DELFT UNIVERSITY OF TECHNOLOGY

REPORT 06-06

Study Report of IFISS Package(version 2.1)

M. ur Rehman, C. Vuik G. Segal

ISSN 1389-6520

Reports of the Department of Applied Mathematical Analysis

Delft 2006

STUDY REPORT OF IFISS PACKAGE (VERSION 2.1)

M. UR REHMAN, C. VUIK, AND G. SEGAL

Abstract. The main core of this report is to explore the functionality of IFISS package [1]. IFISS stands for Incompressible Flow Iterative Solution Software (IFISS) designed in Matlab by David J. Silvester [2], Howard C. Elman [3] and Alison Ramage [4]. IFISS comprises of discretization and solving schemes for four type of PDEs (Poisson, Convection diffusion, Stokes and Navier Stokes equations) . All these schemes along with problem specifications are explored in this report.

Contents

List of Figures

List of Tables

Introduction

The main feature of the package concerns problem specification and finite element discretization. The discretized problem is then solved with Krylov iterative solution method using suitable preconditioners. IFISS is used to solve four types of problems in fluid dynamics

(1) Poisson Equation
(2) Convection Diffusion Equation
(3) Stokes Problem
(4) Navier Stokes Problem

In each of these problems there are four subproblems solved on different domains and boundary conditions. We will discuss these problems along with discretization schemes and preconditioned iterative solvers. At the end, few results were obtained to have an idea,how IFISS works.

1. The Poisson Equation

The Poisson problem is given as

$$-\nabla^2 u = f \text{ in } \Omega, \tag{1}$$

$$u = g_D \text{ on } \partial\Omega_D, \ \frac{\partial u}{\partial n} = 0 \ \text{ on } \ \partial\Omega_N,$$

Problem-1: Square domain with Constant Source with zero Dirichlet conditions on boundary
problem-2: The source function and boundary conditions are the same as above but here the domain is L-Shaped.
Problem-3: The source function is identically zero and the boundary conditions are chosen for the problem with analytical solution

$$u(x,y) = \frac{2(1+y)}{(3+x)^2 + (1+y)^2}$$

in a square domain.
Problem-4: The source function is identically zero and the boundary conditions are chosen such that the problem having the analytical solution

$$u(r,\theta) = r^{2/3} sin(\frac{2\theta + \pi}{3}),$$

Where r is the radial distance from the origin and θ is the angle with vertical axis. The problem is solved in an L-shaped domain.

2. Convection-Diffusion Equation

The equation is given as

$$-\varepsilon\nabla^2 u + \vec{w}.\nabla u = 0 \text{ in a 2-dimensional domain } \Omega, \tag{2}$$

$$\text{with boundary conditions } u = g_D \text{ on } \partial\Omega_D, \ \frac{\partial u}{\partial n} = 0 \text{ on } \partial\Omega_N$$

The following problems are solved

(1) For a constant convection velocity vector $\vec{w} = (0,1)$, the function

$$u(x,y) = x(\frac{1 - e^{\frac{y-1}{\epsilon}}}{1 - e^{-\frac{2}{\epsilon}}})$$

satisfies the convection-diffusion equation exactly. Dirichlet conditions on the boundary satisfy $u(x,-1) = x$, $u(x,1) = 0$, $u(-1,y) \approx -1$ $u(1,y) \approx 1$,

the problem has also an option to apply the natural boundary condition at outflow.

(2) Here $\vec{w} = (0, 1+(x+1)^2/4)$, so wind is vertical but increasing in x direction. The function u is one at inflow and decreases to zero quadratically on the right wall and cubically on the left wall.

(3) Here $\vec{w} = (-sin\frac{\pi}{6}, cos\frac{\pi}{6})$, constant wind at an angle of 30° to the left of the vertical.The Dirichlet conditions are zero on the left and top boundaries and unity on the right boundary.

(4) The wind $\vec{w} = (2y(1-x^2), -2x(1-y^2))$ determines recirculating flow. The Dirichlet conditions have value one on the right wall and zero every where else. Applying the Galerkin method to discretize the problem often results in oscillations. These oscillations are minimized though use of a streamline diffusion method by adding diffusion in streamline direction [8] [9]. The resulting discrete equation becomes
$\epsilon(\nabla u_h, \nabla v_h) + (\vec{w}.\nabla u_h, v_h) + \sum_k \delta_k(\vec{w}.\nabla u_h, \vec{w}.\nabla v_h)_{\square_k} = 0 \ \forall v_h \in V_h$ where the sum is taken over all elements in the grid and

$$\delta_k = \begin{cases} \frac{h_k}{2|\vec{w}|}(1 - \frac{1}{P_h^k}) & \text{if } P_h^k > 1, \\ 0 & \text{if } P_h^k \leq 1 \end{cases}$$

and $P_h^k = |\vec{w}|h_k/(2\epsilon)$ is so called element Peclet number.

3. The Stokes Equation

The equation is given by

$$-\nabla^2\vec{u} + \nabla p = \vec{0}, \tag{3}$$

$$\nabla.\vec{u} = 0, \tag{4}$$

with boundary conditions $\vec{u} = \vec{w}$ on $\partial\Omega_D$, $\frac{\partial\vec{u}}{\partial n} - \vec{n}p = \vec{0}$ on $\partial\Omega_N$. The variable $\vec{u}$ represents a vector valued function representing the velocity of fluid and scalar function p represents pressure. The assumption is made that the flow is 'low speed', so that convection effects can be neglected. After discretization we come up with the system

$$\begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \tag{5}$$

This is the system of dimension $n_u + n_p$ where n_u and n_p are the number of velocity and pressure basis functions.The matrix A is vector Laplacian matrix and B is n_px n_u rectangular matrix. Different discretization schemes are discussed in Section 5. In the case when velocity and pressure are defined on the same grid points, we may expect that sometimes there are more pressure unknowns than velocity unknowns and the system will be either inconsistent or singular. For example in the case of an enclosed flow problem on a square grid with an even number of elements, the null space of B^T is actually eight-dimensional (chapter 5 of [18]). To get a stable method, the velocity approximation needs to be enhanced relative to the pressure. The easiest way of doing this is to use biquadratic approximation for the velocity components rather than bilinear. In case of unstable discretization, IFISS uses a stabilization scheme, in which the system given in (5) becomes

$$\begin{bmatrix} A & B^T \\ B & -\beta C \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \tag{6}$$

where $\beta > 0$ is a stabilization parameter and C is a n_px n_p symmetric positive definite matrix. The following problems are solved in IFISS

(1) This problem represents flow in channel, known as Poiseuille flow. The Dirichlet conditions at inflow $x = -1$ is given by $\vec{u} = (1 - y^2, 0)$ and no slip conditions $\vec{u} = 0$ are applied at top and bottom boundaries. One can apply Dirichlet or Neumann conditions at the outflow boundary.
(2) This problem represents flow over a step with $L = 5$. The inflow, top and boundary conditions are the same as that in Problem 1, at output Neumann conditions are applied.
(3) This problem used in fluid dynamics is known as driven cavity flow. It models a square cavity with lid moving from left to right. Different choices of non zero horizontal velocity on the lid give rise to different computational models:

$$\{y = 1; -1 \leq x \geq 1 |\ u_x = 1\},\ \text{a leaky cavity;}$$

$$\{y = 1; -1 < x < 1 |\ u_x = 1\},\ \text{a water tight cavity;}$$

$$\{y = 1; -1 \leq x \geq 1 |\ u_x = 1 - x^4\},\ \text{a regularized cavity;}$$

(4) The analytic problem is associated with following solution of Stokes equation system:

$$u_x = 20xy^3;\ u_y = 5x^4 - 5y^4;\ p = 60x^2y - 20y^3 + constant$$

It is simple model of colliding flow, the Dirichlet conditions on each boundary can be specified from u_x and u_y.

