PRECONDITIONED SOLENOIDAL BASIS METHOD FOR INCOMPRESSIBLE FLUID FLOWS

A Thesis

by

XUE WANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2004

Major Subject: Computer Science

PRECONDITIONED SOLENOIDAL BASIS METHOD FOR INCOMPRESSIBLE FLUID FLOWS

A Thesis

by

XUE WANG

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Vivek Sarin (Chair of Committee)	Andreas Klappenecker (Member)
Hamn-Ching Chen (Member)	Valerie E. Taylor (Head of Department)

December 2004

Major Subject: Computer Science

ABSTRACT

Preconditioned Solenoidal Basis Method
for Incompressible Fluid Flows. (December 2004)
Xue Wang, B.E., South China University of Technology; M.E., Osaka University
Chair of Advisory Committee: Dr. Vivek Sarin

This thesis presents a preconditioned solenoidal basis method to solve the algebraic system arising from the linearization and discretization of primitive variable formulations of Navier-Stokes equations for incompressible fluid flows. The system is restricted to a discrete divergence-free space which is constructed from the incompressibility constraint. This research work extends an earlier work on the solenoidal basis method for two-dimensional flows and three-dimensional flows that involved the construction of the solenoidal basis P using circulating flows or vortices on a uniform mesh. A localized algebraic scheme for constructing P is detailed using mixed finite elements on an unstructured mesh. A preconditioner which is motivated by the analysis of the reduced system is also presented. Benchmark simulations are conducted to analyze the performance of the proposed approach.

To my family

ACKNOWLEDGMENTS

It would be impossible for me to finish this thesis without the kind help and encouragement that I have received from many people. To all of them I would like to offer my sincere and heartful thanks. And there are a special few whom I have to mention individually.

First of all, I am extremely grateful to my advisor, Dr. Vivek Sarin, for his invaluable suggestions, consistent guidance, support and patience throughout the development of this research and thesis. This thesis would not have been completed without his help.

I would also like to thank my committee members Dr. Hamn-Ching Chen and Dr. Andreas Klappenecker for their help, support and valuable suggestions. A special thank-you is offered to Dr. Bart Childs for his encouragement and help.

Great thanks are also offered to Hemant, Juttu, Radhika, Thomas, Kasturi and Meiqiu for their research discussions. A special thank-you goes to my friends Jie, Ingrid, Shanyn family and Kari family for making my stay in college station a pleasant experience.

Last but certainly not least, I would like to particularly take this opportunity to thank my family for all their support, encouragement and love throughout my graduate study. I also thank my boyfriend Jianjun for his patience and inspiration.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Advances in computing have made it possible to simulate the fluid flows using computational fluid dynamics(CFD). Realistic simulation of fluid flows requires heavy computational power to solve the nonlinear system. By using operator-splitting techniques, the nonlinear system is reduced to a constrained linear system, which must be solved alternately with an unconstrained nonlinear system. This research work presents an effective approach to solve the linear system in incompressible fluid flows for 2D $P2/P1$ triangular element meshes using mixed finite element schemes.

A. Incompressible Flow Governing Equations

The Navier-Stokes equations (NSE) have been regarded as the fundamental governing equations for Newtonian incompressible viscous fluid flows for over 150 years. Such incompressible potential flows are important both in fluid mechanics and in heat or mass transfer. The flow is governed by the following Navier-Stokes equations

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} - \frac{1}{Re}\nabla^2\mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla p = 0 \qquad & in \quad \Omega, \\ \nabla\cdot\mathbf{u} = 0 \qquad & in \quad \Omega, \quad \text{(incompressibility condition)} \\ \mathbf{u} = \mathbf{g} \qquad & on \quad \partial\Omega, \end{aligned} \tag{1.1}$$

where Ω and $\partial\Omega$ denote the region of the flow and its boundary, respectively. We use standard notation, in (1.1), $\nabla = \{\frac{\partial}{\partial x_i}\}_{i=1}^2$, $\mathbf{u} = \{u_i\}_{i=1}^2$ is the flow velocity, p is pressure, and Re is the Reynolds number.

This thesis follows the style and format of *IEEE Transactions on Software Engineering*.

To incorporate the boundary condition from the incompressibility constraint of the fluid, the given function $\mathbf{g}$ has to satisfy

$$\int_{\partial\Omega} \mathbf{g} \cdot \mathbf{n} d\Omega = 0, \tag{1.2}$$

where $\mathbf{n}$ is the outward unit vector normal to $\partial\Omega$. Finally, for the time dependent problem, an initial condition such as

$$\mathbf{u}(x, 0) = \mathbf{u}_0(x) \quad on \quad \Omega, \tag{1.3}$$

for any given $\mathbf{u}_0$, is usually prescribed and assumed to satisfy the incompressibility constraint, i.e., $\nabla \cdot \mathbf{u}_0 = 0$.

