRES (which take about the same amount of time even though GMRES needs more iterations, which is likely due to BICGSTAB doing more matrix-vector products).

Figure 2 shows plots of the IPY solution from the 796,722 total degrees of freedom (dof) Scott-Vogelius solution. Figure 2 shows the sliceplane plots of the velocity field, and these agree well with those found in Wong and Baker's results from [39]. Also shown is the centerline x velocity for the IPY solution and the Wong and Baker solution, and they are in excellent agreement.

3.3.3. *A 2D bifurcation flow.* For our third test, we compute with IPY for different values of γ and ν on a 2D bifurcation inspired by the geometry used in [20]. The purposes of this test are to assess the robustness of IPY with respect to ν, to investigate the impact of γ on convergence, and to compare the convergence of IPY to that of Usual Picard. We choose parameters $\nu = 0.0133$ and 0.00667 (which

TABLE 1. Re=100 3D driven cavity statistics for IPY and Usual Picard solvers, using various element choices and meshes.

(P_2, P_1) **Taylor-Hood elements (γ=1)**

	IPY		Usual Picard		
dof	nonlin its	avg PCG its / sec	nonlin its	avg BICGSTAB its/sec	avg GMRES its/sec
1,093	11	16 / 0.02	11	16 / 0.03	25 / 0.03
10,637	14	14 / 0.36	14	14 / 0.60	24 / 0.57
38,229	17	12 / 2.69	17	15 / 4.56	25 / 4.26
93,469	19	12 / 9.26	19	15 / 19.75	25 / 20.82
185,957	20	12 / 26.80	20	15 / 63.90	26 / 67.31

(P_3, P_2^{disc}) **Scott-Vogelius elements** (γ=1)

	IPY		Usual Picard		
dof	nonlin its	avg PCG its / sec	nonlin its	avg BICGSTAB its/sec	avg GMRES its/sec
702	8	8 / 0.01	8	8.5 / 0.01	13 / 0.01
16,770	17	8 / 0.10	17	8.5 / 0.16	14 / 0.17
206,874	21	7 / 1.79	21	9.5 / 3.71	15 / 3.89
796,722	21	7 / 11.91	21	9.5 / 25.03	16 / 25.66

correspond to Re = 200 and 400) and γ =1 and 3. Plots of the velocity field for these choices of ν are shown in Figure 3.

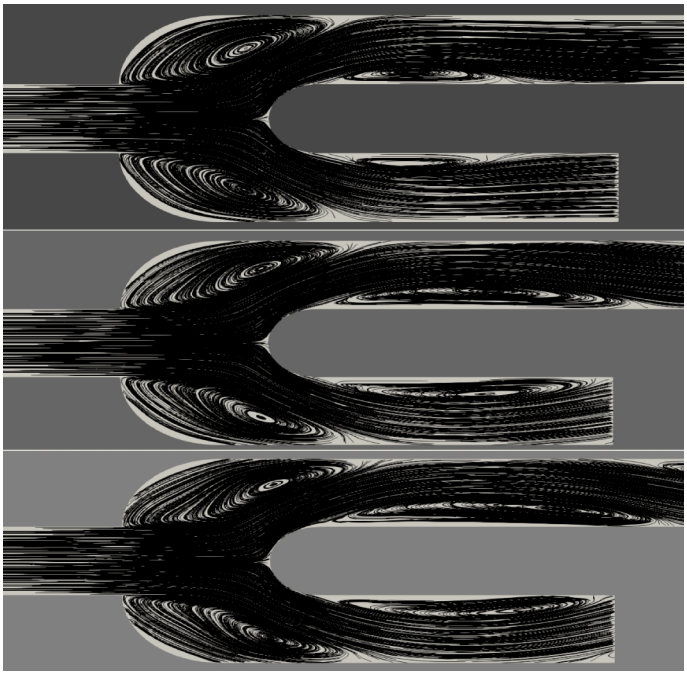

FIGURE 3. Reference solutions for $\nu = .0133$ (top) and $\nu = .0067$ (bottom).

We use (P_2, P_1) Taylor-Hood finite elements on a mesh with 3,493 elements, leading to 16,434 total dof. Along the walls, we prescribe no-slip boundary conditions,

TABLE 2. Iteration statistics for a 2D bifurcation test, with Usual Picard using the linear solver from [19] .

2D Bifurcation Test, $\nu = .0133$ (Re=200)

	IPY		Usual Picard	
γ	nonlin its	avg PCG its / sec	nonlin its	avg GMRES its / sec
1.0	62	12 / 0.26	48	20 / 0.38
3.0	51	11 / 0.26	48	14 / 0.28

2D Bifurcation Test, $\nu = .00667$ (Re=400)

	IPY		Usual Picard	
γ	nonlin its	avg PCG its / sec	nonlin its	avg GMRES its / sec
1.0	96	12 / 0.26	72	20 / 0.38
3.0	73	11 / 0.25	75	14 / 0.28

at the inlet we prescribe a standard parabolic inflow profile with peak velocity 2, and at the two outlets we assign zero-traction outflow conditions.