4. Navier Stokes equations

The steady state Navier-Stokes systme of equations is given as

$$-\nu\nabla^2\vec{u} + \vec{u}.\nabla\vec{u} + \nabla p = \vec{f}, \tag{7}$$

$$\nabla.\vec{u} = 0, \tag{8}$$

where $\nu > 0$ is a given constant called the *kinematic viscosity*. The boundary conditions for the 2-dimensional domain Ω is given

$$\vec{u} = \vec{w} \text{ on } \partial\Omega_D,\ \nu\frac{\partial\vec{u}}{\partial n} - \vec{n}p = \vec{0} \text{ on } \partial\Omega_N,$$

The discretization leads to

$$\begin{bmatrix} F & B^T \\ B & -\frac{1}{\nu}C \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \tag{9}$$

in case of stable approximation $C = 0$. The following Navier Stokes problems are solved.

(1) The Poiseuille channel flow solution with parabolic inflow and natural outflow boundary condition in a square domain $\Omega_\square = (-1, 1)^2$,

$$u_x = 1 - y^2;\ u_y = 0;\ p = -2\nu x;$$

It satisfies the natural outflow boundary condition $\nu\frac{\partial u_x}{\partial x} - p = 0$ and $\frac{\partial u_y}{\partial x} = 0$.
(2) A Poiseuille flow profile is imposed on the inflow boundary ($x = -1;\ 0 \leq y \leq 1$) and a no flow condition is applied on the walls. The mean outflow pressure is set to zero by applying Neumann condition at outflow boundary ($x = 5; -1 < y < 1$).
(3) The Problem 3 of Stokes flow is now solved with varying Re in a square domain.

(4) This problem models boundary layer flow over a flat plate, or Blasius flow. The problem is equivalent to that of computing the steady flow over a flat plate moving at constant speed through a fluid that is at rest. The parallel flow Dirichlet condition $\vec{u} = (1, 0)$ is imposed at the inflow boundary $(x = -1; -1 \leq y \leq 1)$ and also on the top and bottom of the channel $(-1 \leq x \leq 5; y = \pm 1)$, representing walls moving from left to the right with speed unity. A no flow condition is imposed on the internal boundary $(0 \leq x \leq 5; y = 0)$, and a Neumann condition is applied at the outflow boundary $(x = 5; -1 < y < 1)$.

5. Finite Element discretization

IFISS solves the problems with these three physical domains

- A square grid (-1, 1) x (-1, 1) is used in solving diffusion and convection diffusion problems.
- The L-Shaped region (-1, L) x (-1,1), In case of a diffusion problem, $L = 1$, for Stokes problem $L = 5$ and L can be varied in Navier Stokes.
- The rectangular region (-1, 5) x (-1, 1) with slit along the line where $0 \leq x \geq 5$ and $y = 0$; In most of the problems the grid is uniformly divided, but in some cases there is also an option for stretched grids.

The following options are used for discretization of different problems.

(1) Q_1 - Bilinear quadrilateral consists of four basis function where each function is of the form $(ax + b)(cy + d)$(four unknowns per element). This approximation is used in Poisson and convection-diffusion problems.
(2) Q_2 - Biquadratic quadrilateral consists of nine basis function each of the form $(ax^2 + bx + c)(dy^2 + ey + f)$(nine unknowns per element). This approximation is used in Poisson problems.
(3) Q_2-Q_1: The element from the *Taylor Hood family* uses continuous approximations for velocity and pressure. Q_2 approximation for velocity and Q_1 approximation for pressure.
(4) $Q_2 - P_{-1}$: The element from Crouzeix Raviart family uses a Q_2 approximation for velocity and a discontinuous linear pressure. P_{-1} element has central node with three associated degrees of freedom, the pressure at the centroid and its derivatives in both directions.
(5) Q_1-Q_1: This combination corresponds to velocity - pressure in Stokes and Navier Stokes problems, in which velocity and pressure both are approximated by bilinear basis functions.
(6) Q_1-P_0:This represent bilinear approximation for velocity and constant approximation for the pressure. $Q_1 - Q_1$ and $Q_1 - P_0$ are considered to be unstable pairs of approximation. The stabilization scheme is discussed in section 4.

Solving the Navier-Stokes equations requires non-linear iterations with a linearized problem to be solved at each step. In this case initial velocity estimates are required to start the iterative process. Three iterative procedures are implemented.
Newton Iteration: Suppose we want to compute solution u^{k+1} at a present iteration level $k + 1$. The convective term can be linearized as

$$u^{k+1}.\nabla u^{k+1} = u^{k+1}.\nabla u^k + u^k.\nabla u^{k+1} - u^k.\nabla u^k.$$

If the initial estimate is good, Newton iteration converges quadratically. However, if the distance between the estimated and exact solution is too large, then iterations converges slowly or may diverge.
Picard Iteration: In this method the convection term is approximated by

$$u^{k+1}.\nabla u^{k+1} \simeq u^k.\nabla u^{k+1}$$

The Picard iteration seems to have larger convergence region, which means that iterations does not require an accurate initial estimate, however this method converges linearly.
Hybrid Method: This method uses the nice properties of the Newton and Picard method, some Picard iterations are followed by Newton iterations. Based on Picard iterations, Newtons method gets a good initial estimate. An initial guess for Picard is provided by the solution of the Stokes problem.

6. Error Estimations

IFISS computes an error approximation whenever the $Q1$ velocity approximation is used. Given a finite element subdivision $\mathcal{T}_h$ and a solution u_h, a local error estimate η_T is calculated such that $\|\nabla\eta_T\|$ approximates the local energy error $\|\nabla(u-u_h)\|_T$ for every element T in $\mathcal{T}_h$. We start with the Poisson problem to find $u \in \mathcal{H}^1_E$, the weak formulation of Poisson problem leads to

$$\int_\Omega \nabla u.\nabla v = \int_\Omega vf + \int_{\partial\Omega} vg_N \text{ for all } v \in \mathcal{H}^1_{E_0}, \tag{10}$$

We restate the problem as:
Find $u \in \mathcal{H}^1_E$ such that

$$a(u,v) = \ell(v) \text{ for all } v \in \mathcal{H}^1_{E_0}, \tag{11}$$

where $a(u,v) = \int_\Omega \nabla u.\nabla v$ and $\ell(v) = \int_\Omega vf + \int_{\partial\Omega} vg_N$, for simplicity it is assumed that the Neumann boundary condition is homogeneous, so that $\ell(v) = \int_\Omega vf$. The corresponding discrete problem is then given by: Find $u_h \in \mathcal{S}^h_E$ such that

$$a(u_h,v_h) = \ell(v_h) \text{ for all } v_h \in \mathcal{S}^h_0, \tag{12}$$

For an error estimation, subtract $a(u_h,v)$ from (11) to give

$$a(u,v) - a(u_h,v) = \ell(v) - a(u_h,v).$$

An assumption is that $\mathcal{S}^h_E \subset \mathcal{H}^1_E$ implies that $e = u - u_h \in \mathcal{H}^1_{E_0}$ and satisfies

$$a(e,v) = \ell(v) - a(u_h,v) \text{ for all } v \in \mathcal{H}^1_{E_0}, \tag{13}$$

The error equation (13) may be broken up into element contributions $\mathcal{T} \in \mathcal{T}_h$.