The difficulties with the Navier-Stokes equations are the nonlinear terms $\mathbf{u} \cdot \nabla \mathbf{u}$, and the incompressibility condition. By using convenient operator-splitting techniques for the time discretization of the Navier-Stokes equations, these difficulties will be decoupled. The following algorithm [9] shows the simplest two-stage operator-splitting scheme

$$\begin{aligned} \frac{\mathbf{u}^{n+\theta} - \mathbf{u}^n}{\theta \Delta t} - \alpha \frac{1}{Re} \nabla^2 \mathbf{u}^{n+\theta} + \mathbf{u}^* \cdot \nabla \mathbf{u}^{n+\theta} &= \beta \frac{1}{Re} \nabla^2 \mathbf{u}^n - \nabla p^n \qquad in \quad \Omega, \\ \mathbf{u}^{n+\theta} &= \mathbf{g}^{n+\theta} \qquad on \quad \partial\Omega, \end{aligned} \tag{1.4}$$

$$\begin{aligned} \frac{\mathbf{u}^{n+1} - \mathbf{u}^{n+\theta}}{(1-\theta)\Delta t} - \beta \frac{1}{Re} \nabla^2 \mathbf{u}^{n+1} + \nabla p^{n+1} &= \alpha \frac{1}{Re} \nabla^2 \mathbf{u}^{n+\theta} - \mathbf{u}^* \cdot \nabla \mathbf{u}^{n+\theta} \qquad in \quad \Omega, \\ \nabla \cdot \mathbf{u}^{n+1} &= 0 \qquad in \quad \Omega, \\ \mathbf{u}^{n+1} &= \mathbf{g}^{n+1} \qquad on \quad \partial\Omega, \end{aligned} \tag{1.5}$$

with given $\mathbf{u}^0$, $\theta \in [0, 1]$, $\alpha \in (0, 1)$, and $\beta \in (0, 1)$. The natural choice of $u^* = u^n$ gives a linear method but in this case the accuracy is only first order. Second order

accuracy can be achieved by setting $u^* = u^{n+\theta}$. Through this kind of operator-splitting scheme, the nonlinear system gets reduced to a constrained linear system, called the generalized Stokes problem(1.5), which must then be solved alternatively with an unconstrained nonlinear system. We use the following notation to express the generalized Stokes equations:

$$\begin{aligned} \alpha\mathbf{u} - \frac{1}{Re}\nabla^2\mathbf{u} + \nabla p = \mathbf{f} \qquad & in \quad \Omega, \\ \nabla \cdot \mathbf{u} = 0 \qquad & in \quad \Omega, \\ \mathbf{u} = \mathbf{g} \qquad & on \quad \partial\Omega \quad , \end{aligned} \tag{1.6}$$

where $\alpha = \frac{1}{\Delta t}$ is positive parameter, $\mathbf{f}$ is the force coming from convection term.

In matrix notation, after suitable discretization and linearization, (1.6) can be represented as the following linear system:

$$\begin{bmatrix} A & B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ p \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix}, \tag{1.7}$$

where B^T is the discrete divergence matrix, which ensures the constraint of discrete null divergence on the solution vector $\mathbf{u}$, and A is the matrix arising from the terms with $\mathbf{u}$:

$$A = \alpha M + \frac{1}{Re}L, \tag{1.8}$$

in which M is the $n \times n$ mass matrix, L is a symmetric positive definite matrix corresponding to the Laplace operator. For a system with n velocity unknowns and m pressure unknowns, $A \in R^{n\times n}$, $B \in R^{n\times m}$, $u \in R^n$, and $p \in R^m$.

The linear system (1.7) is large, sparse, and is a saddle-point problem. Although the matrix A is symmetric and positive definite, the linear system is indefinite because of the incompressibility constraints $B^T u = 0$. The indefiniteness is the main cause of

difficulty for iterative methods and preconditioning techniques.

B. Iterative Methods

The methods commonly used to solve the linear system in equation (1.7) can be broadly classified as Uzawa-type methods [3][6][10], preconditioned Krylov subspace methods [12][15], and projection based methods. Uzawa-type methods employ block elimination to obtain a reduced system for pressure unknowns, which is then solved via iterative methods.

$$B^T A^{-1} B p = B^T A^{-1} f. \tag{1.9}$$

Preconditioned Krylov subspace methods are a general class of techniques that employ well known iterative methods with commonly used preconditioners. A good preconditioner is a matrix M whose inverse is a close approximation to the inverse of A. Instead of $Ax = b$, we solve the system $M^{-1}Ax = M^{-1}b$. The most popular class of preconditioners utilize the incomplete LU factorizations that form the matrix M by constructing sparse approximations of the LU factors of A. Projection based methods solve a reduced problem in a subspace of divergence-free fluid velocity via iterative methods. Solenoidal basis method [13][17] belongs to this class of methods.

The solenoidal basis method is a projection technique that uses a discrete divergence-free basis to represent velocity. This has the advantage of automatically satisfying the incompressibility condition constraint. A hierarchical algebraic scheme to construct the solenoidal basis was developed in [16]. The local divergence free basis introduced in [17][13] for 2D, 3D MAC scheme leads to the construction of a very effective and robust preconditioner for the linearized system.

Since the projection basis, P, spans the null space of B^T, a divergence-free velocity vector can be expressed as $u = Px$ for any arbitrary vector x. Such a velocity

vector automatically satisfies the incompressibility constraints $B^T u = B^T P x = 0$. As a result, the linear system (1.7) reduces to

$$APx + Bp = f. \tag{1.10}$$

After multiplying P^T to both sides of equation (1.10), the system is transformed into a reduced system

$$P^T APx = P^T f, \tag{1.11}$$

which must then be solved for x. The reduced system (1.11) is solved by an appropriate iterative method such as the conjugate gradient (CG) [8] or generalized minimum residual method (GMRES) [11]. Velocity is recovered by computing $u = Px$ and pressure is obtained by solving the least square problem $Bp = f - APx$.