We use a different approach to linear system solving for Usual Picard in this test and the next, solving the problem monolithically. At each iteration we solve the full saddle point system using GMRES preconditioned with the following block-triangular preconditioner found in [19]:

$$P^{-1} = \begin{bmatrix} A & B^T \\ 0 & (\nu+\gamma)^{-1} M_p \end{bmatrix}^{-1}, \tag{3.14}$$

where M_p is the lumped pressure mass matrix and A the velocity block with grad-div stabilization. This will allow for additional comparison of IPY vs. Usual Picard with different solvers, since the efficiency of Usual Picard is more heavily dependent on the linear solvers that are used. We use a tolerance of 10^{-6} for the outer GMRES solver and solve the linear systems arising from the inner blocks in (3.14) directly. For IPY we solve Steps 1 and 3 using a sparse direct solver and Step 2 with CG preconditioned with the pressure mass matrix, with tolerance set to 10^{-6}. The inner solves of Step 2 are also computed with a sparse direct solver. A nonlinear solver tolerance was chosen as 10^{-6} for both IPY and Usual Picard.

We report our results in Table 2. We note that while the value of γ did affect aspects of the linear solve in Usual Picard, it did not have any discernible impact on the convergence of the nonlinear iteration. For $\nu = .0133$, the convergence of Usual Picard was better than IPY for $\gamma = 1$ (48 vs. 62 iterations), but for γ=3 the number of iterations needed for convergence was almost the same. Similar results are observed for $\nu = .00667$.

In terms of the numerical cost at each iteration, as reported in Table 2, we see consistent solve times and outer iteration counts for the different values of ν for a given γ, suggesting little dependence on ν for both methods. The changes in γ do not appear to affect the cost of each iteration of IPY, though this has a mild but observable positive effect on Usual Picard. Overall, from this test we conclude that the convergence of IPY compared to Usual Picard depends on both ν and γ, with improvement in IPY coming from slightly higher γ when using lower values of ν.

3.3.4. *ANEURISK case* 32. Our last test with the IPY method is for a nontrivial application problem, illustrating the possible practical usefulness of the proposed scheme. We compute the steady-state solutions on case 32 from the publicly available ANEURISK database [1] over a range of different Reynolds numbers with

varying flow rates and viscosities. This is a patient-specific reconstruction of a basilar artery with an aneurysm at the terminal bifurcation. This case is intended to demonstrate IPY's viability for large-scale three-dimensional problems with complex geometries over different flow configurations. Such problems represent an area where the proposed approach may significantly reduce computational costs for the reasons discussed above. For the case Re=40, we will compare the IPY solution to the time-average of the benchmark (unsteady) computation. For the other cases, Re=100 and 160, we do not have benchmark data and this test is strictly a comparison of solver efficiency.

We ran these simulations with Taylor-Hood elements on a moderately fine tetrahedral mesh, with 36,693 total elements and 224,180 total dof. In accordance with the benchmark computation, we set $\nu = .04$ g/cm-s and inflow Q=.4005 ml/s based on the values found in [8, 9]. For Re=100 and 160 we run two configurations: one in which we keep $\nu = .04$ and vary Q, and one in which we keep Q=.4005 ml/s and vary ν. We note despite the identical Reynolds numbers these configurations are not the same, physically or numerically. We prescribe no-slip boundary conditions along the vessel walls and homogenous Neumann conditions at the outflows in all cases, using parabolic Poiseuille profiles for inflow, properly scaled to match the desired Q. The IPY method is then used with $\gamma = 1$ to solve the nonlinear problem, using a tolerance of 10^{-3}, which is purposely chosen large (relative to the usual small tolerance) since large measurement errors are already present in the data. We ran the simulations in FreeFem++, computing each step of IPY and Usual Picard with the same setup used in the bifurcation test problem above.

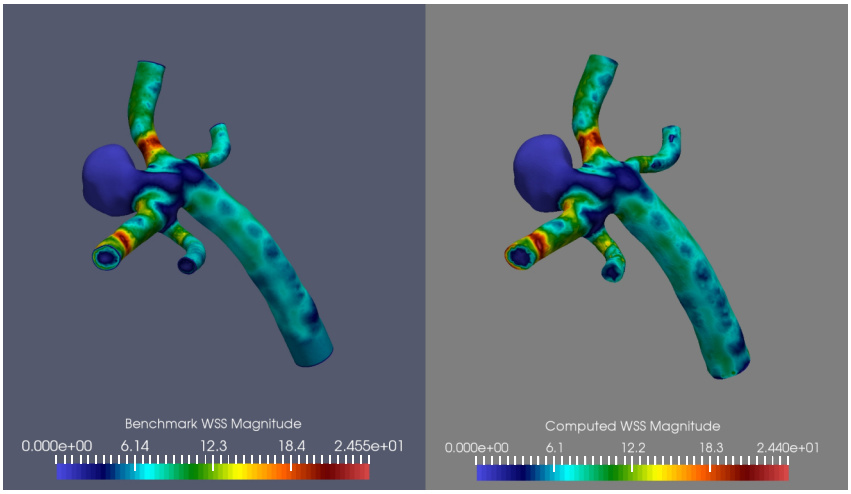

FIGURE 4. Wall shear stresses for Re = 40 from benchmark time average data (left) and IPY steady solution (right).

In Table 3, we report performance statistics for both methods at each problem configuration. For both IPY and Usual Picard, the number of required nonlinear iterations increases as both ν decreases and Q increases. The number of nonlinear iterations is identical for Usual Picard for cases with the same Reynolds number, however for IPY we observe minor differences, with the high ν/high Q cases converging slightly slower. On the whole, we see that IPY requires a few extra iterations

TABLE 3. Avg. time/solve only includes the system solve time; avg. total time includes assembly costs. Outer Krylov iterations denotes avg. num. of outer GMRES iterations for GMRES and outer PCG iterations for the Schur complement step for splitting methods per nonlinear iteration.