$$\sum_{\mathcal{T}\in\mathcal{T}_h} a(e,v)_T = \sum_{\mathcal{T}\in\mathcal{T}_h} (f,v)_T - \sum_{\mathcal{T}\in\mathcal{T}_h} a(u_h,v)_T \tag{14}$$

Integrating by parts elementwise then gives

$$-a(u_h,v)_T = (\nabla^2 u_h, v)_T - \sum_{\mathcal{E}\in\varepsilon(T)} \langle \nabla u_h.\vec{n}_{E,T}, v\rangle_E, \tag{15}$$

Where $\varepsilon(T)$ denotes the set of edges of element T, $\vec{n}_{E,T}$ is the outward normal with respect to E, $\langle .,. \rangle_E$ is the L_2 inner product on E, and $\nabla u_h.\vec{n}_{E,T}$ is the discrete normal flux. The finite element approximation typically has a discontinuous normal derivative across inter-element boundaries. So it is convenient to define the flux jumps across edge or face E adjoining elements T and S as

$$\left[\frac{\partial u}{\partial n}\right] := (\nabla u|_T - \nabla u|_S).\vec{n}_{E,T} \tag{16}$$

Putting the expressions (15) and (16) in (14), (14) becomes

$$\sum_{\mathcal{T}\in\mathcal{T}_h} a(e,v)_T = \sum_{\mathcal{T}\in\mathcal{T}_h} \left[(f+\nabla^2 u_h, v)_T - \tfrac{1}{2}\textstyle\sum_{E\in\varepsilon(T)} \langle \left[\frac{\partial u}{\partial n}\right], v\rangle_E\right] \tag{17}$$

The right hand side of (17) shows that e has two distinct components; these are the interior residual $R_T := f + \nabla^2 u_h|_T$, and the inter-element flux jumps $R_E := \left[\frac{\partial u}{\partial n}\right]$. In the case of Q_1 approximation, R_E is piecewise linear and $R_T = f|_T$ can be

approximated by a piecewise constant function R_T^0 by evaluating f at the element centroid. Consider the higher order correction space

$$\mathcal{Q}_T = Q_T \oplus B_T,$$

so that Q_T is the space spanned by biquadratic edge bubbles and B_T is the space spanned by interior biquadratic bubbles. The energy norm of the error is estimated by solving a 5x5 local Poisson problem posed over each element of the grid:

$$(\nabla e_T, \nabla v) = (R_T^0, v)_T - \sum_{E \in \varepsilon(T)} \langle R_E, v \rangle_E \ \forall v \in \mathcal{Q}_T \tag{18}$$

The local error estimator is given by $\eta_T = \|\nabla e_T\|_T$, and the global error estimator is given by

$$\eta := \left(\sum_{\mathcal{T} \in \mathcal{T}_h} \eta_T^2\right)^{1/2} \approx \|\nabla(u - u_h)\|$$

In case of the Navier Stokes problem, the local error estimation in IFISS is given by the combination of energy norm of the velocity error and the L_2 norm of the element divergence error, that is given as

$$\eta_T^2 = \|\nabla \vec{e_T}\|_T^2 + \|R_T\|_T^2 \tag{19}$$

Where $\nabla \vec{e}_T$ is obtained by solving a local Poisson problem (chapter 5 of [18]) [20].

$$\nu(\nabla \vec{e_T}, \nabla \vec{v})_T = (\vec{R_T}, \vec{v})_T - \sum_{E \in \varepsilon(T)} \langle \vec{R}_E^*, \vec{v} \rangle_E \ \text{ for all } \vec{v} \in \vec{\mathcal{Q}_T} \tag{20}$$

where

$$\vec{R}_E^* = \begin{cases} \frac{1}{2}\left[\nu \nabla \vec{u_h} - p_h \vec{I}\right] \ E \in \varepsilon_{h,\Omega} \\ -(\nu \frac{\partial \vec{u_h}}{\partial n_{E,T}} - p_h \vec{n}_{E,T}) \ E \in \varepsilon_{h,N} \\ 0 \ E \in \varepsilon_{h,D}, \end{cases} \tag{21}$$

known as the equidistributed flux jump operator and computed by sampling it at the mid-point of the edge,

$$\vec{R_T} = \{\vec{f} + \nu \nabla^2 \vec{u_h} - \vec{u_h}.\nabla \vec{u_h} - \nabla p_h\} \tag{22}$$

and $R_T = \{\nabla . \vec{u_h}\}|_T$. $\vec{R_T}$ and R_T are the interior residuals. The global error estimator is given as $\eta = (\sum_{\mathcal{T} \in \mathcal{T}_h} \eta_T^2)^{1/2}$. This scheme is for the Navier Stokes problem only implemented for the $Q1 - P0$ discretization.

7. Iterative Solver and Preconditioners

7.1. **Solvers.** The following solvers are implemented in IFISS

- Conjugate Gradient Method (CG) [10]: It is an effective method for symmetric positive definite systems. The computational work is two inner products, three vector updates and one matrix vector product. Increase in diagonal dominance leads to better convergence. The *pcg* built-in function of Matlab is used in IFISS.
- Minimum residual method (MINRES) [17]: Minres is a variant of the CG method satisfying optimality conditions. It is used when the matrix arises from discretization is symmetric indefinite. IFISS uses the matlab built-in routine *minres*.
- Generalized minimum residual method(GMRES) [15]: The GMRES method is designed to solve nonsymmetric linear systems. These methods have long recurrences and have certain optimality properties. It uses restarts to control storage requirement. If no restart, then it converges in n iterations. IFISS uses a modified routine written by Kelly [12].

- BICGSTAB(ℓ) [16]: The BiCGSTAB was developed to solve nonsymmetric linear systems while avoiding the often irregular convergence patterns of CGS method. In contrast to GMRES this algorithm does not satisfy an optimality condition but has fixed computational costs at each step. The BICGSTAB(ℓ) performs additional stabilization by minimizing over spaces of dimension ℓ greater than one. In practice $\ell = 2$ is usually sufficient to resolve the difficulties caused by complex eigenvalues. Table 3 shows iterative methods used in various problems with the different choices of preconditioners.

7.2. **Preconditioner.** The convergence rate of iterative methods depends on the spectral properties of the coefficient matrix. Hence, one may attempt to transform the linear system into one that is equivalent in the sense that it has same solution, but that has more favorable spectral properties. A *Preconditioner* is a matrix that effects such a transformation. If M approximates the coefficient matrix A in some way, the transformed system $M^{-1}Ax = M^{-1}b$
has the same solution as the original system $Ax = b$, but the spectral properties of its coefficient matrix $M^{-1}A$ may be more favorable. Some of the preconditioners implemented in IFISS are given below.

(1) Jacobi or Diagonal Preconditioner: In this case the matrix M is a diagonal matrix whose entries are equal to the system matrix.

$$m_{i,j} = \begin{cases} a_{i,j} \text{ if i=j} \\ 0 \text{ otherwise} \end{cases}$$

$M = diag(A)$, the computing action of M^{-1} involves n divisions. Such simple scaling is useful for stretched grids.

(2) Incomplete Cholesky and LU factorization [13]: For a symmetric system, incomplete Cholesky factorization is used, for which $U = L^T$. The basic principle is either preselect some sparsity pattern outside which any nonzero entry that would arise in L or U are dropped, or else to specify some threshold magnitude and drop all the entries in L or U that are smaller than this. The threshold is chosen such that the preconditioner is not too dense and leads to fast convergence. The common choice is to select the allowed fill to exactly match the sparsity pattern of original matrix of A.