Since B is large and sparse, it is very expensive to compute a QR factorization to obtain the null space matrix P. The convergence of the iterative method for the reduced system (1.11) depends on the choice of P. The basis P should be computed efficiently, and the choice of P must allow effective preconditioning of the reduced system $P^T AP$, so that the system will converge to a solution faster.

In this research, we present techniques to construct the solenoidal basis and the preconditioner for the reduced system arising from mixed finite element discretization scheme. We extend an earlier work on the solenoidal basis method for two-dimensional flows [17] and three-dimensional flows [13] that involved construction of the solenoidal basis P using circulating flows or vortices on an uniform mesh. We also present effective preconditioning techniques for the reduced system that are motivated by the analysis of the reduced system.

The thesis is organized as follows. Chapter II gives brief introduction of solenoidal basis method, and the construction of local solenoidal basis using 2D and 3D Marker-

and-Cell scheme of discretization on a uniform mesh. In chapter III, we extend the approach presented in [17][14] to a 2D $P2/P1$ unstructured mesh using the finite element scheme. We outline the details of constructing local solenoidal basis for this scheme. We also propose a preconditioner to solve the reduced system. Chapter IV shows the experimental results of test problems using the scheme given in chapter III. Finally, chapter V provides a concluding summary of the research work.

CHAPTER II

SOLENOIDAL BASIS METHOD FOR MARKER-AND-CELL SCHEME

In this chapter, the introduction of solenoidal basis method for Marker-and-Cell scheme is presented. The construction of local solenoidal basis is given in both 2D MAC scheme and 3D MAC scheme. An effective preconditioner is also given after analyzing the structure of divergence operator matrix B^T. The effectiveness of preconditioned solenoidal basis method on uniform meshes using MAC scheme motivates the extension of a similar methodology to unstructured meshes in later chapter.

A. 2D Marker-and-Cell Scheme

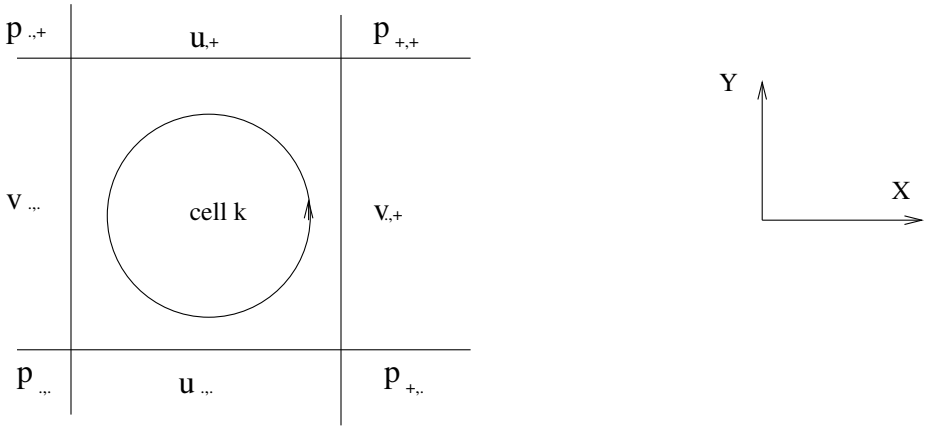

Fig. 1. Local solenoidal flows for Marker-and-Cell scheme in two dimensional domains

The simplest approach to construct a local solenoidal basis is to generate circulating flows in local regions of the mesh such that each flow satisfies the divergence constraint. In [17] Sarin presented solenoidal basis method using Marker-and-Cell scheme for discretization on 2D uniform mesh, and outlined an optimal preconditioning technique for the generalized Stokes problem. Pressure unknowns are assigned to the grid nodes and velocity unknowns are assigned to the midpoints of edges. As

shown in Figure 1, a local solenoidal flow is obtained by assigning unit velocity to the edges of a cell in a mesh oriented in anti-clock wise direction, where $u_{.,.} = 1$, $u_{.,+} = -1$, $v_{.,.} = -1$, and $v_{.,+} = 1$. Since the net outflow at any pressure node is zero, it ensures the flow is solenoidal. For example, after scaling with h, the row corresponding to the node $p_{.,.}$ in Figure 1 has the nonzeros [-1 1 -1 1] that multiply with $[u_{-,.}\ u_{.,.}\ v_{.,-}\ v_{.,.}]$ to enforce the divergence-free condition.

$$(u_{.,.} - u_{-,.}) + (v_{.,.} - v_{.,-}) = 0 \tag{2.1}$$

The solenoidal flows can be constructed algebraically after analyzing the structure of the discrete divergence operator matrix B^T. The submatrix with rows corresponding to pressure nodes $[u_{.,.}\ u_{+,.}\ v_{.,.}\ v_{.,+}]$, and columns corresponding to velocity edges $[u_{.,.}\ u_{+,.}\ v_{.,.}\ v_{.,+}]$, is

$$\left[B^T\right]_k = \begin{array}{c} \\ p_{.,.} \\ p_{+,.} \\ p_{.,+} \\ p_{+,+} \end{array} \begin{array}{c} \begin{array}{cccc} u_{.,.} & u_{+,.} & v_{.,.} & v_{.,+} \end{array} \\ \begin{bmatrix} 1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 0 & -1 \end{bmatrix} \end{array}. \tag{2.2}$$

