INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2003; **43**:1407–1427 (DOI: 10.1002/fld.627)

Efficient solution of the steady-state Navier–Stokes equations using a multigrid preconditioned Newton–Krylov method

Syamsudhuha[†] and David J. Silvester[*,‡]

Department of Mathematics, UMIST, Manchester M60 1QD, U.K.

SUMMARY

An inexact Newton's method is used to solve the steady-state incompressible Navier–Stokes equations. The equations are discretized using a mixed finite element approximation. A new efficient preconditioning methodology introduced by Kay *et al.* (*SIAM J. Sci. Comput.*, 2002; **24**:237–256) is applied and its effectiveness in the context of a Newton linearization is investigated. The original strategy was introduced as a preconditioning methodology for discrete Oseen equations that arise from Picard linearization. Our new variant of the preconditioning strategy is constructed from building blocks consisting of two component multigrid cycles; a multigrid *V*-cycle for a scalar convection–diffusion operator; and a multigrid *V*-cycle for a pressure Poisson operator. We present numerical experiments showing that the convergence rate of the preconditioned GMRES is independent of the grid size and relatively insensitive to the Reynolds number. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: Navier–Stokes; multigrid; Krylov; non-linear; Newton

1. INTRODUCTION

The incompressible Navier–Stokes equations form one of the most important models in Computational Fluid Dynamics. The development of numerical solution methods is accordingly a very active field of research. Non-linear solution methods such as Newton-like methods have been widely used. We give three examples here.

Engelman [1], applied quasi-Newton methods. These methods are compared with the commonly employed successive substitution and (exact) Newton procedures. Ghia *et al.* [2] considered a non-linear multigrid method to solve a 2-D driven cavity problem in a square domain with Reynolds number as high as 10 000 and meshes consisting of as many as 257×257 points. An inexact Newton's method was used by McHugh and Knoll in Reference [3]. Two conjugate gradient-like algorithms, the transpose-free quasi minimal residual algorithm (TFQMR) and the conjugate gradient squared algorithm (CGS) preconditioned by incomplete

[*]Correspondence to: D. J. Silvester, Department of Mathematics, UMIST, Manchester M60 1QD, U.K.
[‡]E-mail: d.j.silvester@umist.ac.uk
[†]Supported by DUE-Project Universitas Riau, Indonesia.

Received 27 December 2002
Revised 30 June 2003

lower–upper (ILU) factorizations are used to solve the linear systems arising on each Newton iteration. Subsequently, Knoll and Rider [4] used the 2-D driven cavity as a test problem and solved it using the Newton–Krylov algorithm. In particular, the application of multigrid as a preconditioner of a matrix-free Newton–Krylov method was studied.

In this paper we consider the numerical implementation of an efficient preconditioning strategy for the linearized incompressible Navier–Stokes equations. The method builds upon the preconditioning methodology originally proposed by Kay and Loghin [5] for the Oseen equations arising from Picard linearization, and subsequently analysed by Silvester *et al.* [6] and Elman *et al.* [7]. The strategy has two basic building blocks; a multigrid V-cycle for a scalar convection–diffusion operator, and a multigrid V-cycle for a pressure Poisson operator. Our aim here is to study the performance of this preconditioning methodology for the solution of a linear system involving the Jacobian matrix resulting from a Newton linearization.

We consider a standard finite element discretization of the Navier–Stokes equations. We employ an inexact Newton method to solve the non-linear equations. The linearized system is solved by the right preconditioned GMRES method. We shall restrict our implementation to the stationary case in two space dimensions. Our observations are that the number of iterations for convergence is independent of the mesh size, but slowly increase as Reynolds number is increased.

The outline of this paper is as follows. In the remainder of this section we present the standard weak formulation of the steady-state Navier–Stokes equations. Non-linear iteration methods and our preconditioning strategy will be described in Section 2. Section 3 will present some computational experiments demonstrating the computational efficiency of our preconditioning approach, and illustrating the effect of different choices of the forcing term on the convergence of the inexact Newton method. We go on to compare results with those published by Ghia *et al.* [2] and Knoll *et al.* [4], in Section 4. In the last section, we give some conclusions.

1.1. The stationary Navier–Stokes equations

We consider the steady-state incompressible Navier–Stokes equations written in a primitive variable formulation in terms of the velocity $\boldsymbol{u}$ and the pressure p:

$$-\varepsilon \triangle \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \nabla p = \boldsymbol{f} \quad \text{in } \Omega \tag{1}$$

$$\nabla \cdot \boldsymbol{u} = 0 \quad \text{in } \Omega \tag{2}$$

$$\boldsymbol{u} = \boldsymbol{g} \quad \text{on } \Gamma \tag{3}$$

where the scalar $\varepsilon > 0$ is a 'viscosity' parameter, which in a non-dimensional setting is inversely proportional to the *Reynolds* number $Re = 1/\varepsilon$. $\boldsymbol{f}$ is the given body force per unit mass; also $\boldsymbol{g}$ is a given function such that $\int_\Gamma \boldsymbol{g} \cdot \boldsymbol{n} \, \mathrm{d}\Gamma = 0$, where $\boldsymbol{n}$ is the unit vector normal to the boundary Γ.

1.1.1. Weak formulation. We first consider the case of the homogeneous Dirichlet boundary condition

$$\boldsymbol{u} = \boldsymbol{0} \tag{4}$$

Having $\mathbf{H}_0^1(\Omega)$ (a Sobolev space of vector valued function with square integrable derivatives of order up to one and $L_0^2(\Omega)$ (the standard space of real valued functions that are square integrable over Ω) as the velocity and the pressure test spaces, the standard weak formulation of (1), (2) and (4) is obtained by multiplying Eq. (1) by a vector test function and integrating over the flow domain. Similarly, the continuity equation or incompressibility constraint (2) is multiplied by a scalar test function and integrated over the domain. Given $\boldsymbol{f} \in \mathbf{H}^{-1}(\Omega)$, we seek $\boldsymbol{u} \in \mathbf{H}_0^1(\Omega)$ and $p \in L_0^2(\Omega)$ satisfying

$$a_0(\boldsymbol{u}, \boldsymbol{v}) + a_1(\boldsymbol{u}, \boldsymbol{u}, \boldsymbol{v}) + b(\boldsymbol{v}, p) = (\boldsymbol{f}, \boldsymbol{v}) \quad \forall \boldsymbol{v} \in \mathbf{H}_0^1(\Omega) \tag{5}$$

$$b(\boldsymbol{u}, q) = 0 \qquad \forall q \in L_0^2(\Omega) \tag{6}$$

where

$$\begin{aligned} a_0(\boldsymbol{u}, \boldsymbol{v}) &= \varepsilon \int_\Omega \nabla \boldsymbol{u} \cdot \nabla \boldsymbol{v} \, \mathrm{d}\Omega \qquad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbf{H}_0^1(\Omega) \\ a_1(\boldsymbol{w}, \boldsymbol{u}, \boldsymbol{v}) &= \int_\Omega \sum_{i,j=1}^{d} w_j \frac{\partial u_i}{\partial x_j} v_i \, \mathrm{d}\Omega \quad \forall \boldsymbol{w}, \boldsymbol{u}, \boldsymbol{v} \in \mathbf{H}_0^1(\Omega) \\ b(\boldsymbol{v}, q) &= -\int_\Omega q \nabla \cdot \boldsymbol{v} \, \mathrm{d}\Omega \qquad \forall \boldsymbol{v} \in \mathbf{H}_0^1(\Omega) \text{ and } \forall q \in L_0^2(\Omega) \end{aligned} \tag{7}$$