(3) Geometric Multigrid [5]: The preconditioner consists of three steps.
   - The quickly oscillating component in the error is removed by iterative technique, this step is called pre-smoothing, In IFISS, ILU, Jacobi and Gauss Seidel methods are used for the pre-smoothing step.
   - The fine grid problem is projected on to a coarser grid to remove slowly varying component of the error.
   - High frequency modes are again removed with the help of the smoothing step.
     IFISS uses a V-cycle of GMG preconditioner.

(4) Algebraic Multigrid [5]: AMG is an extension of the classical idea of GMG (smoothing and coarse grid correction) to a certain class of algebraic systems of equations. AMG is independent of geometric information of the problem. IFISS code provides an interface to the algebraic multigrid solver of the commercial code FEMLAB, which enables the use of AMG as a preconditioner.

Preconditioners used for solving Stokes has the form $\mathbf{M} = \begin{bmatrix} P & 0 \\ 0 & T \end{bmatrix}$

and for Navier Stokes problems has the form $\mathbf{M} = \begin{bmatrix} P & B^T \\ 0 & T \end{bmatrix}$

(5) Ideal block Preconditioner: In ideal choice $P = A$, requires an exact solution of the Poisson equation for each of the velocity components and $T = Q_p$, Q_p is pressure mass matrix. In the case of a discontinuous pressure approximation like P_0 and P_{-1}, the corresponding mass matrix is diagonal. The idea of $T = diag(Q_p)$ still provides a uniform approximation to Q_p.

(6) Diagonal Block preconditioner: in this case $P = diag(A)$ and $T = diag(Q_p)$

(7) Stokes Multigrid preconditioner: In this case $P = A$ and P^{-1} is obtained by applying one GMG V-cycle to the Laplacian component, and $T = diag(Q_p)$

(8) Pressure Convection Diffusion preconditioner(PCD) [11], [14]: For a system as given in (9), $P = F$. The optimal choice for $T = -S$, where S is the Schur complement operator. In case of (9), $S = BF^{-1}B^T$. This approximation is not feasible as computation of M^{-1} is expensive. We replace S by its approximation $\tilde{S} = A_p F_p^{-1} Q$ where A_p is a pressure Laplacian matrix , F_p is the convection diffusion operator on pressure space and Q is mass matrix on pressure space.

(9) Least squares Commutator preconditioning(LSC) [6], [7]: in this case $P = F$ and $T = -S$, where S is approximated by
$S = (B\hat{Q}_v^{-1}B^T)(B\hat{Q}_v^{-1}F\hat{Q}_v^{-1}B^T)^{-1}(B\hat{Q}_v^{-1}B^T)$,
where $\hat{Q}_v$ is the diagonal of the velocity mass matrix. In the last two preconditioners, there is a choice to find inverse of the P with GMG and AMG approximations. For details on the last two preconditioners, see Appendix A.

Problem	CG	MINRES	GMRES	BICGSTAB(ℓ)	Preconditioner
Poission	$\odot$	$\odot$	–	–	$1, 2, 3$
Conv. Diff.	–	–	$\odot$	$\odot$	$1, 2, 3, 4$
Stokes	–	$\odot$	–	–	$5, 6, 7$
Navier Stokes	–	–	$\odot$	$\odot$	$8, 9$

TABLE 1. Iterative methods used for Problems in IFISS

8. Running the Software (IFISS) and Results

The IFISS package code is distributed in different sub directories to maintain modularity. The code distribution and installation can be seen in the IFISS manual [19]. Different driver functions are implemented for solving different category of problems. m files *diff_testproblem.m* for Diffusion Problems, *cd_testproblem.m* for convection diffusion *stoke_testproblem.m* for Stokes problem and *navier_testproblem.m* for solving Navier Stokes problems. A sample run is shown here, which has maximum options.

```
navier_testproblem

specification of reference Navier-Stokes problem.

choose specific example (default is cavity)
     1  Channel domain
     2  Flow over a backward facing step
     3  Lid driven cavity
     4  Flow over a plate
 : 2
        1 file(s) copied.
        1 file(s) copied.
horizontal dimensions [-1,L]: L? (default L=5) :


Grid generation for a step shaped domain.
grid parameter: 3 for underlying 8x4*(L+1) grid (default is 4) :
Q1-Q1/Q1-P0/Q2-Q1/Q2-P1: 1/2/3/4? (default Q1-P0) :
setting up Q1-P0 matrices...  done
system matrices saved in step_stokes_nobc.mat ...
Incompressible flow problem on step domain ...
viscosity parameter (default 1/50) :
Picard/Newton/hybrid linearization 1/2/3 (default hybrid) :
number of Picard iterations (default 2) :
number of Newton iterations (default 4) :
nonlinear tolerance (default 1.d-5) :
stokes system ...
Stokes stabilization parameter (default is 1/4) :
setting up Q1 convection matrix...  done.
computing Q1-P0 element stress flux jumps...  done
computing local error estimator... done.
estimated velocity error (in energy):  (5.608605e-001,2.358500e-001)
computing divergence of discrete velocity solution ...  done
estimated velocity divergence error:  1.806448e-002
plotting element data... done


initial nonlinear residual is 2.858319e+000
Stokes solution residual is 1.926740e+000

Picard iteration number 1
setting up Q1 convection matrix...  done.
nonlinear residual is 2.146864e-002
   velocity change is 2.264285e+000
```

```
Picard iteration number 2
setting up Q1 convection matrix...  done.
nonlinear residual is 7.301981e-003
   velocity change is 1.106196e+000

Newton iteration number 1
setting up Q1 Newton Jacobian matrices...  done.
setting up Q1 convection matrix...  done.
nonlinear residual is 7.020365e-004
   velocity change is 7.023304e-001

Newton iteration number 2
setting up Q1 Newton Jacobian matrices...  done.
setting up Q1 convection matrix...  done.
nonlinear residual is 1.155423e-006
   velocity change is 3.295976e-002

finished, nonlinear convergence test satisfied

computing Q1-P0 element stress flux jumps...  done
computing Oseen local error estimator... done.
estimated velocity error (in energy):  (7.225145e-001,3.127542e-001)
computing divergence of discrete velocity solution ...  done
estimated velocity divergence error:  1.585456e-002
estimated overall error is 7.874604e-001
plotting 2x2 element data... done

>> it_solve
inflow/outflow (step) problem ...

solving Jacobian system generated by solution from last Newton step
setting up Q1 Newton Jacobian matrices...  done.

GMRES/Bicgstab(2)  1/2 (default GMRES) :
stopping tolerance? (default 1e-6) :
maximum number of iterations? (default 100) :
preconditioner:
   0  none
   1  unscaled least-squares commutator (BFBt)
   2  pressure convection-diffusion (Fp)
   3  least-squares commutator
default is pressure convection-diffusion : 3
ideal / GMG iterated / AMG iterated  preconditioning? 1/2/3 (default ideal) :
ideal least-squares commutator preconditioning ...
GMRES iteration ...
convergence in  21 iterations

    k  log10(||r_k||/||r_0||)
    0           0.0000
    1          -0.0159
    2          -0.0740
    |
    |
```

```
    |
   16          -4.3919
   17          -4.7565
   18          -5.1692
   19          -5.5689
   20          -5.9791
   21          -6.3791
Bingo!

  1.0244e+001 seconds


use new (enter figno) or existing (0) figure, default is 0 :
figure number (default is current active figure) :
colour (b,g,r,c,m,y,k): enter 1--7 (default 1) :
>>
```