At each pressure node, incoming edges have a value of -1 and outgoing edges have a value of 1. The null space of $[B^T]_k$ is vector

$$[P]_k = \begin{bmatrix} 1 & -1 & -1 & 1 \end{bmatrix}^T, \tag{2.3}$$

which is identical to the discrete local solenoidal flow shown in Figure 1. $[P]_k$ represents solenoidal function local to cell k. By adding zeros to expand the local null space vector $[P]_k$ into a global null space vector P_i with length of n, the divergence

constraint $B^T P_i = 0$ can be satisfied. The discrete solenoidal basis P is the set of vectors P_i for all cells in the grid.

In order to better understand the structure of $P^T AP$, we need to first analyze the structure of $P^T P$. The matrix-vector product $y = P^T u$ is analyzed as follows. Since the kth column of P is a local solenoidal flow along the edges of the kth cell, the inner product of u with this column is the discrete curl of the flow in the cell k. The product $y = P^T u$ computes the curl of the flow represented by u at the center of each cell. The matrix-vector product $u = Px$ computes the discrete curl of a suitable function. Thus the product $y = P^T Pw$ represents $\bigtriangledown \times \bigtriangledown \times w$ in a discrete setting. After simplification, it shows that the matrix $P^T P$ is equivalent to the Laplace operator on the solenoidal function space. The reduced system for the generalized Stokes problem can be approximated by

$$P^T AP \approx \left[\frac{1}{\Delta t} M + \frac{1}{Re} P^T P\right] P^T P, \tag{2.4}$$

and the corresponding preconditioner is defined as

$$G_p = \left[\frac{1}{\Delta t} M + \frac{1}{Re} L_s\right] L_s, \tag{2.5}$$

where L_s is the Laplace operator for the local solenoidal functions. The preconditioned system is equivalent to a symmetric positive definite matrix, which can be solved via CG method. The experimental results show that the rate of convergence of the iterative method for the preconditioned CG method is nearly independent of the problem parameters such as the mesh width, time step and viscosity.

B. 3D Marker-and-Cell Scheme

In [14], Sambavaram extended the approach presented in [17] to 3D Marker-and-Cell scheme. The solenoidal flows are defined on each face of a cubic mesh cell(see Figure 2). Pressure unknowns are assigned to grid nodes and velocity unknowns are assigned to the mid-points of edges. The construction of local solenoidal functions is identical to that constructed for the 2D Marker-and-Cell scheme.

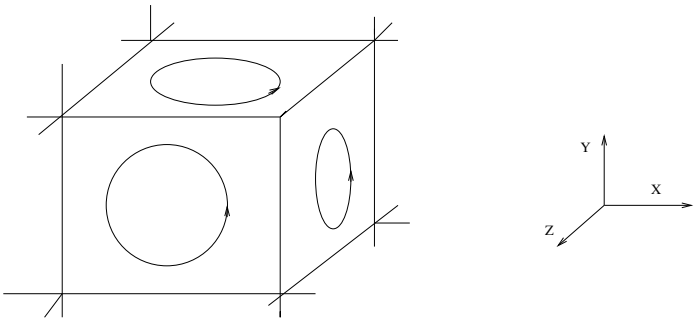

Fig. 2. Local solenoidal flows for Marker-and-Cell scheme in three dimensional domains. Only three such flows are shown

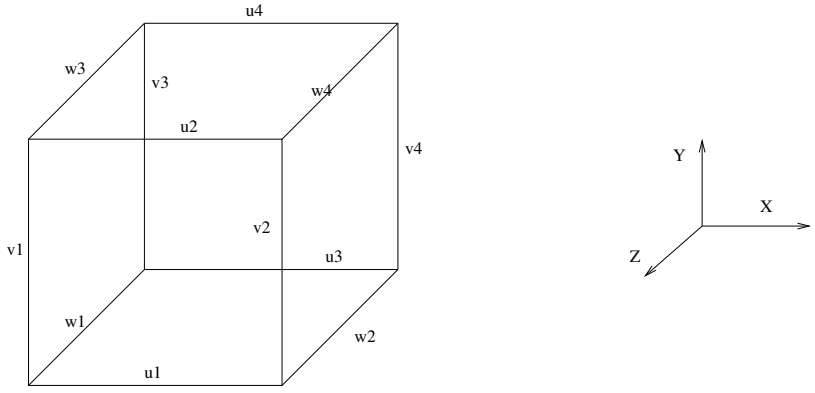

Fig. 3. Velocity nodes on 3D Marker-and-Cell

The velocity assignment for the solenoidal functions for each face in the 3D

Table I. Solenoidal functions for the 3D cell shown in Figure 3

	Front	Back	Left	Right	Bottom	Top
u_1	1	0	0	0	-1	0
u_2	-1	0	0	0	0	1
u_3	0	-1	0	0	1	0
u_4	0	1	0	0	0	-1
v_1	-1	0	1	0	0	0
v_2	1	0	0	-1	0	0
v_3	0	1	-1	0	0	0
v_4	0	-1	0	1	0	0
w_1	0	0	1	0	-1	0
w_2	0	0	0	-1	1	0
w_3	0	0	-1	0	0	1
w_4	0	0	0	1	0	-1