3D BSA Aneurysm, 224,180 DOF: $Q = 0.4005$ ml/s, $\nu = .04$ g/cm-s (Re=40)				
Method	nonlin its	Outer Krylov its	avg solve time (sec)	total time (sec)
IPY	12	32	34.5	804.0
Usual Picard	7	48	47.9	713.6
3D BSA Aneurysm, 224,180 DOF: $Q = 1.005$ ml/s, $\nu = .04$ g/cm-s (Re=100)				
Method	nonlin its	Outer Krylov its	avg solve time (sec)	total time (sec)
IPY	19	31	32.2	1212.2
Usual Picard	11	66	113.2	1245.2
3D BSA Aneurysm, 224,180 DOF: $Q = 0.4005$ ml/s, $\nu = .016$ g/cm-s (Re=100)				
Method	nonlin its	Outer Krylov its	avg solve time (sec)	total time (sec)
IPY	15	28	31.4	955.5
Usual Picard	11	47	47.1	965.8
3D BSA Aneurysm, 224,180 DOF: $Q = 1.6005$ ml/s, $\nu = .04$ g/cm-s (Re=160)				
Method	nonlin its	Outer Krylov its	avg solve time (sec)	total time (sec)
IPY	30	31	32.5	1917.0
Usual Picard	28	90	95.3	3880.8
3D BSA Aneurysm, 224,180 DOF: $Q = 0.4005$ ml/s, $\nu = .01$ g/cm-s (Re=160)				
Method	nonlin its	Outer Krylov its	avg solve time (sec)	total time (sec)
IPY	28	26	28.6	1699.6
Usual Picard	28	45	47.7	2514.4

to converge compared to Usual Picard; over the set of simulations, IPY required an average of four extra iterations. It is worth observing that the number of extra necessary iterations did not appear related to the Reynolds number. We note that IPY was able to find good results on this problem, and in Figure 4 we observe that the IPY solution matches the long time average benchmark solution quite well. Since our computation is steady, and the benchmark comes from time-averaging an unsteady computation, we do not expect perfect agreement. Additional statistics were calculated for the IPY solution, including wall shear stress along a vessel, and these all matched the benchmark data quite well (these results are omitted).

We now turn our attention to the cost for each nonlinear iteration. For both our method and the comparison, the dominant driver of costs are the solves in X_h (i.e., the solves with A and $\tilde{A}$), the total number of which is determined by the number of outer GMRES iterations for Usual Picard and the number of outer iterations in Step 2 in IPY (plus two more for Steps 1 and 3). Referring to the results in Table 3, we see that while the preconditioner (3.14) for Usual Picard is robust with respect to ν, confirming the findings in [19], its performance worsens as Q increases. In comparison, IPY is robust with respect to both parameters, showing no sensitivity at all to Q and a mild *improvement* in performance as ν decreases. Although IPY requires a similar (or slightly larger) number of nonlinear iterations to converge, the cost per iteration is much lower, reflected in the low number of outer iterations. As the Reynolds number increases we see improved performance of IPY with respect to total time, but the opposite with Usual Picard: IPY slightly

underperforms Usual Picard at Re=40, performs comparably for both cases with Re=100, and significantly outperforms Usual Picard for both cases where Re=160.

4. The incremental Yosida-Newton iteration

We consider in this section similar ideas of section 3 for the IPY iteration, but here apply them to the Newton iteration. That is, we develop a new iteration based on using an incremental Newton algorithm combined with Yosida algebraic splitting, and a grad-div stabilized finite element discretization. Like the IPY, the incremental Newton-Yosida (INY) will have an SPD Schur complement that is the same at each iteration, which is advantageous compared to the Usual Newton linear systems which have nonsymmetric Schur complements that change at each iteration. After presenting the algorithm, we prove it converges to the discrete NSE solution, and finally give numerical results.

The INY iteration is defined as follows.

Algorithm 4.1. *The incremental Newton-Yosida iteration for the steady Navier-Stokes is given by:*

Step 1*: Guess* $u_0 \in X_h$, $p_0 \in Q_h$.

Step k *consists of the following three steps:*

k.1 *Find* $z_k \in X_h$ *satisfying for all* $v \in X_h$,

$$\begin{aligned}\gamma(\nabla\cdot z_k, \nabla\cdot v) + b^*(u_{k-1}, z_k, v) + b^*(z_k, u_{k-1}, v) + \nu(\nabla z_k, \nabla v)\\ = (f, v) + (p_{k-1}, \nabla\cdot v) + b^*(u_{k-1}, u_{k-1}, v).\end{aligned}$$

k.2 *Find* $(w_k, \delta_k^p) \in (X_h, Q_h)$ *satisfying for all* $(v, q) \in (X_h, Q_h)$,

$$\begin{aligned}\gamma(\nabla\cdot w_k, \nabla\cdot v) - (\delta_k^p, \nabla\cdot v) + \nu(\nabla w_k, \nabla v) &= 0,\\ (\nabla\cdot w_k, q) &= -(\nabla\cdot z_k, q).\end{aligned}$$

k.3 *Set* $p_k = p_{k-1} + \delta_k^p$ *and find* $u_k \in X_h$ *satisfying for all* $v \in X_h$,

$$\begin{aligned}\gamma(\nabla\cdot u_k, \nabla\cdot v) + b^*(u_{k-1}, u_k, v) + b^*(u_k, u_{k-1}, v) + \nu(\nabla u_k, \nabla v)\\ = (f, v) + (p_k, \nabla\cdot v) + b^*(u_{k-1}, u_{k-1}, v).\end{aligned}$$

The implementation should be done at the algebraic level, with the corresponding finite element submatrices, analogous to the IPY implementation discussed in section 3.