IFISS also provides a batchmode facility via which data may be input from a pre-prepared file rather than directly from the terminal. These problems are stored in the specified *test_problems* directory. For example type command *batchmode('P1')* will use *p1_batch.m* to generate and solve the discrete Poisson equation. To solve these problems in another domain, one has to change files containing domain and boundary conditions. In the whole process IFISS generates three plots containing grid, error and streamlines. The results are obtained through the Matlab sparse solver. If the user wants to solve it through an iterative solver, then one has to run an extra function *it_solve.m* which gives the number of iterations on the command window and plot of the residual. IFISS also has the facility to solve some of the problems with the Multigrid Solver *mg_solve.m*.
All the problems in IFISS are solved on a structured grid. Natural horizontal line ordering is used for nodes numbering. In the case of an L-shaped domain, the region $x < 0, 0 \leq y \leq 1$ is numbered first and then $x \geq 0, -1 \leq y \leq 1$. To deal with singularity and boundary layer, a strecthed grid option is available in most of the problems in the square domain.
To have an idea of how IFISS solves different kind of problems, Stokes and Navier Stokes results are plotted . Figure 1, 2 and 3(a) shows different plots associated with the solution of a problem. Tables 2-6 show mainly different results obtained from preconditioned solvers. The number of iterations of the solver is effected by many factors like type of problem, preconditioner, Reynolds number, mesh size, type of nonlinear iterations etc. We will discuss some of the factors here. Figure3(b) is obtained to have an idea of the convergence behavior of GMRES with different preconditioners. Table 2,3 and 4 give an idea of the preconditioned iterative solver used for solving Stokes problem, which shows that by using block diagonal preconditioner, the number of iterations rises with reduction in h. In case of Ideal and GMG preconditioner, the iterations remain almost unchanged for the specific problem. The number of iterations also varies with different discretization scheme. In case of the Navier Stokes problem, from Table 5, it is evident that an increase in Reynolds number increases the number of iterations for both preconditioners. From Table 6 , the iteration counts obtained from pressure convection diffusion preconditioner is almost independent of the grid size, with ten times increase in Reynolds number, the number of iterations become doubled. For the least square commutator, with smaller Reynolds number the iteration count increases with increase in h, however the dependence becomes less pronounced on higher Reynolds number. PCD can be used for both stable and unstable discretization. LSC can only be used for stable discretization. m-file *cg_test.m* gives you an idea of number of CG

iterations. With an artificial example, CG shows different convergence behavior for eigenvalues distribution as shown in Figure 6. CG converges fast for two distinct eigenvalues. The convergence of CG is fast if the eigenvalues are clustered in different groups as compared to the uniform or perfectly distributed eigenvalues.
For seeing the effect of grid stretching, we implemented the domain which can be strecthed and can get element with high aspect ratio. The stretched grid and results obtained in the grid are shown in Figure 5 and 6, respectively. The results in Table 7 shows that iterations remains almost same with ideal block preconditioner in Stokes problem. In case of Navier Stokes probelm in Table 8, the number of iterations increases with grid stretching, also the time consumed by an iterative solver to converge to the solution is large. This is the only last iteration which is applied to linearized problem, which shows that the both preconditioners are time and memory consuming for even a small Reynolds number.

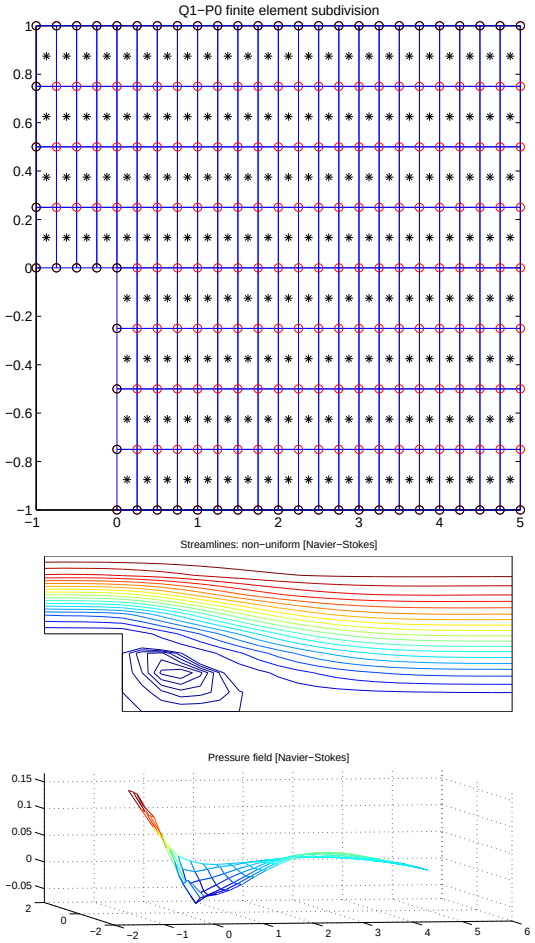

FIGURE 1. (a)-Grid for 8x4(L+1) divisions with L=5, (b)-Streamline plot(top) and pressure plot(bottom) of Navier Stokes equation with $Re = 100$

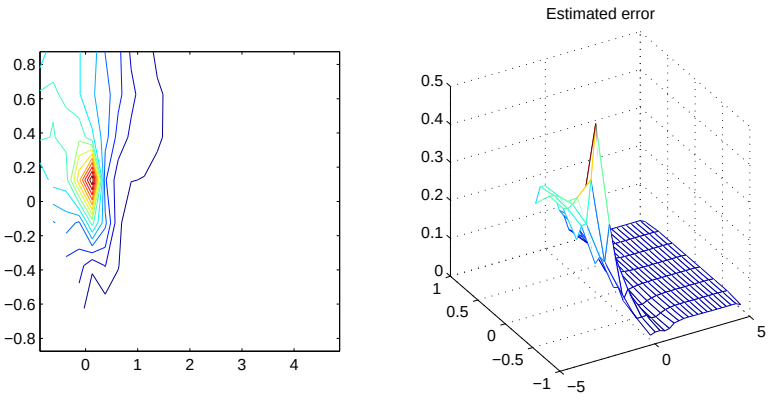

FIGURE 2. Estimated error in the computed solution

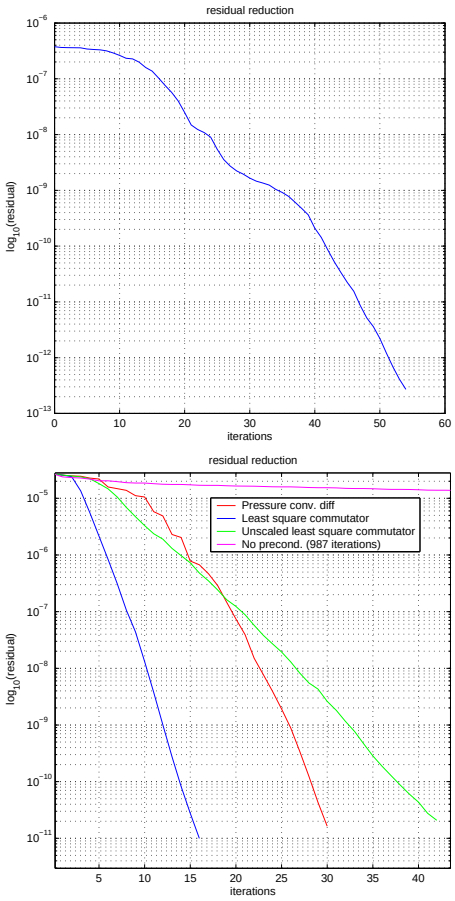

FIGURE 3. (a)-Plot for residual error, (b)-GMRES-Residual for Navier Stokes equation with $Re = 50$ using different preconditioner