cell, as shown in Figure 3, is given in Table I. The six cell flows are not linearly independent. Solenoidal flow on the top face can be constructed from the remaining flows. It is easily to be observed that the front face solenoidal flow of the 3D neighbor cell which is behind the current cell is exactly the negative of the current cell's back solenodial flow. Within the whole domain, the adjacent cells have same solenoidal flow with opposite direction on the sharing faces. Hence, a linearly independent solenoidal basis consists of all front and left solenoidal flows for each cell, and those lie on the bottom, right and back boundaries. The preconditioner for the iterative method is the same as that for 2D MAC scheme, and the experimental results show that the preconditioner ensures a stable rate of convergence independent of problem

parameters such as mesh width, time step and viscosity.

CHAPTER III

SOLENOIDAL BASIS METHOD FOR MIXED FINITE ELEMENT METHOD

The finite element method [4] is an approximate method of solving differential equations of boundary and/or initial value problems in engineering and mathematical physics. In this method, the domain Ω is discretized into many small elements $\Omega_e,\ e = 1, 2, \cdots, Ne$(Ne is number of elements) of convenient shapes − triangles, quadrilaterals, etc. Choosing suitable points within the elements, the variable in the differential equation is written as a linear combination of appropriately selected interpolation functions and the values of the variable or its various derivatives are specified at the nodes. Using variational principles or weighted residual methods, the governing differential equations are transformed into "finite element equations" governing all isolated elements. These local elements are finally collected together to form a global system of differential or algebraic equations with proper boundary and/or initial conditions imposed. The nodal values of the variable are determined from this system of equations.

A. Local Solenoidal Basis

1. Interpolation and Finite Element Discretization

For the finite element solution of the equation (1.7), the domain and the discrete solution variables are commonly discretized using $P1$ *iso*-$P1$ triangular element, which uses linear shape functions for the velocity and pressure, and $P2/P1$ triangular element, which uses quadratic functions for velocity and linear functions for pressure. In [14], Sambavaram presented the technique to construct local solenoidal basis using $P1$ *iso*-$P1$ triangular element(see Figure 4). The basis functions are given by

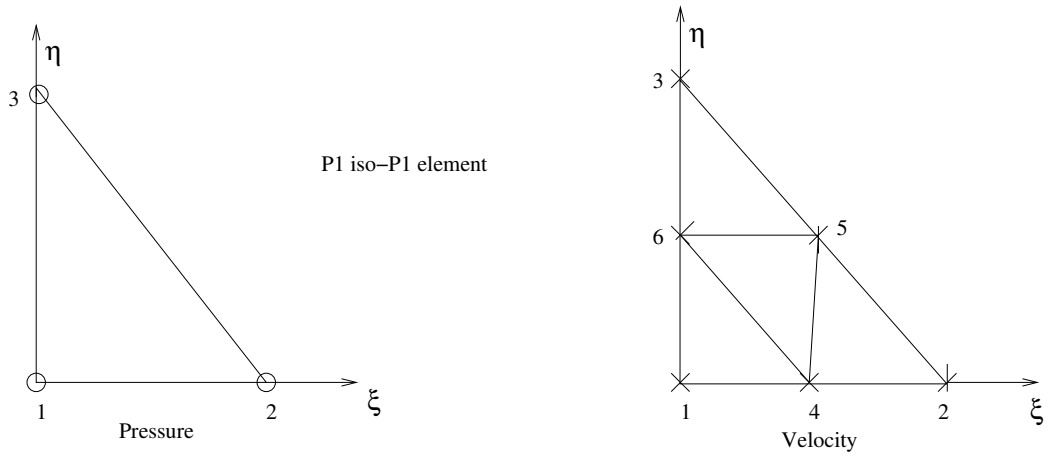

Fig. 4. Degrees of freedom in P1 iso-P1 element

$$\begin{aligned} \phi_1(\xi,\eta) &= 1-\xi-\eta, \\ \phi_2(\xi,\eta) &= \xi, \\ \phi_3(\xi,\eta) &= \eta. \end{aligned} \tag{3.1}$$

In this research work, we are focusing on construction of local solenoidal basis using $P2/P1$ triangular element(see Figure 5), where the discrete velocity solution u, v are piecewise quadratic and continuous all over Ω. On each triangle, there is one velocity basis function per vertex, and one velocity basis function per midside node. The discrete pressure solution p is interpolated using a linear polynomial. On each triangle, it is determined by its values at the three vertices,

$$\begin{aligned} u(x,y) &= \sum_{j=1}^{n} u_j \varphi_j, \\ v(x,y) &= \sum_{j=1}^{n} v_j \varphi_j, \\ p(x,y) &= \sum_{j=1}^{m} p_j \phi_j, \end{aligned} \tag{3.2}$$

where φ_j are the basis functions for velocities, ϕ_j are basis functions for pressure,