4.1. Convergence.

We now prove convergence of the INY iteration. As expected, due to the complexity of Newton compared with Picard, both the proof and the data restriction are also more complex.

Theorem 4.1. *Let* $\epsilon > 0$ *and define*

$$\tilde{\alpha} < \min\left\{1, (1+24\beta^{-2})^{-1},\ 1 - \left(\frac{80^2(1+\epsilon)^3 M\|f\|_{-1}^3}{\nu^2\beta^4} - \frac{24\nu}{\beta^2} + \frac{\nu\beta^2}{80}\right)^{-1/2}\right\}.$$

Denote by (u, p) *the solution of system* (2.3)–(2.4)*,* $(u_0, p_0) \in (X_h, Q_h)$ *the initial guess for INY, and* (u_k, p_k) *the INY Step* k *solution. Then if* $\gamma \geq \nu$ *and* $\nu(1-\tilde{\alpha})\|\nabla(u-u_0)\|^2 + \gamma^{-1}\|p-p_0\|^2 \leq \|\nabla u\|^2$*, the sequence* (u_k, p_k) *converges to* (u, p).

Remark 4.2. Even though Newton's method converges quadratically, we would not expect Algorithm 4.1 to converge quadratically, since approximations are being made. From the proof, in particular (4.8), observe that if the pressure terms are small, then quadratic convergence of the velocity is essentially recovered. In our numerical tests, we find sometimes the INY method performs about the same as Newton, but sometimes exhibits linear convergence behavior. How well it performs is pressure dependent, γ-dependent, and dependent on the initial condition. INY dramatically outperforms Usual Picard and IPY on all of our tests.

Proof. We begin the proof with an induction hypothesis, that the sequence $\|u - u_k\| \leq \min\{\nu^{-1/2}(1-\tilde{\alpha})^{-1/2}, \epsilon\|\nabla u\|\}$ for all $k \in \mathbb{N}$, where ϵ is some positive constant. Hence for all k, we have

$$\|\nabla u_{k-1}\| \leq \|\nabla(u - u_{k-1})\| + \|\nabla u\| \leq (1+\epsilon)\nu^{-1}\|f\|_{-1} \tag{4.1}$$

by using the triangle inequality, and the a priori upper bound of true solution u. We further assume that

$$\nu(1-\tilde{\alpha})\|\nabla(u - u_{k-1})\|^2 + \gamma^{-1}\|p - p_{k-1}\|^2 \leq \|\nabla u\|^2,$$

and by proving that the sequence defined by $\nu(1-\tilde{\alpha})\|\nabla(u-u_{k-1})\|^2+\gamma^{-1}\|p-p_{k-1}\|^2$ is decreasing, this will imply the condition at the next iteration.

Denote $e_k^u := u - u_k$ and $e_k^z := u - z_k$. Subtracting Step k.1 of Algorithm 4.1 from the unique steady solution from (2.3)–(2.4), we obtain for all $v \in X_h$,

$$\begin{aligned}\gamma(\nabla \cdot e_k^z, \nabla \cdot v) + \nu(\nabla e_k^z, \nabla v) \\ = (p - p_{k-1}, \nabla \cdot v) - b^*(e_{k-1}^u, e_{k-1}^u, v) - b^*(e_k^z, u_{k-1}, v) - b^*(u_{k-1}, e_k^z, v).\end{aligned}$$

Choosing $v = e_k^z$ and applying upper bounds similar to the Picard case above, causes the last term to vanish and leaves

$$\gamma\|\nabla \cdot e_k^z\|^2 + \nu(1-\tilde{\alpha})\|\nabla e_k^z\|^2 \leq \gamma^{-1}\|p - p_{k-1}\|^2 + \frac{M^2}{\nu(1-\tilde{\alpha})}\|\nabla e_{k-1}^u\|^4 \tag{4.2}$$

thanks to the Young's inequality, (4.1), and the definition of $\tilde{\alpha}$.

Similarly, for Step k.3, we obtain

$$\gamma\|\nabla \cdot e_k^u\|^2 + \nu(1-\tilde{\alpha})\|\nabla e_k^u\|^2 \leq \gamma^{-1}\|p - p_k\|^2 + \frac{M^2}{\nu(1-\tilde{\alpha})}\|\nabla e_{k-1}^u\|^4. \tag{4.3}$$

Next, we give a bound of $\|p - p_k\|$. Begin by adding Steps k.1 and k.2, and subtracting it from the unique steady solution equation (2.3). Then we obtain the error equation for all $v \in X_h$,

$$\begin{aligned}\gamma(\nabla \cdot (z_k + w_k - u), \nabla \cdot v) + \nu(\nabla(z_k + w_k - u), \nabla v) \\ = (p_k - p, \nabla \cdot v) + b^*(e_{k-1}^u, e_{k-1}^u, v) + b^*(u_{k-1}, e_k^z, v) + b^*(e_k^z, u_{k-1}, v).\end{aligned} \tag{4.4}$$

Choosing $v = z_k + w_k - u \in V_h$ causes the pressure term to vanish and yields the bound

$$\gamma\|\nabla\cdot(z_k+w_k-u)\|^2+\nu\|\nabla(z_k+w_k-u)\|^2 \leq 8\nu\tilde{\alpha}^2\|\nabla e_k^z\|^2+2M^2\nu^{-1}\|\nabla e_{k-1}^u\|^4 \tag{4.5}$$

thanks to the Young's inequality, (4.1), and the definition of $\tilde{\alpha}$. Now applying the inf-sup condition to (4.4), we find

$$\beta\|p_k - p\| \leq \gamma\|\nabla \cdot (z_k + w_k - u)\| + \nu\|\nabla(z_k + w_k - u)\| + M\|\nabla e_{k-1}^u\|^2 + 2\nu\tilde{\alpha}\|\nabla e_k^z\|.$$