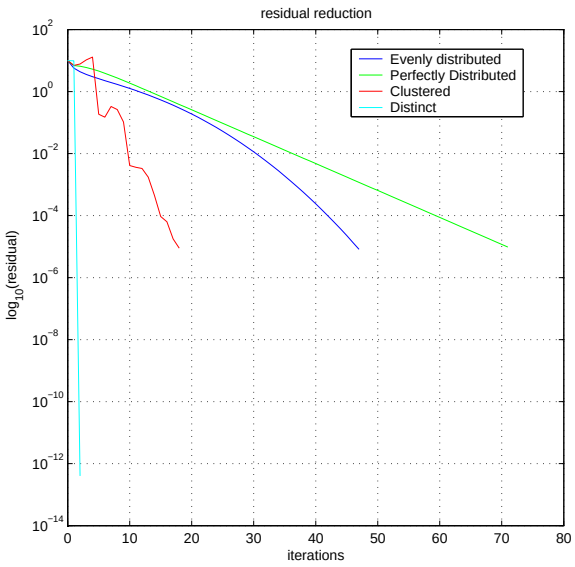

FIGURE 4. CG convergence analysis

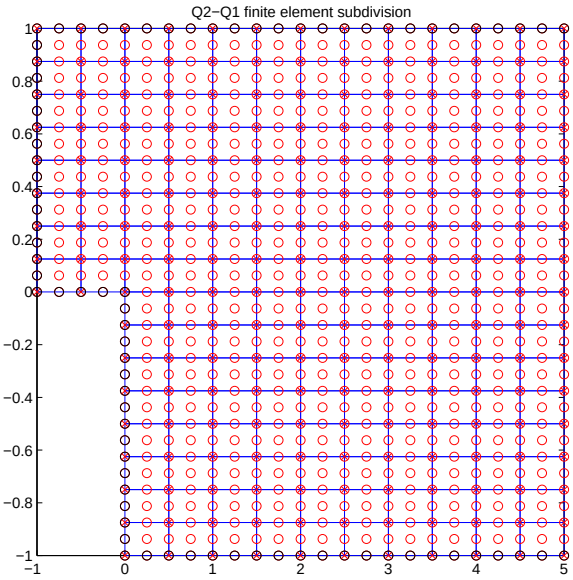

FIGURE 5. 32x24 Q2-Q1 Strecthed grid implemented in IFISS

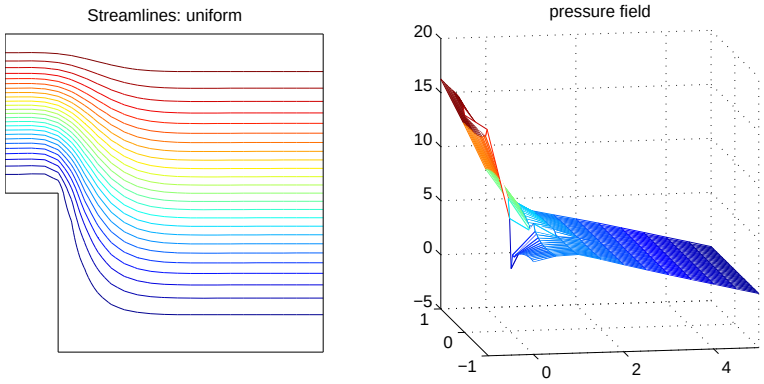

FIGURE 6. Streamlines and pressure calculated in stretched grid with $Re = 100$

Precond type	32x32 Iter/t(sec)	64x64 Iter/t(sec)	128x128 Iter/t(sec)
None	226/2.8	437/3.24	878/28.0
Block Diagonal	346/2.64	784/23.4	1922/252
Ideal Block	35/2.28	37/13.9	35/93
GMG	-	-	-

TABLE 2. No. of iteration for Stokes problem(channel flow) in square domain with $Q_1 - P_0$ discretization

Precond. Type	32x32 Iter/t(sec)	64x64 Iter/t(sec)	128x128 Iter/t(sec)
None	611/1.14	1121/9.57	1879/69
Block Diagonal	232/1.73	520/14.92	1409/176
Ideal Block	27/2.4	29/13.9	29/85
GMG	-	-	-

TABLE 3. No. of iteration for Stokes problem(channel flow) in square domain with $Q_2 - Q_1$ discretization

Precond. Type	32x96 Iter/t(sec)	64x192 Iter/t(sec)	128x384 Iter/t(sec)
None	1066/6.35	2029/52	3779/398
Block Diagonal	464/9	1090/90	2916/1017
Ideal Block	40/11	40/52	40/357
GMG	62/7	67/29	70/135

TABLE 4. No. of iteration for Stokes problem(Backward facing) in step domain with $Q_2 - Q_1$ discretization

Precond. Type	$Re = 50$ Iter/t(sec)	$Re = 150$ Iter/t(sec)	$Re = 250$ Iter/t(sec)
None	1017/65	1324/109	1483/136
Pressure Convec.Diff	40/18	61/27	84/35
Least Square Comm.	17/10	23/13	35/21

TABLE 5. Effect on iterations with varying Reynolds number for Navier Stokes problem in L-shaped domain with $Q_2 - Q_1$ discretization

Precond.	$Re = 10$			$Re = 100$		
Type	16x48	32x96	64x192	16x48	32x96	64x192
-	Iter./t (sec)	Iter./t (sec)	Iter./t (sec)	Iter./t (sec)	Iter./t (sec)	Iter./t (sec)
PCD	23/1.8	26/12	31/90	47/3.62	52/23	61/175
LSC	12/1.2	16/9.8	20/75	21/2.15	19/11	25/93

TABLE 6. Effect of grid increase and Reynolds number on iterations for Navier Stokes backward facing problem.

Grid Type	Ideal block Preconditioner Iter/t(sec)
8x24	28/0.43
16x24	30/0.74
32x24	30/1.74
64x24	30/7.25

TABLE 7. Effect of grid stretching on iterations with MINRES ($eps = 1e-4$) for Stokes problem in L-shaped domain with $Q_2 - Q_1$ discretization

$eps = 1e-6$	BiCGSTAB		GMRES	
	PCD	LSC	PCD	LSC
Grid	iter/time(s)	iter/time(s)	iter/time(s)	iter/time(s)
8x24	46/1.3	18/0.73	44/0.67	22/0.46
16x24	60/4.07	16/1.55	45/1.6	22/1.1
32x24	80/12.0	20/4.3	50/4.0	26/2.94
64x24	86/31.0	26/12.0	58/10.0	34/8.1

TABLE 8. Effect of grid stretching on iterations for Navier Stokes problem in L-shaped domain with $Q_2 - Q_1$ discretization and $Re = 100$

9. Conclusions

IFISS is a nice package to deal with basic problems in computational fluid dynamics. By using this software, one can get an idea of

- Finite element discretization of the different problems.
- Based on the coefficient matrix, a selection of the suitable iterative method.
- Effect of preconditioner on the convergence of iterative methods.
- Features associated with individual problem, like stable and unstable discretization in the Stokes and Navier Stokes problems etc.

Despite of having so many features, the following drawbacks were noted during exploring IFISS.

- Running problem with batchmode has problems with Matlab version below 7.0. As some of the Matlab built-in functions are not available in previous versions.
- There is no iterative solver available for solving Problem-1 and 4 of Navier Stokes domain.
- Stretched Grid facility is not available for varying length in Navier Stokes problem.
- In the non-linear iteration loop, there is a need to solve the linear problem with the help of an iterative solver.
- In the case of Stokes problems, Although the block diagonal preconditioner converges in less iterations as compared to using MINRES directly, but it takes more time, which does not justify the property of the preconditioner. In the case of ideal precontitioner, the time taken also suspects the credibility of the preconditioner.