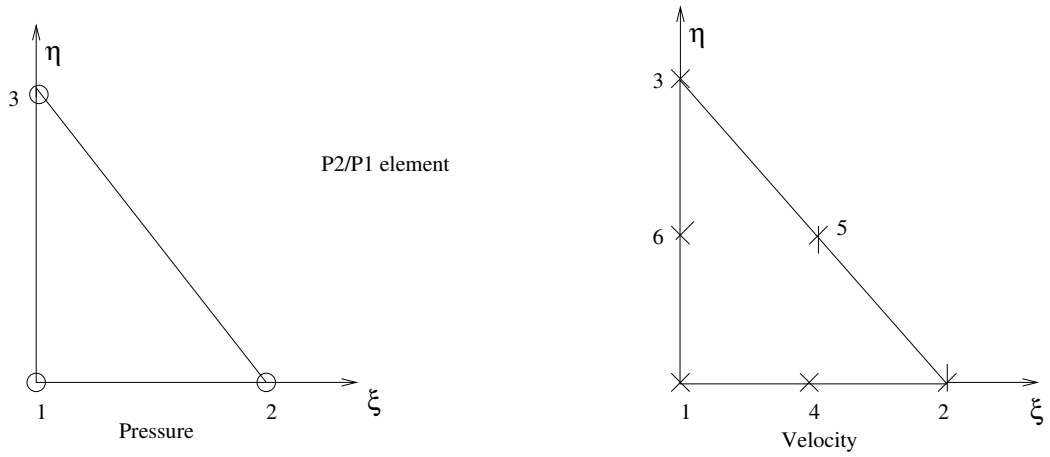

Fig. 5. Degrees of freedom in P2/P1 element

u_j, v_j are the degrees of freedom of u, v, p_j are the degrees of freedom of p, n is the number of velocity nodes, and m is the number of pressure nodes.

Since we have to integrate quadratic polynomials, it appears easy to use the same $\hat{\Omega}$ for all triangles Ω_e in the mesh for function integration. The common choice of $\hat{\Omega}$ is shown in Figure 6.

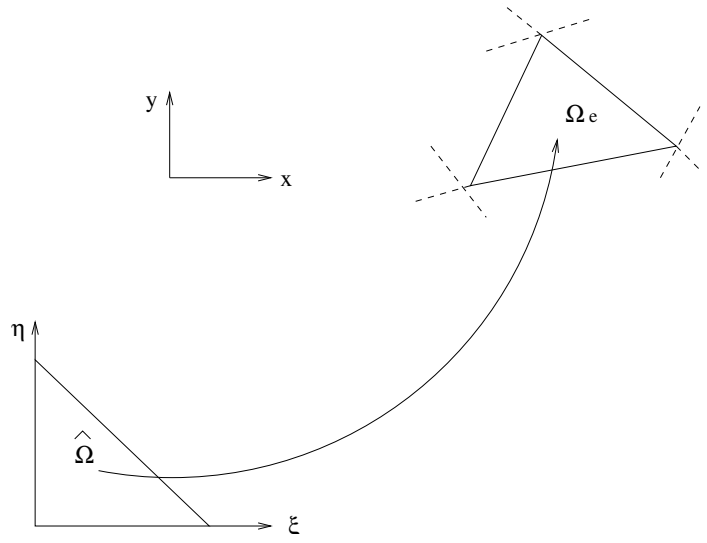

Fig. 6. Master element $\hat{\Omega}$ and corresponding element Ω_e

Let $\hat{\varphi}_i = \hat{\varphi}_i(\xi,\eta)$, $\hat{\phi}_i = \hat{\phi}_i(\xi,\eta)$ be the polynomial basis on the master element $\hat{\Omega}$ and $x_e(\xi,\eta)$, $y_e(\xi,\eta)$ the transformation from (ξ,η) coordinates for $\hat{\Omega}$ to (x,y) coordinates for Ω_e (Figure 6). The map is usually defined parametrically using the element basis functions,

$$\begin{aligned} x_e(\xi,\eta) &= \sum_{i=1}^{6} \hat{\varphi}_i x_i^e, \\ y_e(\xi,\eta) &= \sum_{i=1}^{6} \hat{\varphi}_i y_i^e, \end{aligned} \tag{3.3}$$

where (x_i^e, y_i^e) are nodal coordinates defining Ω_e.

For the quadratic velocities, the basis functions on the master element $\hat{\Omega}$ are given by

$$\begin{aligned} \hat{\varphi}_1(\xi,\eta) &= 2(1-\xi-\eta)(\frac{1}{2}-\xi-\eta), \\ \hat{\varphi}_2(\xi,\eta) &= 2\xi(\xi-\frac{1}{2}), \\ \hat{\varphi}_3(\xi,\eta) &= 2\eta(\eta-\frac{1}{2}), \\ \hat{\varphi}_4(\xi,\eta) &= 4(1-\xi-\eta)\xi, \\ \hat{\varphi}_5(\xi,\eta) &= 4\xi\eta, \\ \hat{\varphi}_6(\xi,\eta) &= 4\eta(1-\xi-\eta), \end{aligned} \tag{3.4}$$

and for the linear interpolation of the pressure on the master element is given by

$$\begin{aligned} \hat{\phi}_1(\xi,\eta) &= 1-\xi-\eta, \\ \hat{\phi}_2(\xi,\eta) &= \xi, \\ \hat{\phi}_3(\xi,\eta) &= \eta. \end{aligned} \tag{3.5}$$