Squaring both sides, using that $\gamma \geq \nu$, and reducing yields

$$\beta^2\|p_k - p\|^2 \leq 4\gamma\left(\gamma\|\nabla\cdot(z_k+w_k-u)\|^2 + \nu\|\nabla(z_k+w_k-u)\|^2 \right. \\ \left. +\nu^{-1}M^2\|\nabla e_{k-1}^u\|^4 + 2\nu\tilde{\alpha}^2\|\nabla e_k^z\|^2\right).$$

Using the bound (4.5) reduces this estimate to

$$\|p-p_k\|^2 \leq 4\gamma\beta^{-2}\left(10\nu\tilde{\alpha}^2\|\nabla e_k^z\|^2 + 3\nu^{-1}M^2\|\nabla e_{k-1}^u\|^4\right), \tag{4.6}$$

and combining (4.2) and (4.6) and multliplying both sides by γ^{-1} produces

$$\gamma^{-1}\|p-p_k\|^2 \\ \leq 4\beta^{-2}\left(\frac{10\tilde{\alpha}^2}{1-\tilde{\alpha}}\gamma^{-1}\|p-p_{k-1}\|^2 + \nu^{-1}M^2\left(3+\frac{10\tilde{\alpha}^2}{(1-\tilde{\alpha})^2}\right)\|\nabla e_{k-1}^u\|^4\right). \tag{4.7}$$

Next combine the bound (4.3) with the pressure term bound (4.7) to find that

$$\gamma\|\nabla\cdot e_k^u\|^2 + \nu(1-\tilde{\alpha})\|\nabla e_k^u\|^2 \\ \leq \frac{40\beta^{-2}\tilde{\alpha}^2}{1-\tilde{\alpha}}\gamma^{-1}\|p-p_{k-1}\|^2 \\ + \frac{\nu^{-1}M^2}{(1-\tilde{\alpha})^2}\left(\nu^{-1}+12\beta^{-2}(1-\tilde{\alpha})^2+40\beta^{-2}\tilde{\alpha}^2\right)\|\nabla e_{k-1}^u\|^4,$$

and add this bound with (4.7) to get

$$\gamma^{-1}\|p-p_k\|^2 + \nu(1-\tilde{\alpha})\|\nabla e_k^u\|^2 \\ \leq \tilde{\alpha}\left(\frac{80\beta^{-2}\tilde{\alpha}}{1-\tilde{\alpha}}\gamma^{-1}\|p-p_{k-1}\|^2 \right. \\ \left. + \frac{\tilde{\alpha}^{-1}\nu^{-1}M^2}{(1-\tilde{\alpha})^2}\left(\nu^{-1}+24\beta^{-2}(1-\tilde{\alpha})^2+80\beta^{-2}\tilde{\alpha}^2\right)\|\nabla e_{k-1}^u\|^4\right).$$

Now applying the small data assumption that $\tilde{\alpha} < 1$, we obtain

$$\gamma^{-1}\|p-p_k\|^2 + \nu(1-\tilde{\alpha})\|\nabla e_k^u\|^2 \\ \leq \tilde{\alpha}\left(\gamma^{-1}\|p-p_{k-1}\|^2 \right. \\ \left. + \frac{\tilde{\alpha}^{-1}\nu^{-3}M^2}{(1-\tilde{\alpha})^4}\left(\nu^{-1}+24\beta^{-2}(1!-\tilde{\alpha})^2+80\beta^{-2}\tilde{\alpha}^2\right)\left(\nu(1-\tilde{\alpha})\|\nabla e_{k-1}^u\|^2\right)^2\right) \\ \leq \tilde{\alpha}\left(\gamma^{-1}\|p-p_{k-1}\|^2 \right. \\ \left. + \frac{\nu^3}{(1+\epsilon)^3M\|f\|_{-1}^3}\frac{\tilde{\alpha}^2}{(1-\tilde{\alpha})^4}\left(\nu^{-1}+24\beta^{-2}(1-\tilde{\alpha})^2+80\beta^{-2}\tilde{\alpha}^2\right)\left(\nu(1-\tilde{\alpha})\|\nabla e_{k-1}^u\|^2\right)^2\right) \\ \leq \tilde{\alpha}\left(\gamma^{-1}\|p-p_{k-1}\|^2 \right. \\ \left. + \frac{\nu^3(80^{-1}\beta^2)^2}{(1+\epsilon)^3M\|f\|_{-1}^3}\left(\frac{\nu^{-1}}{(1-\tilde{\alpha})^2}+24\beta^{-2}+80\beta^{-2}(80^{-1}\beta^2)^2\right)\left(\nu(1-\tilde{\alpha})\|\nabla e_{k-1}^u\|^2\right)^2\right) \\ \leq \tilde{\alpha}\left(\gamma^{-1}\|p-p_{k-1}\|^2 + \left(\nu(1-\tilde{\alpha})\|\nabla e_{k-1}^u\|^2\right)^2\right). \tag{4.8}$$

By the second assumption provided in the beginning, $\nu(1-\tilde{\alpha})\|\nabla(u-u_{k-1})\|^2 \leq 1$. We therefore have proven that $\nu(1-\tilde{\alpha})\|\nabla(u-u_k)\|^2 + \gamma^{-1}\|p-p_k\|^2$ is a contractive sequence in k, and thus converges. Since the solution of the problem (2.3)–(2.4)

is unique and bounded by the data, we have that the limit of the INY iteration converges to the solution of (2.3)–(2.4). □

4.2. **Numerical tests for INY.** We now test the INY method, and compare its behavior to that of Usual Newton. We repeat the use of test problems from the previous section on Picard solvers, and for simplicity use the same settings, parameters, and tolerances as used above for these problems unless stated otherwise.