References

[1] IFISS2.1 can be downloaded from website:<http://www.ma.umist.ac.uk/djs/ifiss/>

[2] School of Mathematics, University of Manchester, Manchester, UK. <na.silvester@na-net.ornl.gov.>

[3] Department of Computer Sciences, University of Maryland, College Park, Maryland, USA.<elman@cs.umd.edu>

[4] Department of Mathematics, University of Strathclyde, Glasgow, UK.<a.ramage@strath.ac.uk.>

[5] Briggs, W., Henson, V., and McCromick, S. 2000, *A multigrid tutorial*, SIAM, Philadelphia.

[6] Elman, H. C. 1999. Preconditioning for the steady-state Navier Stokes equations with low viscosity. *SIAM J. Sci. Comput. 20,* 1299-1316.

[7] Elman, H. C.,Howle, V.E., Shadid, J., Silvester, D., and Tuminaro, R. In preparation, 2006. *Block preconditioner based on approximate Commutators.* Tech. rep., Institute for Advanced Studies, University of Maryland.

[8] Elman, H. C. and Ramage, A. 2002. An analysis of smoothing effects of the upwinding strategies for the convection-diffusion equation.*SIAM, J. Numer. Anal. 40*, 254-281.

[9] Fischer, B., Ramage, A. Silvester, D., and Wathen, A. 1999. On parameter choice and iterative convergence for stabilized discretizations of the advection-diffusion problems.*Comput. Methods Appl. Mech. Eng. 179*, 185-202.

[10] Hestenes, M. R. and Stiefel, E. 1952. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of standards 49*, 49, 409-435.

[11] Kay, D., Loghin, D., and Wathen, A. 2002. A preconditioner for the steady state Navier Sotkes equations. *SIAM J. Sci. Comput. 24*, 237-256.

[12] Kelley, T. 1995.*Iterative Methods for linear and nonlinear equations*, SIAM, Philadelphia.

[13] Meijerink, J. A. and Van Der Vorst, H. A. 1977. An iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix. *Math. Comp. 31*, 148-162.

[14] Silvester, D., Elman, H., Kay, D., and Wathen, A. 2001. Efficient preconditioning of the linearized Navier-Stokes for the incompressible flow. *J. Comp. Appl. Math. 128*, 261-279.

[15] Saad, Y. and Schultz, M. H. 1986. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. State. Comput. 7*,856-869.

[16] Sleijpen, G. L. G. and Fokkema, D. R. 1993. BICGSTAB(ℓ) for linear equations involving unsymmetric matrices with complex spectrum.*Elec. Numer. Math. 1*, 11-32.

[17] Paige, C and Saunders, M. 1975. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal. 12*,617-629.

[18] Elman, H. C., Silvester, D., and Wathen, A. J. 2005. *Finite Elements and Fast iterative solvers with applications in incompressible fluids dynamics.* Oxford University Press, Oxfords, UK.

[19] Silvester, D., Elman, H. C., and Ramage, A. *Incompressible Flow Iterative Solution Software (IFISS), Installation and Software Guide.* version 2.1, released 28 September 2005.

[20] Kay, D. and Silvester, D. 1999. A posteriori error estimation for the stabilized mixed approximations of the Stokes equations.*SIAM J. Sci. Comput. 21,* 1321-1336.

[21] M. F. Murphy, G. H. Golub, And A J. Wathen , A note on preconditioning for indefinite linear systems.*SIAM J. Sci. Comput. 21(2000),* 1969-1972.

[22] Elman, H. C., Silvester, D. 1996. Fast nonsymmetric iterations and preconditioning for Navier-Stokes equations,*SIAM J. Sci. Comput., 17*,33-44.

Appendix A. Preconditioners forNavierr Stokes Equations

We will consider the preconditioner of the form

$$\mathbf{M} = \begin{bmatrix} M_F & 0 \\ 0 & M_S \end{bmatrix}, \tag{A-1}$$

for a coefficient matrix of the form

$$\mathbf{K} = \begin{bmatrix} F & B^T \\ B & 0 \end{bmatrix} \tag{A-2}$$

For the moment we take $M_F = F$ and explore what is needed for M_S. Intuitively, if $\mathbf{M}$ is chosen so that $\mathbf{M}^{-1}$ is an inexpensive approximate inverse of $\mathbf{K}$, then this might make a good preconditioner; however, it is not necessary for a good preconditioner to have that $\mathbf{M}^{-1}$ be an approximate inverse of $\mathbf{K}$. A sufficient condition for a good preconditioner is that the preconditioned matrix $\mathbf{M}^{-1}\mathbf{K}$ has a low degree minimum polynomial. We show how preconditioners can be derived for systems of the form (A-2) based upon exact preconditioner which yields a preconditioned matrix with exactly three or two distinct eigenvalues. To investigate the eigenvalues of the preconditioned system with the coefficient matrix (A-2) and preconditioner (A-1), we start with a generalized eigenvalue problem

$$\begin{bmatrix} F & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \lambda \begin{bmatrix} F & 0 \\ 0 & M_S \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix}$$

From the first row we get

$$Fu(1-\lambda) + B^T p = 0. \tag{A-3}$$

There are two possibilities : $\lambda = 1$ corresponds to eigenvector $[u\ 0]^T$ and in case of $\lambda \neq 1$, the equation (A-8) becomes

$$u = \frac{1}{\lambda - 1} F^{-1} B^T p \tag{A-4}$$

and then eliminating u from the second block equation gives

$$BF^{-1}B^T p = \mu M_S p, \tag{A-5}$$

where $\mu = \lambda(\lambda - 1)$. Thus the optimal choice is $M_S = S$, where

$$S = BF^{-1}B^T, \tag{A-6}$$

is the Schur complement operator. This means that $\mu = 1$ and there are precisely three eigenvalues. $\lambda = 1$, and $\lambda_{2,3} = \frac{1 \pm \sqrt{5}}{2}$. This preconditoning strategy would thus result in convergence to the solution in three GMRES steps. However it is not feasible to use the Schur complement operator because it is computationally expensive to calculate M_S^{-1}. So we seek good approximations of the Schur complement operator. Suppose that (A-5) arises from the Stokes equations, then best choice is $M_S = Q$ or even its diagonal, where Q is the pressure mass matrix. In this case μ will not depend on the discretization step size and λ will be either one or lie in two intervals , one right to unity and the other is left to zero. MINRES convergence will be fast. Suppose (A-5) arises from the Navier Stokes Equations. Then $M_S = 1/\mu Q$ and our goal should be to cluster eigenvalues μ in the small region in complex plane. If this is done then
$\lambda \leftarrow \frac{1 \pm \sqrt{1+4\mu}}{2}$,
The eigenvalues will be tightly clustered on each side of the imaginary axis.

THEOREM 1. *The eigenvalues of the generalized Schur complement (A-5) for the Oseen operator are contained in a rectangular box in the right half plane whose borders are bounded independently of h.*

for proof see [22].