Furthermore, defining

$$a(\boldsymbol{w}, \boldsymbol{u}, \boldsymbol{v}) = a_0(\boldsymbol{u}, \boldsymbol{v}) + a_1(\boldsymbol{w}, \boldsymbol{u}, \boldsymbol{v}) \tag{8}$$

the weak formulation (5) and (6) can be written in a simpler form: find $\boldsymbol{u} \in \mathbf{H}_0^1(\Omega)$ and $p \in L_0^2(\Omega)$ such that

$$a(\boldsymbol{u}, \boldsymbol{u}, \boldsymbol{v}) + b(\boldsymbol{v}, p) = (\boldsymbol{f}, \boldsymbol{v}) \quad \forall \boldsymbol{v} \in \mathbf{H}_0^1(\Omega) \tag{9}$$

$$b(\boldsymbol{u}, q) = 0 \qquad \forall q \in L_0^2(\Omega) \tag{10}$$

For the general case of a non-homogeneous Dirichlet boundary condition (3), we introduce $\boldsymbol{u}_* \in \mathbf{H}^1$ such that $\boldsymbol{u}_* = \boldsymbol{g}$ on Γ, then the standard weak formulation of (1)–(3) is defined as follows: Given $\boldsymbol{f} \in \mathbf{H}^{-1}(\Omega)$ and $\boldsymbol{g} \in \mathbf{H}^{1/2}(\Gamma)$, find $\boldsymbol{u} \in \mathbf{H}^1(\Omega)$ such that $\boldsymbol{u} - \boldsymbol{u}^* \in \mathbf{H}_0^1(\Omega)$ and $p \in L_0^2(\Omega)$ satisfying

$$a(\boldsymbol{u}, \boldsymbol{u}, \boldsymbol{v}) + b(\boldsymbol{v}, p) = (\boldsymbol{f}, \boldsymbol{v}) \quad \forall \boldsymbol{v} \in \mathbf{H}_0^1(\Omega) \tag{11}$$

$$b(\boldsymbol{u}, q) = 0 \qquad \forall q \in L_0^2(\Omega) \tag{12}$$

Full details concerning weak formulations of the Navier–Stokes system may be found in Reference [8, pp. 284–362].

1.1.2. Finite element discretization. To generate a discrete system, we use the standard finite element discretization of (5)–(6) by choosing finite element subspaces $\mathbf{X}_0^h \subset \mathbf{H}_0^1(\Omega)$ and

$M^h \subset L_0^2(\Omega)$, where h is a representative mesh parameter, and seek $\boldsymbol{u}^h \in \mathbf{X}_0^h$ and $p^h \in M^h$ satisfying

$$a(\boldsymbol{u}^h, \boldsymbol{u}^h, \boldsymbol{v}^h) + b(\boldsymbol{v}^h, p^h) = (\boldsymbol{f}, \boldsymbol{v}^h) \quad \forall \boldsymbol{v}^h \in \mathbf{X}_0^h \tag{13}$$

$$b(\boldsymbol{u}^h, q^h) = 0 \qquad \forall q^h \in M^h \tag{14}$$

The stability and optimal accuracy of this discrete approximation is governed by the following five conditions. The first three of these are *continuity* conditions, as follows:

$$|a_0(\boldsymbol{u}^h, \boldsymbol{v}^h)| \leqslant \kappa_1 \|\boldsymbol{u}^h\|_1 \|\boldsymbol{v}^h\|_1 \qquad \boldsymbol{u}^h, \boldsymbol{v}^h \in \mathbf{X}_0^h$$

$$|b(\boldsymbol{v}^h, q^h)| \leqslant \kappa_2 \|\boldsymbol{v}^h\|_1 \|q^h\|_0 \qquad \boldsymbol{v}^h \in \mathbf{X}_0^h, \ q^h \in M^h$$

$$|a_1(\boldsymbol{w}^h, \boldsymbol{u}^h, \boldsymbol{v}^h)| \leqslant \kappa_3 \|\boldsymbol{w}^h\|_1 \|\boldsymbol{u}^h\|_1 \|\boldsymbol{v}^h\|_1 \qquad \boldsymbol{w}^h, \boldsymbol{u}^h, \boldsymbol{v}^h \in \mathbf{X}_0^h$$

where κ_1, κ_2 and κ_3 are constants whose values are independent of h.

The fourth condition is a *coercivity* condition

$$a_0(\boldsymbol{z}^h, \boldsymbol{z}^h) \geqslant \gamma_a |\boldsymbol{z}^h|_1^2 \quad \forall \boldsymbol{z}^h \in \mathbf{Z}^h$$

where

$$\mathbf{Z}^h = \{\boldsymbol{v}^h \in \mathbf{X}^h \mid b(\boldsymbol{v}^h, q^h) = 0 \ \forall q^h \in M^h\}$$

and γ_a is a constant whose value is independent of h.

The fifth condition is known as the inf–sup condition and is given by

$$\sup_{\boldsymbol{v}^h \in \mathbf{X}_0^h} \frac{b(\boldsymbol{v}^h, q^h)}{|\boldsymbol{v}^h|_1} \geqslant \gamma_b \|q^h\|_0 \quad \forall q^h \in M^h \tag{15}$$

where γ_b is a constant whose value is independent of h. By assuming that the chosen finite element spaces satisfy the inf–sup condition, the existence and uniqueness of a solution to (13)–(14), for sufficiently small $Re = 1/\varepsilon$, follows. See Reference [8, pp. 279–283] for details.

1.2. Newton linearization

Once the finite element spaces are prescribed, the discrete problem (13)–(14) reduces to solving a system of non-linear algebraic equations which can be solved by iteration. A simple iterative technique to solve this non-linear problem can be defined as follows: choose $\boldsymbol{u}_0^h \in \mathbf{X}_0^h$; for every $k \geqslant 1$, if $\boldsymbol{u}_{k-1}^h \in \mathbf{X}_0^h$ is known, find $\{\boldsymbol{u}_k^h, p_k^h\}$ such that

$$\begin{aligned} a(\boldsymbol{u}_{k-1}^h, \boldsymbol{u}_k^h, \boldsymbol{v}^h) + b(p_k^h, \boldsymbol{v}^h) &= (\boldsymbol{f}, \boldsymbol{v}^h) \quad \forall \boldsymbol{v}^h \in \mathbf{X}_0^h \\ b(\boldsymbol{u}_k^h, q^h) &= 0 \qquad \forall q^h \in M^h \end{aligned} \tag{16}$$

In this Picard iteration method, the velocity computed in the preceding iteration is substituted into the convective term. Hence, the Navier–Stokes equations are linearized such that the coefficient matrix is changed at each step.

For each k this discretization leads to a system of linear equations. To see this, let $\Omega \in \mathbb{R}^2$ and let $\{\phi_i\}$, $i=1,\ldots,m$, and $\{(\psi_j,0),(0,\psi_j)\}$, $j=1,\ldots,n$ be the basis for the pressure and velocity trial spaces M^h and $\mathbf{X}^h$, respectively. Then

$$p_k^h = \sum_{i=1}^{m} \alpha_i^k \phi_i \quad \text{and} \quad \boldsymbol{u}_k^h = (u_1^k, u_2^k) = \left(\sum_{j=1}^{n} \beta_j^k \psi_j, \sum_{j=n+1}^{2n} \beta_j^k \psi_{j-n} \right) \tag{17}$$

for some constants α_i^k, $i=1,2,\ldots,m$, and β_j^k, $j=1,2,\ldots,2n$. Hence, the approximate problem is equivalent to the linear algebraic system of $(2n+m)\times(2n+m)$ matrix system

$$\begin{pmatrix} A^k & B^{\mathrm{T}} \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}^k \\ \mathbf{p}^k \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix} \tag{18}$$

Here $A^k \in \mathbb{R}^{2n\times 2n}$ is the vector valued discrete convection–diffusion operator with entries

$$[A]_{i,j}^k = \begin{Bmatrix} a(\boldsymbol{u}_{k-1}, \psi_j, \psi_i) & 1 \leqslant i,j \leqslant n \\ a(\boldsymbol{u}_{k-1}, \psi_{j-n}, \psi_{i-n}) & n+1 \leqslant i,j \leqslant 2n \\ 0 & \text{otherwise} \end{Bmatrix} \tag{19}$$

The matrix $B \in \mathbb{R}^{m\times 2n}$ is the divergence matrix defined by

$$[B]_{i,j} = \begin{Bmatrix} -\displaystyle\int_\Omega \frac{\partial \psi_j}{\partial x}\, \phi_i \, \mathrm{d}\Omega & 1 \leqslant i \leqslant m \quad 1 \leqslant j \leqslant n \\ -\displaystyle\int_\Omega \frac{\partial \psi_{j-n}}{\partial y}\, \phi_i \, \mathrm{d}\Omega & 1 \leqslant i \leqslant m \quad n+1 \leqslant j \leqslant 2n \end{Bmatrix} \tag{20}$$

The first block of the right hand side vector is simply the $2n$-vector given by

$$[\mathbf{f}]_i = \begin{Bmatrix} (f_x, \psi_i) & 1 \leqslant i \leqslant n, \\ (f_y, \psi_{i-n}) & n+1 \leqslant i \leqslant 2n \end{Bmatrix} + [\mathbf{f}_\Gamma]_i \tag{21}$$

where the vector $\mathbf{f}_\Gamma$ arises from non-zero Dirichlet conditions on the velocity. The vectors $\mathbf{u}^k \in \mathbb{R}^{2n}$ and $\mathbf{p}^k \in \mathbb{R}^m$ are the coefficients of the discrete velocity and pressure respectively. Furthermore, the vector $\mathbf{g}$ also arises from any inhomogeneous boundary conditions on the velocity.