The derivatives of shape functions in (x,y) coordinates can be transformed using the

chain rule. Introducing the Jacobian matrix

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix},$$

with

$$\begin{aligned} \frac{\partial x}{\partial \xi} &= \sum_{j=1}^{6} x_j^e \frac{\partial \hat{\varphi}_j}{\partial \xi}, & \frac{\partial y}{\partial \xi} &= \sum_{j=1}^{6} y_j^e \frac{\partial \hat{\varphi}_j}{\partial \xi}, \\ \frac{\partial x}{\partial \eta} &= \sum_{j=1}^{6} x_j^e \frac{\partial \hat{\varphi}_j}{\partial \eta}, & \frac{\partial y}{\partial \eta} &= \sum_{j=1}^{6} y_j^e \frac{\partial \hat{\varphi}_j}{\partial \eta}, \end{aligned} \tag{3.6}$$

the derivative operators can be expressed as

$$\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix}. \tag{3.7}$$

The elemental area in Cartesian coordinates (x,y) can be expressed in terms of the area in the local coordinates (ξ, η) as

$$dxdy = |\mathbf{J}| d\xi d\eta, \qquad |\mathbf{J}| = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}, \tag{3.8}$$

where $|\mathbf{J}|$ is the determinant of the Jacobian. In order to obtain the finite element formulation of equations(1.7), an arbitrary velocity basis function φ_i, and an arbitrary pressure basis function ϕ_i are multiplied and integrated into the domain:

$$\begin{aligned} \int_\Omega (\frac{\partial u}{\partial t} - \nu \nabla^2 u + \frac{\partial p}{\partial x}) \varphi_i dxdy &= \int_\Omega f_1 \cdot \varphi_i dxdy, \\ \int_\Omega (\frac{\partial v}{\partial t} - \nu \nabla^2 v + \frac{\partial p}{\partial y}) \varphi_i dxdy &= \int_\Omega f_2 \cdot \varphi_i dxdy, \\ \int_\Omega (\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}) \phi_i dxdy &= 0, \end{aligned} \tag{3.9}$$

where ν is the viscosity. Applying the divergence theorem on the diffusive term and

setting the boundary integral to zero, we obtain

$$\begin{aligned}
\int_\Omega \frac{\partial u}{\partial t}\cdot\varphi_i dxdy+\nu\int_\Omega(\frac{\partial u}{\partial x}\frac{\partial \varphi_i}{\partial x}+\frac{\partial u}{\partial y}\frac{\partial \varphi_i}{\partial y})dxdy-\int_\Omega p\frac{\partial \varphi_i}{\partial x}dxdy &= \int_\Omega f_1\cdot\varphi_i dxdy,\\
\int_\Omega \frac{\partial v}{\partial t}\cdot\varphi_i dxdy+\nu\int_\Omega(\frac{\partial v}{\partial x}\frac{\partial \varphi_i}{\partial x}+\frac{\partial v}{\partial y}\frac{\partial \varphi_i}{\partial y})dxdy-\int_\Omega p\frac{\partial \varphi_i}{\partial y}dxdy &= \int_\Omega f_2\cdot\varphi_i dxdy,\\
\int_\Omega(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y})\phi_i dxdy &= 0. \qquad (3.10)
\end{aligned}$$

The resulting semi-discrete finite element system has the form

$$\begin{bmatrix} \hat{M} & 0 & 0\\ 0 & \hat{M} & 0\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{p}\end{bmatrix}+\begin{bmatrix} \nu\hat{A} & 0 & B^x\\ 0 & \nu\hat{A} & B^y\\ (B^x)^T & (B^y)^T & 0\end{bmatrix}\begin{bmatrix}u\\ v\\ p\end{bmatrix}=\begin{bmatrix}F_1\\ F_2\\ 0\end{bmatrix}, \qquad (3.11)$$

where

$$[\dot{u}\quad \dot{v}\quad \dot{p}]\equiv d/dt[u\quad v\quad p],\qquad \hat{M}=\int_\Omega \varphi_i\varphi_j dxdy, \qquad (3.12)$$

$$\hat{A}=\int_\Omega((\varphi_i)_x(\varphi_j)_x+(\varphi_i)_y(\varphi_j)_y)dxdy,\qquad F_k=\int_\Omega f_k(x)\varphi_i dxdy, k=1,2, \qquad (3.13)$$

and

$$(B^x)^T=\int_\Omega(\varphi_i)_x\phi_r dxdy,\qquad (B^y)^T=\int_\Omega(\varphi_i)_y\phi_r dxdy, \qquad (3.14)$$

with $i,j=1,2,\cdots n$, and $r=1,2,\cdots m$.

The formulas for numerical integration [5] of a function f over a triangle of area S are all of the form

$$\int\int f dS = S\sum_{i=1}^{N_G} w_i f(\xi_i,\eta_i,\zeta_i), \qquad (3.15)$$

where ξ_i, η_i, ζ_i are the area co-ordinates of the i-th sampling point, w_i is the weight associated with the i-th sampling point, and N_G is the number of quadrature points. In this thesis, we use three points Gauss integration rule, which ensures the numerical integration to have a degree of precision 2.