COROLLARY 1. *The eigenvalues of the discrete Oseen operator preconditioned by (A-1) consist of $\lambda = 1$ of multiplicity $n_u - n_p$, together with four sets consisting*

of points of the form $1 + (a \pm bi)$ *and* $-a \pm bi$. *These sets can be enclosed in two rectangular regions that are symmetric with respect to* $Re(\lambda) = \frac{1}{2}$, *whose borders are bounded independenlty of* h. for more detail see [22].
In the case of a symmetric matrix it is a good choice to use block diagonal preconditioner as the preconditoned problem retains the symmetry. But in case of Navier Stokes, we have a nonsymmetric matrix and this restricts our options with respect to Krylov subspace iteration. We must give up either short recurrence or optimality. With either choice the advantage of the block diagonal is lost and it turns out that a slight variation on the structure of the preconditioner yields other improvements. In particular consider the block triangular matrix.

$$\mathbf{M} = \begin{bmatrix} M_F & B^T \\ 0 & -M_S \end{bmatrix} \tag{A-7}$$

assume $M_F = F$, the generalized eigenvalue problem is

$$\begin{bmatrix} F & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \lambda \begin{bmatrix} F & B^T \\ 0 & -M_S \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix}$$

From the first row we get

$$(1 - \lambda)(Fu + B^T p) = 0. \tag{A-8}$$

This gives $\lambda = 1$ and $Fu + B^T p = 0$ or $u = -F^{-1}B^T p$, substituting this into second equation gives

$$BF^{-1}B^T p = \lambda M_S p, \tag{A-9}$$

with $M_S = BF^{-1}B^T$, $\mu = \lambda$ in (A-5), the eigenvalues of the preconditioned operator derived from the block triangular preconditioner consists of unity together with the eigenvalues of (A-5). It will give a more rapid convergence as the eigenvalues of the preconditioned system lie on one side of the imaginary axis. The more general form of the coefficient matrix is given by

$$\mathbf{K} = \begin{bmatrix} F & B^T \\ B & -C \end{bmatrix} \tag{A-10}$$

We consider the block LU decomposition

$$\begin{bmatrix} F & B^T \\ B & -C \end{bmatrix} = \begin{bmatrix} I & 0 \\ BF^{-1} & I \end{bmatrix} \begin{bmatrix} F & B^T \\ 0 & -(BF^{-1}B^T + C) \end{bmatrix} \tag{A-11}$$

If the upper triangular matrix is viewed as preconditioner, then the eigenvalues of the preconditioned matrix are identically one, and this operator contains Jordan blocks of dimensions at most 2, and consequently that at most two iterations of a preconditioned GMRES iteration would be needed to solve the system, for any discretization and mesh size h and Reynolds number [21]. The preconditioning strategies are derived by making an approximation to an upper triangular matrix. Now $S = BF^{-1}B^T + C$, is the Schur complement of the stabilized discrete operator. We describe two ways to approximate the Schur complement operator, both of which can be used for M_S in the preconditioner. The good choice of M_S for Stokes problems is the pressure mass matrix Q. With choice $M_S = (\frac{1}{\nu}Q)$, this idea has also merits inNavierr-Stokes with low Reynolds number ($Re < 10$) and shows mesh independent convergence. However it does not take the effect of convection into account. The aim is to develop an approximation which reflects the balance of convection and diffusion in the problem and leads to improved convergence rates at high Reynolds numbers.

A.1. **Pressure Convection Diffusion equation.** We will start with an operator which contains as a component a discrete version of convection-diffusion operator.

$$\mathcal{L} = -\nu\nabla^2 + \vec{w_h}.\nabla. \tag{A-12}$$

Let us suppose that the commutator of the convection diffusion operator with the gradient operator is small in some sense:

$$\varepsilon = (-\nu\nabla^2 + \vec{w_h}.\nabla)\nabla - \nabla(-\nu\nabla^2 + \vec{w_h}.\nabla)_p, \tag{A-13}$$

The term with the p subscript indicates that the operator is defined on the pressure space. Let Q denote the velocity mass matrix. The matrix representation of the discrete gradient operator is $Q^{-1}B^T$, discrete negative divergence operator is $Q^{-1}B$ and for discrete convection diffusion operator is $Q^{-1}F$. The discrete operators in this case are scaled with velocity mass matrix Q. The discrete version of the commutator becomes

$$\varepsilon_h = (Q^{-1}F)(Q^{-1}B^T) - (Q^{-1}B^T)(Q^{-1}F_p) \tag{A-14}$$

premultiplying (A-14) by $BF^{-1}Q$ and postmultiplying by $F_p^{-1}Q$ and assuming that the commutator is small gives

$$BF^{-1}B^T \approx BQ^{-1}B^TF_p^{-1}Q, \tag{A-15}$$

Since $BQ^{-1}B^T$ is computationally expensive, it is replaced by its spectral equivalent matrix A_p known as the pressure Laplacian matrix, so

$$M_S = BF^{-1}B^T \approx A_pF_p^{-1}Q \tag{A-16}$$

With this choice of M_S, the preconditioner is known as the pressure convection diffusion Preconditoner. This preconditioner is also effective for stabilized approximations.

$$M_S = BF^{-1}B^T + C \approx A_pF_p^{-1}Q \tag{A-17}$$

For more details see(chapter 8 of [18]).

A.2. **The Least Square Commutator.** This approach for the Schur complement operator is only applicable when the mixed approximation is uniformly stable with respect to the inf-sup condition.

$$\min_{q_h \neq constant} \max_{\vec{v_h} \neq \vec{0}} \frac{|(q_h, \nabla.\vec{v_h})|}{\|\vec{v_h}\|_{1,\Omega}\|\vec{q_h}\|_{0,\Omega}} \geq \gamma \tag{A-18}$$

Rather than deriving an approximation to the Schur complement from the continuous version of the commutator, we will instead define an approximation to the matrix operator F_p that makes the discrete commutator small. The L_2 norm of the commutator, viewed as an operator defined on the pressure space M^h, is

$$\sup_{p_h \neq 0} \frac{\|[(-\nu\nabla^2 + \vec{w_h}.\nabla)\nabla - \nabla(-\nu\nabla^2 + \vec{w_h}.\nabla)_p]p_h\|}{\|p_h\|} \tag{A-19}$$

$$= \sup_{p \neq 0} \frac{\|[(Q^{-1}F)(Q^{-1}B^T) - (Q^{-1}B^T)(Q^{-1}F_p)]p\|_Q}{\|p\|_Q} \tag{A-20}$$

where $\|v\|_Q = \langle Qv, v\rangle^{1/2}$. We now try to construct F_p so that the norm is small. One way to do this is to minimize the individual vector norms of the columns of the discrete commutator one by one, that is by defining the jth column of F_p to solve the weighted least square problem.

$$min\|[Q^{-1}FQ^{-1}B^T]_j - Q^{-1}B^TQ^{-1}[F_p]_j\|_Q \tag{A-21}$$

The j in the expression $[Q^{-1}FQ^{-1}B^T]_j$ represents the jth column of the corresponding matrix. The associated normal equations are

$$Q^{-1}BQ^{-1}B^TQ^{-1}[F_p]_j = [Q^{-1}BQ^{-1}FQ^{-1}B^T]_j$$

which leads to the following definition of Fp:

$$F_p = Q(BQ^{-1}B^T)^{-1}(BQ^{-1}FQ^{-1}B^T)$$

Substituting this expression into (A-6) gives an approximation of the Schur complement matrix:

$$BF^{-1}B^T \approx (BQ^{-1}B^T)(BQ^{-1}FQ^{-1}B^T)^{-1}(BQ^{-1}B^T) \tag{A-22}$$

In contrast to the pressure convection diffusion, this preconditioner does not require the explicit construction of the matrix A_p and F_p. Also it is not practical to work with Q^{-1}, since it is a dense matrix. A Good idea is to replace Q by $\tilde{Q} = diag(Q)$. So (A-6) becomes

$$M_S = BF^{-1}B^T \approx (B\tilde{Q}^{-1}B^T)(B\tilde{Q}^{-1}F\tilde{Q}^{-1}B^T)^{-1}(B\tilde{Q}^{-1}B^T) \tag{A-23}$$

The main advantage of this preconditioner is that it is fully automated and it does not require the construction of operators A_p and F_p that is needed in the pressure convection diffusion preconditioner.