Newton's method, for the discrete system (13)–(14) is defined as follows. We choose an initial guess $\boldsymbol{u}_0^h$ such that $\boldsymbol{u}_0^h|_\Gamma = \boldsymbol{g}^h$, where $\boldsymbol{g}^h$ is an approximation to $\boldsymbol{g}$ on Γ, and generate the sequence $\{\boldsymbol{u}_k^h, p_k^h\}$ for $k=1,2,\ldots$, by solving the linear problem

$$\begin{aligned} a_N(\boldsymbol{u}_{k-1}^h, \boldsymbol{u}_k^h, \boldsymbol{v}^h) + b(p_k^h, \boldsymbol{u}^h) &= (\boldsymbol{f}, \boldsymbol{v}^h) + a_1(\boldsymbol{u}_{k-1}^h, \boldsymbol{u}_{k-1}^h, \boldsymbol{v}^h) \quad \forall \boldsymbol{v}^h \in \mathbf{X}_0^h \\ b(\boldsymbol{u}_k^h, q^h) &= 0 \qquad\qquad\qquad\qquad\qquad\qquad\quad \forall q^h \in M^h \end{aligned} \tag{22}$$

where

$$a_N(\boldsymbol{w}^h, \boldsymbol{u}_k^h, \boldsymbol{v}^h) = a(\boldsymbol{w}^h, \boldsymbol{u}_k^h, \boldsymbol{v}^h) + a_1(\boldsymbol{u}_k^h, \boldsymbol{w}^h, \boldsymbol{v}^h) \tag{23}$$

Using the same basis functions as in (17), the corresponding linear system will be

$$\begin{pmatrix} A_N^k & B^{\mathrm{T}} \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}^k \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_N^k \\ \mathbf{g} \end{pmatrix} \tag{24}$$

where $A_N^k = A^k + N^k$. The matrix $N^k \in \mathbb{R}^{2n \times 2n}$ and vector $\mathbf{f}_N^k$ arise from the $a_1(\boldsymbol{u}_k^h, \boldsymbol{u}_{k-1}^h, \boldsymbol{v}^h)$ and $(\boldsymbol{f}, \boldsymbol{v}^h) + a_1(\boldsymbol{u}_{k-1}^h, \boldsymbol{u}_{k-1}^h, \boldsymbol{v}^h)$ terms, respectively. Here the entries of N^k and $\mathbf{f}_N^k$ can be written as

$$[N]_{i,j} = \begin{cases} \displaystyle\int_\Omega \psi_j \frac{\partial u_{x_{k-1}}}{\partial x} \psi_i \, \mathrm{d}\Omega & 1 \leqslant i, j \leqslant n \\ \displaystyle\int_\Omega \psi_{j-n} \frac{\partial u_{x_{k-1}}}{\partial y} \psi_i \, \mathrm{d}\Omega & 1 \leqslant i \leqslant n \qquad n+1 \leqslant j \leqslant 2n \\ \displaystyle\int_\Omega \psi_j \frac{\partial u_{y_{k-1}}}{\partial x} \psi_{i-n} \, \mathrm{d}\Omega & 1 \leqslant j \leqslant n \qquad n+1 \leqslant i \leqslant 2n \\ \displaystyle\int_\Omega \psi_{j-n} \frac{\partial u_{y_{k-1}}}{\partial y} \psi_{i-n} \, \mathrm{d}\Omega & n+1 \leqslant i, j \leqslant 2n \end{cases} \tag{25}$$

and

$$[\mathbf{f}_N]_i = \begin{cases} [\mathbf{f}]_i + \displaystyle\int_\Omega \left(u_{x_{k-1}} \frac{\partial u_{x_{k-1}}}{\partial x} + u_{y_{k-1}} \frac{\partial u_{x_{k-1}}}{\partial y} \right) \psi_i \, \mathrm{d}\Omega & 1 \leqslant i \leqslant n \\ [\mathbf{f}]_{i-n} + \displaystyle\int_\Omega \left(u_{x_{k-1}} \frac{\partial u_{y_{k-1}}}{\partial x} + u_{y_{k-1}} \frac{\partial u_{y_{k-1}}}{\partial y} \right) \psi_{i-n} \, \mathrm{d}\Omega & n+1 \leqslant i \leqslant 2n \end{cases} \tag{26}$$

respectively. The convection–diffusion matrix A^k and the divergence matrix B are the same as in (19) and (20), respectively.

The main advantage of Newton's method (22) is that it will be locally and quadratically convergent whenever the initial guess $\boldsymbol{u}_0^h$ for the velocity is sufficiently close to a branch of non-singular solutions (see Reference [8, pp. 362–367]). For this initial guess, we typically use a mesh sequencing procedure.

Moreover, to describe this mesh sequencing algorithm, let h_j, $j = 0, 1, \ldots, J$ be a sequence of mesh spacings, such that $h_0 > h_1 > \cdots > h_J$, with the corresponding finite element spaces $\mathbf{X}^{h_j}(\Omega)$, $M^{h_j}(\Omega)$, where

$$\mathbf{X}^{h_0}(\Omega) \subset \mathbf{X}^{h_1}(\Omega) \subset \cdots \subset \mathbf{X}^{h_J}(\Omega)$$

and

$$M^{h_0}(\Omega) \subset M^{h_1}(\Omega) \subset \cdots \subset M^{h_J}(\Omega)$$

then the algorithm can be described as follows.

Algorithm 1.1 (*Mesh sequencing algorithm*)
For mesh level $j = 0, 1, \ldots, J$ solve the following linearized problems by performing N_j Newton steps at each level.

1. *Set the initial solution*
   (a) *If mesh level $j = 0$, then let $\boldsymbol{u}_0^{h_0}$ and $p_0^{h_0}$ be given.*

(b) *else*
set $\boldsymbol{u}_0^{h_j} = \boldsymbol{u}^{h_{j-1}}$ *and* $p_0^{h_j} = p^{h_{j-1}}$.

2. *For* $k = 1, 2, \ldots, N_j$ *solve the linear problem.*
Find $\boldsymbol{u}_k^{h_j} \in \mathbf{X}^{h_j}$ *such that* $\boldsymbol{u}_k^{h_j}|_\Gamma = \boldsymbol{g}^{h_j}$ *and* $p_k^{h_j} \in M^{h_j}$ *satisfying*

$$a_N(\boldsymbol{u}_{k-1}^{h_j}, \boldsymbol{u}_k^{h_j}, \boldsymbol{v}^h) + b(p_k^{h_j}, \boldsymbol{v}^h) = (\boldsymbol{f}, \boldsymbol{v}^h) + a_1(\boldsymbol{u}_{k-1}^{h_j}, \boldsymbol{u}_{k-1}^{h_j}, \mathbf{v}^h) \quad \forall \boldsymbol{v}^h \in \mathbf{X}_0^{h_j}$$

$$b(\boldsymbol{u}_k^{h_j}, q^h) = 0 \qquad \forall q^h \in M^{h_j}$$

3. *Set* $\boldsymbol{u}^{h_j} = \boldsymbol{u}_{N_J}^{h_j}$, $p^{h_j} = p_{N_j}^{h_j}$.