4.2.1. *3D lid-driven cavity.* We repeat the test above for the 3D lid-driven cavity with Re=100, but now solve with INY and the Usual Newton method. The purpose of this test is to compare the ability of the two methods to solve the nonlinear problem.

All tolerances, parameters, and solver settings are the same as in the Picard tests above for this test problem. Statistics of the computations are shown in Table 4, for different element choices, and on varying meshes, and we make several observations from the table. First, the INY nonlinear iteration converges in the same (or 1 more) number of iterations as Usual Newton. A second observation is that the average number of outer iterations is smaller for INY compared to Usual Newton with either outer solver, but the computation time for each solve was significantly faster for INY. We note that the number of nonlinear iterations and Schur complement outer iterations were not significantly affected by h, and also that the differences between INY and Usual Newton were (relatively) the same for both Taylor-Hood and Scott-Vogelius elements.

TABLE 4. Re=100 3D driven cavity statistics for INY and Newton solvers, using various element choices and meshes.

(P_2, P_1) **Taylor-Hood elements (γ=1)**

	INY		Usual Newton		
dof	nonlin its	avg PCG its / sec	nonlin its	avg BICGSTAB its/sec	avg GMRES its/sec
1,093	6	16 / 0.02	5	15.5 / 0.03	26 / 0.03
10,637	5	14 / 0.36	5	16 / 0.70	25 / 0.60
38,229	6	12 / 2.47	5	16 / 4.84	26 / 4.46
93,469	6	12 / 8.93	5	16.5 / 23.34	25 / 21.26
185,957	6	12 / 25.93	5	18 / 81.46	27 / 72.51

(P_3, P_2^{disc}) **Scott-Vogelius elements** (γ=1)

	INY		Usual Newton		
dof	nonlin its	avg PCG its / sec	nonlin its	avg BICGSTAB its/sec	avg GMRES its/sec
702	6	8 / 0.01	4	8.5 / 0.01	13 / 0.01
16,770	5	7 / 0.08	5	8.5 / 0.18	14 / 0.18
206,874	6	7 / 1.74	5	9.5 / 4.03	15 / 4.04
796,722	6	7 / 11.16	5	9.5 / 26.77	16 / 25.94

4.3. **2D Bifurcation flow.** We now test INY in the same two-dimensional bifurcation geometry as in the previous section, restricting to the case of Re=200 ($\nu = 0.0133$). The purpose of this test is for an additional comparison of the various solvers, and values of γ, as well as to show that INY need not always provide quadratic convergence. Our view is that it is critical to know the drawbacks of algorithms, as well as their strengths, and our proof shows that it may behave

TABLE 5. Iteration statistics for a 2D bifurcation test, with Usual Newton using the linear solver from [19].

INY 2D Bifurcation Test, $\nu = .0133$ (Re=200)

	INY		Usual Newton	
γ	nonlin its	avg PCG its / sec	nonlin its	avg GMRES its / sec
1	53	12 / 0.26	7	20 / 0.39
3	34	11 / 0.26	7	16 / 0.31
5	26	11 / 0.26	7	14 / 0.28
10	18	11 / 0.25	7	14 / 0.27
20	14	11 / 0.25	7	14 / 0.27

quadratically depending on the pressure convergence, and in this test example the pressure is not small relative to the velocity.

Referring to Table 5, we see that for $\gamma = 1$, INY converges only slightly faster than IPY and Usual Picard. As γ increases, this convergence improves significantly. For γ=20, INY still converges more slowly than Usual Newton, though the difference becomes much less pronounced. This is not surprising since the INY iteration's linear error term (the pressure term) in the analysis is scaled by γ^{-1}, so as γ increases this term's effect is reduced. This test suggests that for problems with large pressure, or bad initial guesses at pressure, INY may not perform much better than IPY unless γ is taken relatively large. We reiterate that high values of γ can lead to numerical difficulties, potentially limiting the applicability of INY for some problems.

5. Conclusions

We have developed new, simple, easy to implement, and efficient nonlinear solvers for the steady NSE: the IPY and INY. These methods are created as approximations of Picard and Newton methods, respectively, but use the same SPD Schur complement at each iteration. This makes them more easily computable than the typical methods, which need to resolve a different, nonsymmetric Schur complement at each iteration. Moreover, the SPD Schur complement can effectively use CG for both inner and outer solver, and the outer solver is easily preconditioned with the pressure mass matrix. This makes the method easy to implement as well as efficient. We performed several comparisons of various linear algebraic solvers and preconditioners for solving the linear systems of the proposed methods, and compared them to various linear algebraic solvers and preconditioners for linear systems arising in Usual Picard and Usual Newton: we found overall a significant efficiency gain can be expected by using the proposed methods (although we acknowledge there are very many other interesting and relevant tests of various linear algebraic solvers that could also have been done; we have tried to give a fair assessment using some common linear solving methods).