2. INEXACT NEWTON METHODS

Newton's method is a very attractive way to solve systems of non-linear equations since in most cases it converges rapidly from any sufficiently good initial guess. In this method the non-linear system resulting from discretizing a partial differential equation is solved by first applying a Newton linearization, then using a linear solver to solve the resulting linear systems (the Jacobian systems) for each Newton iteration. However, in the computational process, solving a system of linear equations by a direct method such as Gaussian elimination can be inefficient if the number of the unknowns is large, and may not be justified when the iterate is far from the solution. Therefore, it seems reasonable to solve the linearized system only approximately using iteration. The resulting algorithm is categorised as an *inexact Newton method.*

We denote the discretized system of non-linear equations by

$$F(\mathbf{w}) = 0 \tag{27}$$

where $F : \mathbb{R}^n \to \mathbb{R}^n$ is assumed to be continuously differentiable in a neighbourhood of the solution $\mathbf{w}_*$, i.e. $F(\mathbf{w}_*) = 0$, and the Jacobian $F'(\mathbf{w}_*)$ is non-singular.[§] If $\mathbf{w}_k$ is the approximation solution of (27) at the kth step, Newton's method requires, for the next approximation solution, the evaluation of $F'(\mathbf{w}_k)$ and the solution of the linear system

$$F'(\mathbf{w}_k)\mathbf{s}_k = -F(\mathbf{w}_k) \tag{28}$$

where $\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{s}_k$. For inexact Newton the solution $\mathbf{s}_k$ of (28) is required to satisfy

$$\|F'(\mathbf{w}_k)\mathbf{s}_k + F(\mathbf{w}_k)\| \leqslant \eta_k \|F(\mathbf{w}_k)\| \tag{29}$$

where $\eta_k \in (0, 1]$ is called the *forcing term.*

Algorithm 2.1 (*Inexact Newton Algorithm*)

1. *Let* $\mathbf{w}_0$, $\eta_{\max} \in [0, 1)$ *be given*;

[§]Newton's method converges linearly if $F'(\mathbf{w}_*)$ is singular.

2. *for* $k = 0, 1, \ldots$ *until convergence.*
   - *Choose* $\eta_k \in [0, \eta_{\max}]$;
   - *Apply an iterative method to compute the solution* s_k *of* $F'(\mathbf{w}_k)s_k = -F(\mathbf{w}_k)$ *stopping the process when the condition* $\|F'(\mathbf{w}_k)\mathbf{s}_k + F(\mathbf{w}_k)\| \leqslant \eta_k \|F(\mathbf{w}_k)\|$ *is satisfied*;
   - *set* $\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{s}_k$.

The forcing term controls the local convergence of an inexact Newton method. It is important to specify it correctly; if η_k is too small, than it may lead to *oversolving* the Newton equation. Oversolving may result in little or no decrease in $\|F\|$ and, therefore, little or no progress toward a solution. Thus, a less accurate approximation of the Newton step may be both cheaper and more effective, see Reference [9] for details.

The following theorem describes the possible choices of the forcing term.

Theorem 2.1
Assume that (27) has solution $\mathbf{w}_*$, $F'(\mathbf{w})$ is Lipschitz continuous and $F'(\mathbf{w}_*)$ is non-singular. Then there are δ and $\bar{\eta}$ such that if $\mathbf{w}_0 \in \mathscr{B}(\delta)$, the ball of radius δ about $\mathbf{w}_*$, $\{\eta_k\} \subset [0, \bar{\eta}]$, then the inexact Newton iteration

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{s}_k$$

where

$$\|F'(\mathbf{w}_k)\mathbf{s}_k + F(\mathbf{w}_k)\| \leqslant \eta_k \|F(\mathbf{w}_k)\|$$

converges q-linearly[¶] to $\mathbf{w}_*$. Moreover

- if $\eta_k \to 0$ the convergence is q-superlinear,[¶] and
- if $\eta_k \leqslant K_\eta \|F(\mathbf{w}_k)\|^p$ for some $K_\eta > 0$ the convergence is q-superlinear with q-order $1 + p$.

Proof
See Reference [10, p. 97].

2.1. An efficient preconditioning strategy

Finite element methods applied to Picard linearization (16) or Newton linearization (22) require the solution of a large sparse linear system of equations. The application of a Krylov subspace iterative method to solve such a system produces iteration counts that are dependent on both mesh size, h, and Reynolds number, Re, for a given tolerance. Hence, to limit the growth of iteration counts, a preconditioner that produces little or no dependence on mesh size is essential. The preconditioner should be effective so that the improvement in the convergence rate will be more than enough to make up for the extra cost.

In this section, we briefly discuss an efficient preconditioning strategy for Krylov subspace type methods applied to the linear system arising from a Newton linearization (22). The use of Newton iteration is the main difference between this work and the work introduced by Kay *et al.* in Reference [5].

[¶]For precise definitions, we refer to Reference [10, pp. 65–66].

Here, we focus on the discrete system $\mathscr{L}v = b$ associated with (24), written in the form

$$\begin{pmatrix} A & B^{\mathrm{T}} \\ B & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{g} \end{pmatrix} \tag{30}$$

From now on the matrix $A \in \mathbb{R}^{2n \times 2n}$ and vector $\mathbf{f} \in \mathbb{R}^{2n \times 1}$ refer to A_N and $\mathbf{f}_N$ in (24), respectively.

Consider a right preconditioned system

$$[\mathscr{L}\mathscr{P}^{-1}][\mathscr{P}\mathbf{v}] = \mathbf{b}$$

with block triangular preconditioning

$$\mathscr{P}^{-1} = \begin{pmatrix} A^{-1} & R \\ 0 & -S^{-1} \end{pmatrix} \tag{31}$$

for matrices $R \in \mathbb{R}^{2n \times m}$ and $S \in \mathbb{R}^{m \times m}$. Then

$$\mathscr{L}\mathscr{P}^{-1} = \begin{pmatrix} I_{2n} & AR - B^{\mathrm{T}}S^{-1} \\ BA^{-1} & BR \end{pmatrix} \tag{32}$$

The choice of R and S in (31) is optimal if they satisfy the relationship

$$AR - B^{\mathrm{T}}S^{-1} = 0, \quad BR = I_m$$

that is, $R = A^{-1}B^{\mathrm{T}}S^{-1}$ with $S = BA^{-1}B^{\mathrm{T}}$. For this choice, it follows from (32) that the eigenvalues $\sigma(\mathscr{L}\mathscr{P}^{-1}) = \{1\}$, and preconditioned GMRES converges to the solution of (30) in at most two iterations; See Murphy *et al.* [11].

In the implementation of GMRES with a right preconditioner we need to compute the solution of a system of the form $\mathscr{P}\mathbf{y} = \mathbf{r}$ at every GMRES step. Hence, for the optimal choice R and S above, we seek a vector $\binom{\mathbf{v}}{\mathbf{q}}$ satisfying

$$\begin{pmatrix} \mathbf{v} \\ \mathbf{q} \end{pmatrix} = \begin{pmatrix} A^{-1} & A^{-1}B^{\mathrm{T}}S^{-1} \\ 0 & -S^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{r} \\ \mathbf{s} \end{pmatrix} \tag{33}$$

for given vectors $r \in \mathbb{R}^{2n}$, and $s \in \mathbb{R}^m$. Therefore the optimal preconditioner can be defined by a two-stage process:

$$\text{Solve for } \mathbf{q}\text{: } S\mathbf{q} = -\mathbf{s} \tag{34}$$

$$\text{Solve for } \mathbf{v}\text{: } A\mathbf{v} = \mathbf{r} - B^{\mathrm{T}}\mathbf{q} \tag{35}$$

This two-stage approach reduces the problem of finding preconditioners for (30) to that of finding good approximations of the actions of the inverse operators S^{-1} and A^{-1}.

The subproblem (35) entails approximating the solution of a set of discrete scalar convection diffusion equations. Hence the construction of $A_v^{-1} \approx A^{-1}$ is relatively straightforward. A_v^{-1} could be defined by applying a few cycles of a suitable multigrid method to the system. The

construction of P_S, the approximation to the Schur complement $S = BA^{-1}B^{\mathrm{T}}$, however, is more difficult. In the Oseen case, a good choice P_S for S was introduced by Kay *et al.* [5] and analysed by Silvester *et al.* [6]. The technique is determined by the choice

$$P_S = L_p A_p^{-1} Q_p \tag{36}$$

where A_p and $L_p \in \mathbb{R}^{m\times m}$ represent discrete approximations of convection–diffusion and scaled Laplacian operators defined on the pressure space M^h with a standard Neumann boundary condition, and $Q_p \in \mathbb{R}^{m\times m}$ is the pressure mass matrix, i.e. $[Q_p]_{i,j} = (\phi_i, \phi_j)$ for $i, j = 1, 2, \ldots, m$. This form of the P_S approximation was derived from the Green's tensor for the Oseen operator in [5] as an approximation to the inverse of a continuous representation of the Schur complement.

In a practical implementation, the computation of the preconditioner requires several subsidiary computations. For each step of GMRES, the preconditioning entails an approximation of the action of P_S^{-1} defined in (36) as seen from the following factorization:

$$\mathscr{P}^{-1} = \begin{pmatrix} A^{-1} & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} I & B^{\mathrm{T}} \\ 0 & -I \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & -P_S^{-1} \end{pmatrix}$$

The two-stage preconditioning process (34) and (35), therefore, can be implemented as follows.

Algorithm 2.2 (*Preconditioning process*)

1. *Compute the solution* $\mathbf{q}$ *by using the approximation* P_S^{-1} *of* S^{-1}. *This is done in three steps*:
   (a) *Application of multigrid iteration to a discrete Poisson problem with coefficient matrix* L_p.
   (b) *A matrix–vector product involving the discrete pressure convection–diffusion matrix* A_p.
   (c) *A scaling step corresponding to the solution of a system with coefficient matrix given by the pressure mass matrix* Q_p.
2. *Compute the solution* $\mathbf{v}$ *by applying a multigrid iteration for a diagonal blocks of the Jacobian matrix* A_N *in* (24) *defined by the matrix* A_v.

A good approximation to the Q_p^{-1} in step 1(c) of Algorithm 2.2 can be attained using a small number of (diagonally scaled) conjugate gradient iterations [5, 6].

3. NUMERICAL RESULTS I

In this section we present the results of numerical experiments with the Newton-GMRES algorithm described in Section 2.1. We run all the experiments in Matlab 5.3. As a test problem, we consider driven cavity flow (the steady state Navier–Stokes problem) modelled by Equations (1)–(3). These equations model the motion of a fluid in a unit square ($\{0 < x < 1, 0 < y < 1\}$) container (the domain Ω) whose lid at the boundary $y = 1$ moves from

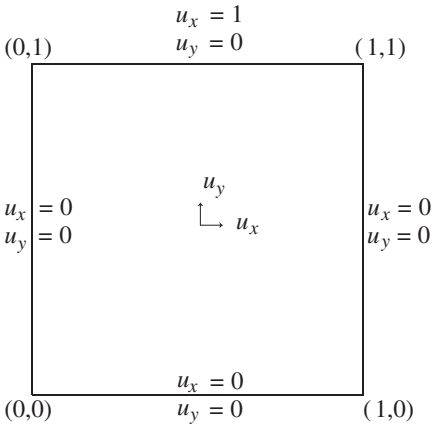

Figure 1. Geometry for driven cavity model problem.

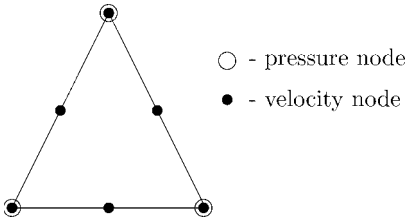

Figure 2. P_2–P_1 finite element.

left to right. A representative solution is given in Figure 3. The geometry and boundary conditions for the model problem are displayed in Figure 1.

The finite element discretization uses a triangular mesh, for example see Figure 2, with the so-called Taylor–Hood element pair; using piecewise quadratic functions on triangles to approximate the velocity and C^0 piecewise linear functions on triangles to approximate the pressure (P_2–P_1). This pair is known to uniformly satisfy the stability condition (15); see Reference [12, pp. 113–117].

The computation will follow Algorithm 1.1 by setting the initial guess $\boldsymbol{u}_0^{h_0} = \mathbf{0}$ at the coarsest mesh level. Furthermore, when we solve the linearized problem at level $j > 0$ with a mesh spacing h_j, we need the solution $\boldsymbol{u}^{h_{j-1}}$ generated on a mesh with spacing h_{j-1} and then interpolate this solution onto the grid with spacing h_j. The result $\boldsymbol{u}_0^{h_j}$ is used as the initial guess at mesh level j. The Newton iteration is terminated when $\|F(\boldsymbol{u}^{h_j})\|_2 \leqslant 10^{-7}\|F(\boldsymbol{u}_0^{h_j})\|_2$. In this computation we use the ℓ^2 norm to measure the non-linear residuals, since the GMRES method uses the scalar product with Euclidean norm.

In the approximation to the preconditioning process (34) described in Algorithm 2.2, we implement

- one multigrid V-cycle with point Gauss–Seidel smoother with one pre- and post-smoothing step on each level using the nodal ordering provided by Matlab PDE toolbox

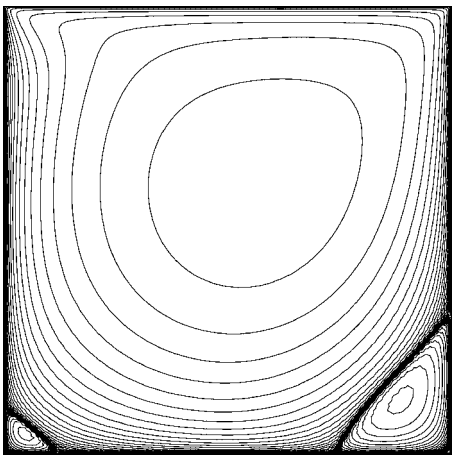

Figure 3. Streamline pattern for $Re = 400$.

software (for step 1a), together with two CG (diagonally scaled conjugate gradient) iterations (for step 1c), and

- two multigrid V-cycles using ILU(0) smoother with one pre- and post-smoothing step on each level, for step 2.

At the coarsest grid level of the multigrid computation we will perform an exact solution. We apply a simple linear interpolation as a prolongation ($\mathscr{I}_{j-1}^{j}$) to transfer linear corrections between grids, and its transpose ($\mathscr{I}_{j}^{j-1}$) as a restriction to transfer residuals between grids. Furthermore, if the local mesh Peclet number is greater than unity on any grid, then a streamline diffusion term [14] is included in the discrete system of convection–diffusion problems. See Reference [14] for details.

Finally, for the stopping criteria of the GMRES iteration, we implement two choices of the forcing term η: keeping η to be constant on all iterations; or else choosing η to vary on each Newton iteration as given in Theorem 2.1; that is $\eta_k = K\|F(\boldsymbol{u}_k^{h_j})\|$. We present the results in both tabular and graphical form. In all tables, we tabulate the number of Newton iterations k, the non-linear residual $\|F(\mathbf{u}_k)\|$, $\|\mathbf{u}_k - \mathbf{u}_{k-1}\|$, the average number of GMRES iterations per Newton iteration $\bar{m}$, the maximum number of GMRES iterations $m_{\max}$, and the total number of floating point operations in millions (Mflops).

In Figures 4 and 5 we plot the relative non-linear residual $\|F(\mathbf{u}_k)\|_2/\|F(\mathbf{u}_0)\|_2$ against the number of Newton iterations for the problem with uniform mesh size $h = 1/16$. The solid, dashed and dot–dashed curves are the plots of the relative non-linear residual for $Re = 50, 100$ and 200, respectively.

From Table I, we can see that setting η to a constant for all iterations seems to be reasonable, but the optimal choice of η itself is problem dependent. If η_k is too large, the convergence of the Newton iteration can be slow; 7 Newton iterations are needed for $\eta_k = 0.1$. Conversely, if η_k is too small, then there will be too many linear iterations and unnecessary work will be needed. This effort can be wasted in the initial stage of the iteration, when the Newton iterate is not within the radius of quadratic convergence. Figure 6, for example, shows the

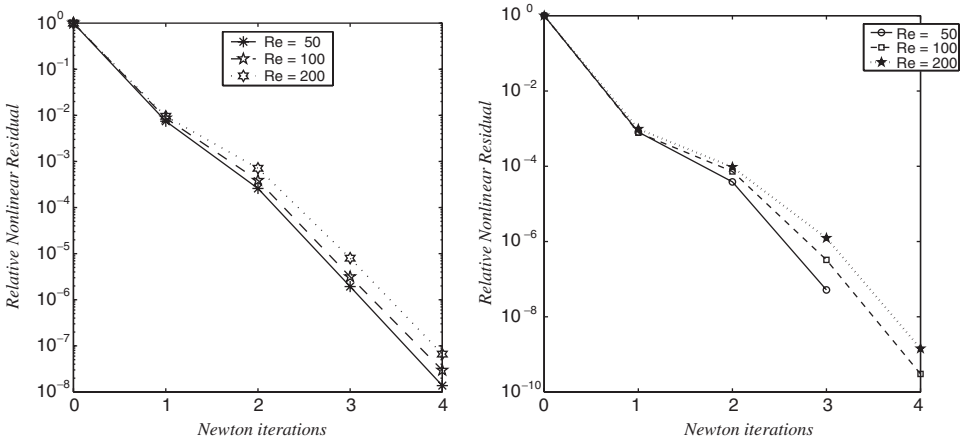

Figure 4. The convergence history of Newton's methods for: (a) $\eta = 0.01$ and (b) $\eta = 0.001$.