We have provided theory for both methods that proves they converge, provided conditions on the data. The small data conditions are somewhat worse than for the usual methods, as it includes the inf-sup constant in the IPY and INY results. However, it seems unavoidable for these additional terms to arise in the analysis, since Yosida splitting alters the discrete pressure equation. For both methods, we have shown results of extensive numerical experiments that show good convergence behavior to the correct solution: IPY typically has the same or similar convergence

behavior as a nonlinear iteration as Usual Picard, while INY compares well with Usual Newton when pressure is small or γ is larger (otherwise, INY may only converge linearly). No lack of robustness of the new methods is observed compared to the usual methods, which the theory suggests might be possible due to the more restrictive data conditions in the proofs.

There are at least two immediate directions for future work. First, one can try to apply these ideas to multiphysics problems such as MHD, which have a block saddle point structure. Second, one can further study the robustness of the methods, to determine analytically and/or numerically, how the various methods behave for higher Reynolds numbers.

References

[1] Aneurisk-Team. AneuriskWeb project website, http://ecm2.mathcs.emory.edu/aneuriskweb. Web Site, 2012.

[2] D. Arnold and J. Qin, *Quadratic velocity/linear pressure Stokes elements*, In R. Vichnevetsky, D. Knight, and G. Richter, editors, Advances in Computer Methods for Partial Differential Equations VII, pages 28–34. IMACS, 1992.

[3] S. Badia, A. Quaini, and A. Quarteroni, *Splitting methods based on algebraic factorization for fluid-structure interaction*, SIAM J. Sci. Comput. **30** (2008), no. 4, 1778–1805, DOI 10.1137/070680497. MR2407141

[4] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numer. **14** (2005), 1–137, DOI 10.1017/S0962492904000212. MR2168342

[5] M. Benzi and M. A. Olshanskii, *An augmented Lagrangian-based approach to the Oseen problem*, SIAM J. Sci. Comput. **28** (2006), no. 6, 2095–2113, DOI 10.1137/050646421. MR2272253

[6] S. Börm and S. Le Borne, *$\mathcal{H}$-LU factorization in preconditioners for augmented Lagrangian and grad-div stabilized saddle point systems*, Internat. J. Numer. Methods Fluids **68** (2012), no. 1, 83–98, DOI 10.1002/fld.2495. MR2874191

[7] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods*, Evolution to complex geometries and applications to fluid dynamics, Scientific Computation, Springer, Berlin, 2007. MR2340254

[8] M. A. Castro, J. R. Cebral, and C. M. Putman, *Computational fluid dynamics modeling of intracranial aneurysms: Effects of parent artery segmentation on intra-aneurysmal hemodynamics*, American Journal of Neuroradiology, 27:1703–1709, 2006.

[9] J. R. Cebral, M. A. Castro, C. M. Putman, and N. Alperin, *Flow-area relationship in internal carotid and vertebral arteries*, Physiological Measurements, 29:585–594, 2008.

[10] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics*, 2nd ed., Numerical Mathematics and Scientific Computation, Oxford University Press, Oxford, 2014. MR3235759

[11] R. S. Falk and M. Neilan, *Stokes complexes and the construction of stable finite elements with pointwise mass conservation*, SIAM J. Numer. Anal. **51** (2013), no. 2, 1308–1326, DOI 10.1137/120888132. MR3045658

[12] K. J. Galvin, A. Linke, L. G. Rebholz, and N. E. Wilson, *Stabilizing poor mass conservation in incompressible flow problems with large irrotational forcing and application to thermal convection*, Comput. Methods Appl. Mech. Engrg. **237/240** (2012), 166–176, DOI 10.1016/j.cma.2012.05.008. MR2947652

[13] P. Gervasio, F. Saleri, and A. Veneziani, *Algebraic fractional-step schemes with spectral methods for the incompressible Navier-Stokes equations*, J. Comput. Phys. **214** (2006), no. 1, 347–365, DOI 10.1016/j.jcp.2005.09.018. MR2208683

[14] V. Girault and P.-A. Raviart, *Finite Element Methods for Navier-Stokes Equations*, Theory and algorithms, Springer Series in Computational Mathematics, vol. 5, Springer-Verlag, Berlin, 1986. MR851383

[15] M. Gunzburger, *Iterative penalty methods for the Stokes and Navier-Stokes equations*, Proceedings from Finite Element Analysis in Fluids conference, University of Alabama, Huntsville, pages 1040–1045, 1989.

[16] J. Guzmán and M. Neilan, *Conforming and divergence-free Stokes elements on general triangular meshes*, Math. Comp. **83** (2014), no. 285, 15–36, DOI 10.1090/S0025-5718-2013-02753-6. MR3120580

[17] Y. He and A. Wang, *A simplified two-level method for the steady Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg. **197** (2008), no. 17-18, 1568–1576, DOI 10.1016/j.cma.2007.11.032. MR2396857

[18] F. Hecht, *New development in freefem++*, J. Numer. Math. **20** (2012), no. 3-4, 251–265, DOI 10.1515/jnum-2012-0013. MR3043640

[19] T. Heister and G. Rapin, *Efficient augmented Lagrangian-type preconditioning for the Oseen problem using grad-div stabilization*, Internat. J. Numer. Methods Fluids **71** (2013), no. 1, 118–134, DOI 10.1002/fld.3654. MR3001218

[20] J. G. Heywood, R. Rannacher, and S. Turek, *Artificial boundaries and flux and pressure conditions for the incompressible Navier-Stokes equations*, Internat. J. Numer. Methods Fluids **22** (1996), no. 5, 325–352, DOI 10.1002/(SICI)1097-0363(19960315)22:5⟨325::AID-FLD307⟩3.0.CO;2-Y. MR1380844

[21] E. W. Jenkins, V. John, A. Linke, and L. G. Rebholz, *On the parameter choice in grad-div stabilization for the Stokes equations*, Adv. Comput. Math. **40** (2014), no. 2, 491–516, DOI 10.1007/s10444-013-9316-1. MR3194715

[22] V. John, P. Knobloch, and J. Novo, *Finite elements for scalar convection-dominated equations and incompressible flow problems: a never ending story*? Computing and Visualization in Science, in press, 2018.