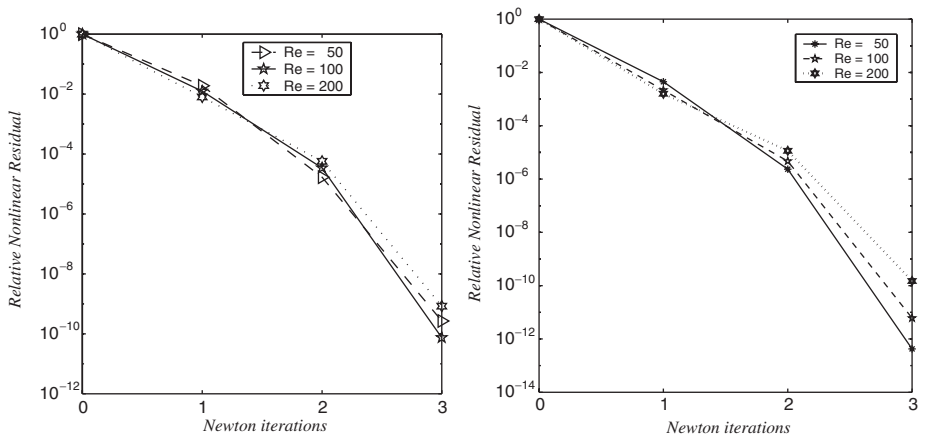

Figure 5. The convergence history of Newton's methods: (a) $\eta_k = 0.5\|F_k\|_2$ and (b) $\eta_k = 0.1\|F_k\|_2$.

convergence history of Newton iterations for various η ($= 0.1, 0.01$ and 0.001), with $Re = 50$. Newton's method, in this case, needs 7, 4 and 3 iterations to converge to the given tolerance for $\eta = 0.1, 0.01$ and 0.001, respectively. The convergence rate of Newton is almost linear for $\eta = 0.1$ and 0.01; On the other hand for $\eta = 0.001$, the convergence rate is close to being quadratic. Since $\eta = 0.001$ is too small for the problem with $Re = 50$, then the solution of the linear equation for the first step of Newton will reduce $\|F\|$ far beyond the level that is really needed. This over solving will engender little or no progress toward the solution. As a consequence, Newton's method will lose it's quadratic convergence property. Hence, choosing η to be a constant for the entire iteration in this case may not be most efficient.

Table I. The performance of preconditioned Newton-GMRES implementation where the forcing term η is constant.

						GMRES		
η	h	Re	Newton (k)	$\|F_k\|_2$	$\|\mathbf{u}_k - \mathbf{u}_{k-1}\|_2$	$\bar{m}$	$m_{\max}$	Mflops
0.1	$\frac{1}{16}$	50	7	4.1e-10	3.7e-7	12	14	227
		100	7	1.4e-9	4.1e-6	17	20	308
		200	7	2.0e-9	1.4e-5	35	41	641
	$\frac{1}{32}$	50	7	8.5e-10	4.4e-6	13	15	991
		100	7	9.0e-10	1.3e-7	19	22	1412
		200	7	1.2e-9	2.1e-5	31	38	2377
0.01	$\frac{1}{16}$	50	4	12.7e-10	1.1e-5	17	19	185
		100	4	14.6e-10	3.7e-5	25	28	264
		200	5	0.4e-10	9.5e-7	47	51	628
	$\frac{1}{32}$	50	4	65.3e-11	3e-5	18	19	805
		100	4	77.1e-11	1.2e-4	25	29	1130
		200	4	1.1e-9	4.3e-5	44	50	2075
0.001	$\frac{1}{16}$	50	3	26.7e-10	2.1e-4	22	23	162
		100	4	0.1e-10	1.1e-7	33	35	324
		200	4	0.4e-10	1.2e-5	59	62	622
	$\frac{1}{32}$	50	3	9.1e-10	4e-4	23	24	723
		100	3	16.7e-10	4.5e-4	31	33	996
		200	4	4.5e-12	3.3e-6	59	62	2649

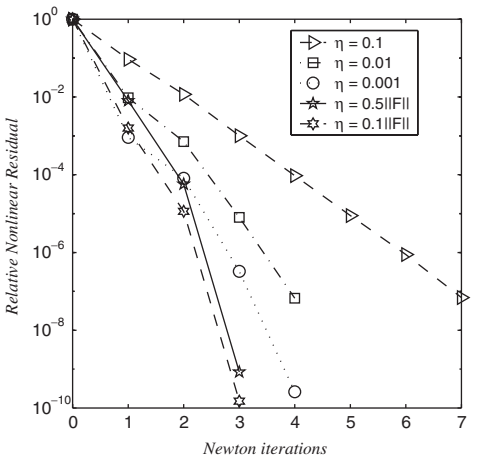

Figure 6. The convergence history of Newton's methods for $Re = 50$.

Tables II and V describe the performance of Newton method for uniform and stretched meshes, respectively. Here the forcing term is set to be different on each non-linear iteration, depending on the norm of non-linear residual, that is $\eta_k = K\|F_k\|$, where $K = 0.5$ and 0.1. With this choice of forcing term, there is a chance that there will be more linear iterations when

Table II. The performance of preconditioned Newton-GMRES implementations where the forcing term $\eta_k = K\|F_k\|_2$.

						GMRES		
K	h	Re	Newton (k)	$\|F_k\|_2$	$\|\mathbf{u}_k - \mathbf{u}_{k-1}\|_2$	$\bar{m}$	$m_{\max}$	Mflops
0.5	$\frac{1}{16}$	50	3	27.9e-12	5.2e-4	23	33	181
		100	3	51.7e-12	13.6e-4	33	46	264
		200	3	11.9e-10	78.9e-4	53	70	449
	$\frac{1}{32}$	50	3	12.7e-12	7.7e-4	24	36	818
		100	3	1.9e-12	3.8e-4	34	51	1172
		200	3	14.2e-12	1.6e-3	55	73	2012
0.1	$\frac{1}{16}$	50	3	0.4e-12	0.7e-4	27	38	211
		100	3	7.87e-12	6.4e-4	37	51	301
		200	3	93.7e-12	40.2e-4	61	76	522
	$\frac{1}{32}$	50	3	2e-14	5.6e-5	29	42	979
		100	3	1.6e-13	1.3e-4	38	54	1249
		200	3	2.6e-12	4.9e-4	64	79	2229

Table III. The performance of Newton-GMRES implementations with exact preconditioning process and $\eta_k = 0.5\|F_k\|_2$.

					GMRES		
h	Re	Newton (k)	$\|F_k\|_2$	$\|\mathbf{u}_k - \mathbf{u}_{k-1}\|_2$	$\bar{m}$	$m_{\max}$	Mflops
$\frac{1}{16}$	50	3	4e-11	3e-4	22	32	441
	100	3	6e-10	1e-3	32	46	634
	200	3	2e-8	8e-3	49	66	1016
$\frac{1}{32}$	50	3	1e-10	1e-3	23	35	4544
	100	3	2e-11	5e-4	33	48	6586
	200	3	1e-10	1e-3	52	69	10459

the Newton iteration is closer to the true solution. Hence, the solution of the linear equation for the last step of Newton iteration will be more accurate. Note, the maximum numbers of linear iterations presented in column 8 of Table II are the numbers of iterations on the last step of the Newton iteration.

The convergence history is presented in Figures 5(a) and 5(b). The figures show that Newton's method converges quadratically and stops after three iterations, for both $K = 0.5$ and 0.1, and all Reynolds numbers $Re = 50, 100$ and 200. In addition, Table II also show that the choice of $K = 0.5$ is more efficient than that of $K = 0.1$. Based on the results presented in all tables it appears that $\eta_k = 0.5\|F_k\|$ is the best selection for this problem.

Furthermore, Table III illustrates the performance of Newton-GMRES where the preconditioning process in Algorithm 2.2 is done exactly, i.e. using the back slash in Matlab to effect the linear solution associated with matrices L_p, Q_p and A_v. We now compare this table with Table II. In particular, we look at the average number of GMRES iterations per Newton iteration ($\bar{m}$) and maximum GMRES iteration count $m_{\max}$ from both tables. It seems

Table IV. The performance of Newton-GMRES implementations with Picard iteration as initial guess and $h = \frac{1}{32}$.

			$\bar{m}$ (GMRES)		Mflops	
η_k	Re	Newton	MG	(RMG)	MG	(RMG)
$0.5\|F_k\|_2$	500	5	125	(121)	8366	(8322)
$0.5\|F_k\|_2$	1000	5	293	(264)	37438	(31385)
0.01	2000	6	317	(294)	45607	(41853)

Table V. The performance of preconditioned Newton-GMRES implementations on non-uniform mesh, where the forcing term $\eta_k = K\|F_k\|_2$.

						GMRES		
K	h	Re	Newton (k)	$\|F_k\|_2$	$\|\mathbf{u}_k - \mathbf{u}_{k-1}\|_2$	$\bar{m}$	$m_{\max}$	Mflops
		50	4	1.0e-12	3.9e-5	29	52	311
	$\frac{1}{16}$	100	3	2.9e-10	3.0e-3	36	53	295
0.5		200	3	5.5e-10	2.6e-3	58	80	508
		50	3	1.1e-7	6.6e-1	28	47	989
	$\frac{1}{32}$	100	3	8.1e-9	5.7e-2	35	56	1755
		200	3	3.0e-11	3.3e-3	66	96	2551
		400	3	1.0e-11	1.2e-3	121	175	5798

that these numbers are very similar. The last column of both tables, however, shows that solving the preconditioner exactly needs more floating point computation. This illustrates that the multigrid used in the preconditioning process is working well.

In Table IV, we present the performance of the non-linear iterative method for higher Reynolds number (i.e. $Re = 500$, 1000 or 2000). In this case one step of the Picard iterative method is applied to provide a good initial approximation for Newton's method. It is worth noting that without the Picard iteration, Newton's method will diverge in this case. The table also displays the average number of GMRES iterations per Newton iteration. Note that the number in the bracket refers to multigrid GMRES with nodes numbered to follow the flow direction (RMG-GMRES). The table shows that Picard iteration can improve the convergence of Newton's method. In addition, numbering the nodes to follow the flow direction gives a small improvement in the convergence rate of the GMRES method. The last two columns of the table, however, show that the improvement seems to be sufficiently effective to compensate for the extra cost.

Finally, Tables I, II and Figure 7 show that the convergence rate of our preconditioned GMRES algorithm is independent of the mesh size and is a slowly growing function of the Reynolds number (as shown in Figure 8).

4. NUMERICAL RESULTS II: COMPARISON

We briefly compare our results with those published in References [2, 4]. In Reference [2], Ghia *et al.* use the coupled strongly implicit multigrid (CSI-MG) method to solve a two di-

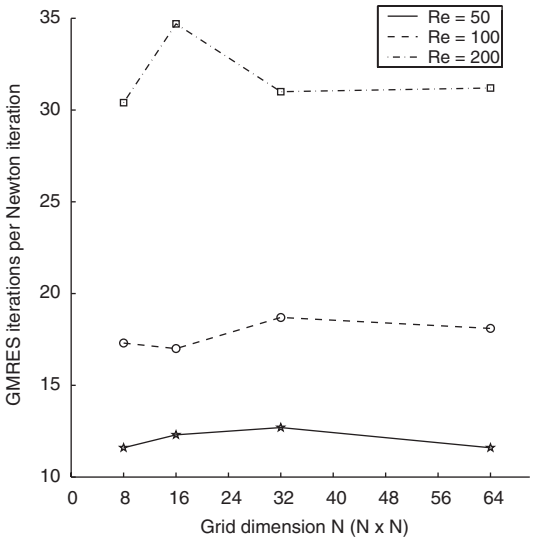

Figure 7. The average number of GMRES iterations per Newton iteration for $\eta = 0.1$.

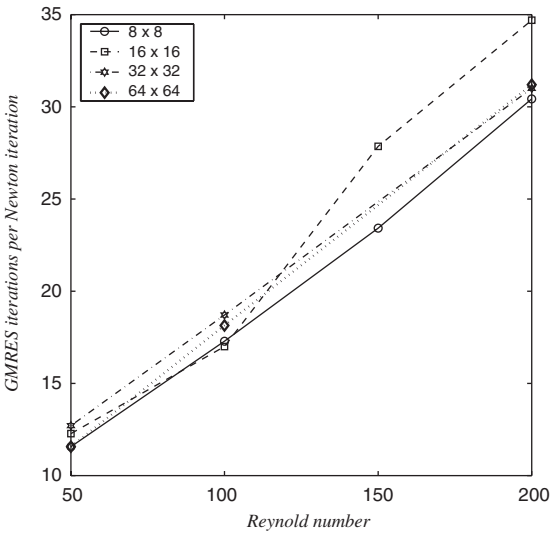

Figure 8. A convergence history of GMRES iterations for various Reynolds numbers, $\eta = 0.1$.

mensional Navier–Stokes problem. The main idea of this approach is to apply FAS-FMG (full approximation scheme—full multigrid) to the non-linear problem with Gauss–Seidel smoother and 9-point prolongation and restriction. The finite difference discretization is implemented on uniform grids. Table 1 in Reference [2], presents the results for velocity along the vertical line through the geometric centre of the cavity on the finest (129×129) grid. In addition,

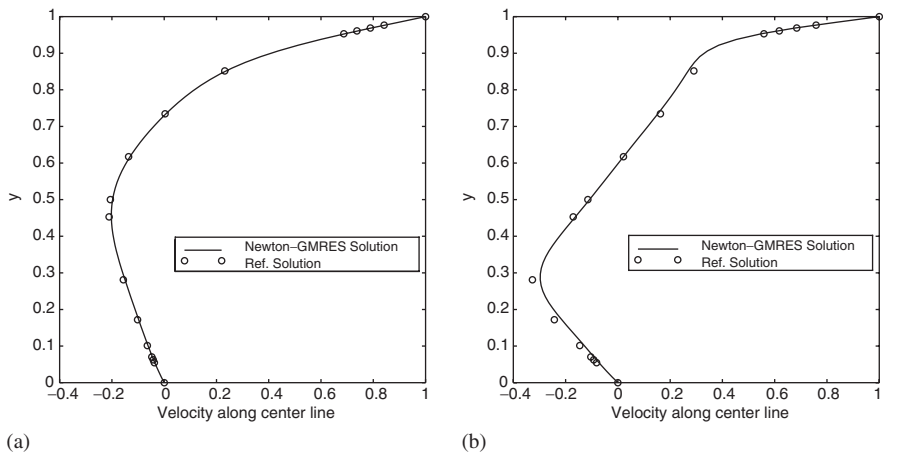

Figure 9. Comparison of u-velocity for: (a) $Re = 100$ and (b) $Re = 400$; uniform mesh.

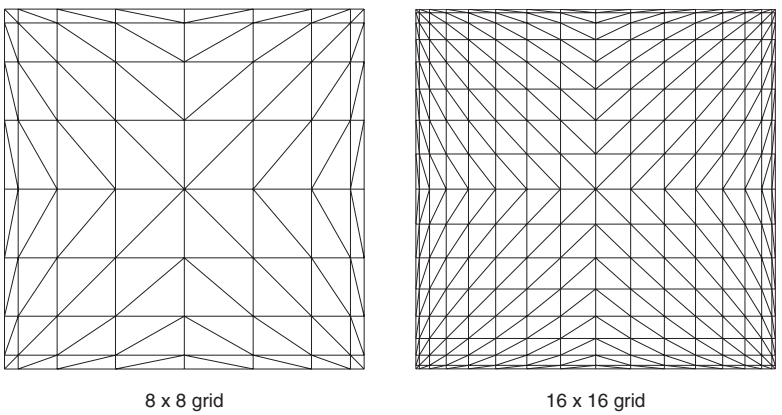

Figure 10. Stretched grids.

we compare our results for $Re = 100$ and 400 with $h = 1/32$ with those in Reference [2]. To do this, we plot the centre line velocity of our results. The graphics are displayed in Figure 9(a) and Figure 9(b). The solid line corresponds to our preconditioned Newton-GMRES solution and the small circles correspond to the reference solution in Reference [2]. At $Re = 100$, see Figure 9(a), the results are almost indistinguishable. Conversely, Figure 9(b) shows slight disagreement in the results at $Re = 400$. This difference arises since our computations were performed on uniform meshes. To obtain better results we perform our computations on stretched grids (see Figure 10), where there are more triangles at the corners. This non-uniform grid will give better resolution at the corners, and then our results are in close agreement with the reference results, as shown in Figure 11. Non-uniform spaced streamlines are displayed in Figure 3. Note that the coarsest grid in our hierarchy is the 8×8 non-uniform grid illustrated in Figure 10.