[23] V. John, A. Linke, C. Merdon, M. Neilan, and L. G. Rebholz, *On the divergence constraint in mixed finite element methods for incompressible flows*, SIAM Rev. **59** (2017), no. 3, 492–544, DOI 10.1137/15M1047696. MR3683678

[24] W. Layton, *A two-level discretization method for the Navier-Stokes equations*, Comput. Math. Appl. **26** (1993), no. 2, 33–38, DOI 10.1016/0898-1221(93)90318-P. MR1220955

[25] W. Layton, *Introduction to the Numerical Analysis of Incompressible Viscous Flows*, With a foreword by Max Gunzburger, Computational Science & Engineering, vol. 6, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008. MR2442411

[26] W. Layton, H. K. Lee, and J. Peterson, *Numerical solution of the stationary Navier-Stokes equations using a multilevel finite element method*, SIAM J. Sci. Comput. **20** (1998), no. 1, 1–12, DOI 10.1137/S1064827596306045. MR1639086

[27] W. Layton and H. W. J. Lenferink, *A multilevel mesh independence principle for the Navier-Stokes equations*, SIAM J. Numer. Anal. **33** (1996), no. 1, 17–30, DOI 10.1137/0733002. MR1377241

[28] M. A. Olshanskii and A. Reusken, *Grad-div stabilization for Stokes equations*, Math. Comp. **73** (2004), no. 248, 1699–1718, DOI 10.1090/S0025-5718-03-01629-6. MR2059732

[29] J. B. Perot, *An analysis of the fractional step method*, J. Comput. Phys. **108** (1993), no. 1, 51–58, DOI 10.1006/jcph.1993.1162. MR1239968

[30] A. Quaini and A. Quarteroni, *A semi-implicit approach for fluid-structure interaction based on an algebraic fractional step method*, Math. Models Methods Appl. Sci. **17** (2007), no. 6, 957–983, DOI 10.1142/S0218202507002170. MR2334549

[31] A. Quarteroni, F. Saleri, and A. Veneziani, *Analysis of the Yosida method for the incompressible Navier-Stokes equations*, J. Math. Pures Appl. (9) **78** (1999), no. 5, 473–503, DOI 10.1016/S0021-7824(99)00027-6. MR1697039

[32] A. Quarteroni, F. Saleri, and A. Veneziani, *Factorization methods for the numerical approximation of Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg. **188** (2000), no. 1-3, 505–526, DOI 10.1016/S0045-7825(99)00192-9. MR1774169

[33] L. G. Rebholz and M. Xiao, *On reducing the splitting error in Yosida methods for the Navier-Stokes equations with grad-div stabilization*, Comput. Methods Appl. Mech. Engrg. **294** (2015), 259–277, DOI 10.1016/j.cma.2015.06.013. MR3373449

[34] L. G. Rebholz and M. Xiao, *Improved accuracy in algebraic splitting methods for Navier-Stokes equations*, SIAM J. Sci. Comput. **39** (2017), no. 4, A1489–A1513, DOI 10.1137/16M1061424. MR3686807

[35] F. Saleri and A. Veneziani, *Pressure correction algebraic splitting methods for the incompressible Navier-Stokes equations*, SIAM J. Numer. Anal. **43** (2005), no. 1, 174–194, DOI 10.1137/S0036142903435429. MR2177140

[36] J. Schöberl, *Multigrid methods for a parameter dependent problem in primal variables*, Numer. Math. **84** (1999), no. 1, 97–119, DOI 10.1007/s002110050465. MR1724348
[37] R. Temam, *Navier-Stokes Equations*, Elsevier, North-Holland, 1991.
[38] A. Viguerie and A. Veneziani, *Algebraic splitting methods for the steady incompressible Navier-Stokes equations at moderate Reynolds numbers*, Comput. Methods Appl. Mech. Engrg. **330** (2018), 271–291, DOI 10.1016/j.cma.2017.10.028. MR3759096
[39] K. L. Wong and A. J. Baker, *A 3d incompressible Navier-Stokes velocity-vorticity weak form finite element algorithm*, International Journal for Numerical Methods in Fluids, 38:99–123, 2002.
[40] S. Zhang, *A new family of stable mixed finite elements for the 3D Stokes equations*, Math. Comp. **74** (2005), no. 250, 543–554, DOI 10.1090/S0025-5718-04-01711-9. MR2114637

Department of Mathematical Sciences, Clemson University, Clemson, South Carolina 29634
Email address: rebholz@clemson.edu

Department of Mathematics and Computer Science, Emory University, Atlanta, Georgia 30322
Current address: University of Pavia, Via Ferrata 3, Pavia, Italy, 27100
Email address: alexander.viguerie@unipv.it

Department of Mathematical Sciences, Clemson University, Clemson, South Carolina 29634
Current address: Department of Mathematics, College of William & Mary, Williamsburg, Virginia 23185
Email address: mxiao01@wm.edu