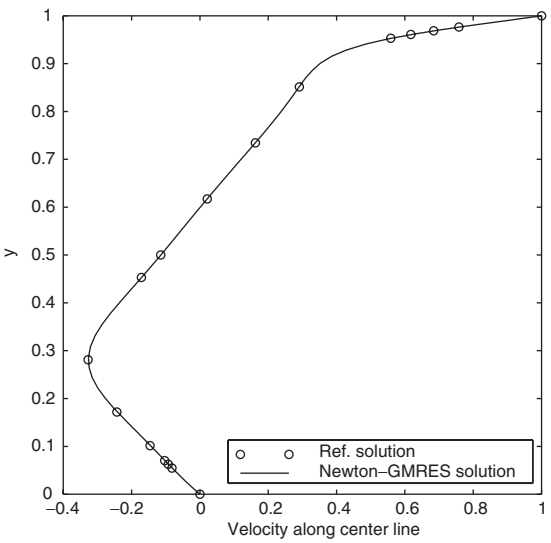

Figure 11. Comparison of u-velocity for $Re = 400$; non-uniform mesh.

In Reference [4], Knoll and Rider solved the driven cavity problem using a Newton–Krylov method (NKMG). A standard multigrid V-cycle is used as a preconditioner with fixed and equal numbers of pre- and post-smoothing steps ($V(\tau,\tau)$) on each level. The smoother is a block SGS (symmetric Gauss–Seidel) method. For Reynolds number $Re = 100$, non-linear convergence tolerance $\|F_k\|_2 \leqslant 10^{-6}$ and the forcing term $\eta_k = 0.01$, the plot of the average of GMRES iterations per Newton iteration against the mesh level is presented in Figure 5 in Reference [4]. The iteration counts are obtained using GMRES(10) and GMRES(20) preconditioned using one multigrid V-cycle with three smoothing sweeps ($V(3,3)$) on each level. On the coarsest level (10×10) 15 block SGS steps are used. We take this result as our first comparison. In particular, we will look at the convergence of our method using the same non-linear tolerance and forcing term. However, in our preconditioning process we use one $V(1,1)$ with a point Gauss–Seidel smoother for the Laplace operator and one $V(1,1)$ with an ILU(0) smoother for the convection–diffusion operator.

Figure 5 in Reference [4] suggests that on the finest grid (32×32) and (64×64) there are about 8 and 9 GMRES(10) iterations per Newton iteration (equivalently 24 and 27 MG cycles), respectively. Compared with our result which is 25 and 27 standard GMRES iterations per Newton iteration, it seems that these iteration counts are comparable.

Furthermore, we consider the iteration counts in Table 8 in Reference [4]. For $Re = 100$ with the accuracy of 10^{-5} and $\eta = 0.05$ (cf. our result in Table I with non-linear tolerance $= 10^{-7}$ and $\eta = 0.01$) NKMG needed 11 $V(10,10)$ cycles with three Newton iterations (or equivalently about 37 standard $V(1,1)$ cycles per Newton iteration) on the finest (20×20) and (40×40) grid (Table VI). Again our results are still competitive. In addition, the convergence rate of the NKMG method seems to be sensitive to the number of smoothing steps. The exciting potential of our method is the fact that the convergence rate is realized using only one smoothing step.

Table VI. The performance of preconditioned Newton-GMRES implementations on the non-uniform mesh, where the forcing term $\eta = 0.1$.

						GMRES		
η	h	Re	Newton (k)	$\|F_k\|_2$	$\|\mathbf{u}_k - \mathbf{u}_{k-1}\|_2$	$\bar{m}$	$m_{\max}$	Mflops
0.1	$\frac{1}{32}$	50	7	1.8e-9	4.0e-5	17	20	1258
		100	7	2.1e-9	2.2e-5	20	25	1535
		200	7	2.3e-9	1.9e-5	35	46	2767

5. CONCLUSIONS

We have demonstrated the performance of a proposed new preconditioning methodology applied to the incompressible Navier–Stokes equations. Through numerical experiments, we observe that the performance of our inexact Newton algorithm is independent of mesh size and Reynolds number. However, the convergence rate of this non-linear iteration does depend on the choice of a forcing term. Choosing the forcing term to be proportional to the non-linear residual is advisable—this leads to second order convergence in the neighbourhood of the solution. The attractive feature of our preconditioning methodology is that the convergence rate of the linear iteration is not affected by mesh size and is relatively insensitive to Reynolds number.

REFERENCES

1. Engelman MS, Strang G, Bathe KJ. The application of quasi-Newton methods in fluid mechanics. *International Journal for Numerical Methods in Engineering* 1981; **17**:707–718.
2. Ghia U, Ghia KN, Shin CT. High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method. *Journal of Computational Physics* 1982; **48**:387–411.
3. McHugh PR, Knoll DA. Fully coupled finite volume solutions of the incompressible Navier–Stokes and energy equations using an inexact Newton method. *International Journal for Numerical Methods in Fluids* 1994; **19**: 439–455.
4. Knoll DA, Rider WJ. A multigrid preconditioned Newton–Krylov method. *SIAM Journal on Scientific Computing* 1999; **21**:691–710.
5. Kay D, Loghin D, Wathen AJ. A preconditioner for the steady state Navier–Stokes equations. *SIAM Journal on Scientific Computing* 2002; **24**:237–256.
6. Silvester DJ, Elman HC, Kay D, Wathen A. Efficient preconditioning of the linearized Navier–Stokes equations. *Journal of Computational and Applied Mathematics* 2001; **128**:261–279.
7. Elman HC. Preconditioning for the steady-state Navier–Stokes equations with low viscosity. *SIAM Journal on Scientific Computing* 1999; **20**:1299–1316.
8. Girault V, Raviart PA. *Finite Element Methods for Navier–Stokes Equations—Theory and Algorithms*. Springer: Berlin, 1986.
9. Eisenstat SC, Walker HF. Choosing the forcing terms in an inexact Newton method. *SIAM Journal on Scientific Computing* 1996; **17**:16–32.
10. Kelley CT. *Iterative Methods for Linear and Nonlinear Equations*. SIAM: Philadelphia, 1995.
11. Murphy MF, Golub GH, Wathen AJ. A note on preconditioning for indefinite linear systems. *SIAM Journal on Scientific Computing* 2000; **21**:1969–1972.
12. Gunzburger MD. Navier–Stokes equations for incompressible flows: finite element methods. *Handbook of Computational Fluid Mechanics*. Academic Press: London, 1996.
13. Gunzburger MD. *Finite Element Methods for Viscous Incompressible Flows*. Academic Press: London, 1989.
14. Ramage A. A multigrid preconditioner for stabilised discretizations of advection-diffusion problems. *Journal of Computational and Applied Mathematics* 1999; **101**:187–203.
15. Layton W, Lee HK, Peterson J. Numerical solution of the stationary Navier–Stokes equations using a multilevel finite element method. *SIAM Journal on Scientific Computing* 1998; **20**:1–12.

16. Pernice M, Wolker HF. NITSOL: A Newton iterative solver for nonlinear system. *SIAM Journal on Scientific Computing* 1998; **19**:302–318.
17. Oosterlee CW, Washio T. An evaluation of parallel multigrid as a solver and a preconditioner for singularly perturbed problems. *SIAM Journal on Scientific Computing* 1998; **19**:87–110.
18. Dembo RS, Eisenstat SC, Steihaug T. Inexact Newton methods. *SIAM Journal on Numerical Analysis* 1982; **19**:400–408.
19. Elman HC, Silvester DJ, Wathen AJ. Performance and analysis of saddle point preconditioners for the discrete steady state Navier–Stokes equations. *Numerical Mathematics* 2002; **90**:665–688.