Hence, for equations (3.12)-(3.14), the shape function and shape function deriva-

tives are evaluated at the quadrature points in $\hat{\Omega}$ and they are used in a quadrature sum for each element. For example, for the integration of function f and shape function φ_j over a triangle, we obtain

$$\begin{aligned}\int_{\Omega_e} f\varphi_j dxdy &= \int_{\hat{\Omega}} \hat{f}\hat{\varphi}_j |\mathbf{J}| d\xi d\eta \\ &= S\sum_{i=1}^{N_G} w_i \hat{f}(\xi_i,\eta_i)\hat{\varphi}_j(\xi_i,\eta_i)|\mathbf{J}(\xi_i,\eta_i)|. \end{aligned} \tag{3.16}$$

2. Local Solenoidal Basis

The structure of local solenoidal functions affects the convergence rate of the iterative method to solve the reduced system (1.11). Construction of local solenoidal functions is complicated, especially on an unstructured mesh. As shown in last chapter, experimental results of 2D and 3D MAC scheme on an uniform mesh illustrate the effectiveness of preconditioning techniques based on circulating solenoidal flow structure. The behaviour of the robust preconditioning technique motivates extension of the same technique to unstructured mesh. We try to construct similar circulating flows on an unstructured mesh similar to circulating flows in local regions in MAC scheme on an uniform mesh. Just like the procedure mentioned in $P1$ *iso-*$P1$ case [14], the solenoidal functions for $P2/P1$ are classified into three groups:

- Nodal solenoidal functions
- Edge solenoidal functions
- Element solenoidal functions.

Given a mesh with m pressure nodes, e edges, and t elements, there are $2m$ nodal solenoidal functions, e edge solenoidal functions and t element solenoidal functions. All these solenoidal functions are constructed after coordinate transformation of edge velocities during preprocessing. A typical element pair is shown in Figure 7, where V_e is the sharing edge velocity of the element pair, P_i is pressure node with correspond-

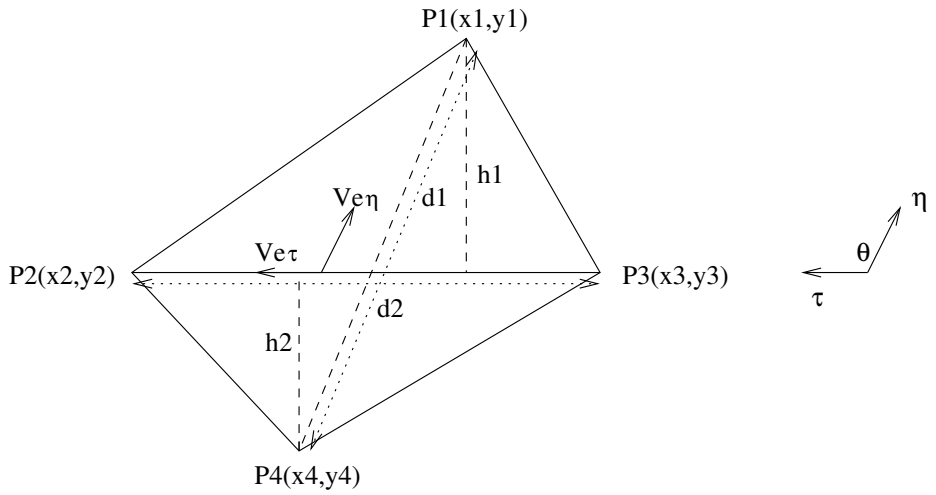

Fig. 7. Coordinate transformation for element pair

ing $(x_i,\ y_i)$ coordinates $(i = 1, 2, 3, 4)$, d_1 is the length of the line joining pressure nodes P_1 and P_4, d_2 is the length of the sharing edge joining pressure nodes P_2 and P_3, h_1 is the vertical distance from pressure node P_1 to the sharing edge, and h_2 is the vertical distance from pressure node P_4 to the sharing edge. The transformed space is τ-η space, where τ direction is parallel to the sharing edge of the element pair and η direction is parallel to the line joining the opposite pressure nodes. This coordinate transformation simplifies the structure of B matrix and makes it sparser. The coordinate transformation of edge velocity from (x, y) to (τ, η) is accomplished by a linear operator matrix G. Then for the kth element as shown in Figure 7 the submatrix G_k is given as

$$G_k = \begin{bmatrix} (x_2 - x_3)/d_2 & (y_2 - y_3)/d_2 \\ (x_1 - x_4)/d_1 & (y_1 - y_4)/d_1 \end{bmatrix}. \tag{3.17}$$

The size of the global matrix G is $2(m+e) \times 2(m+e)$ considering both x direction and y direction for the given mesh with m pressure nodes and e edges. Since G matrix

only transforms edge velocities without any changes for node velocities, the portion of G matrix corresponding to pressure node velocities should be identity matrix. Hence, G matrix has the following structure

$$G = \left[\begin{array}{c|c|c|c} I_m & & & \\ \hline & \ddots & & \ddots \\ & D_{xx} & & D_{xy} \\ & & \ddots & & \ddots \\ \hline & & I_m & \\ \hline & \ddots & & \ddots \\ & D_{yx} & & D_{yy} \\ & & \ddots & & \ddots \end{array}\right], \tag{3.18}$$

where I_m is identity matrix, D_{xx}, D_{xy}, D_{yx}, and D_{yy} are coming from global generation of submatrix G_k. Since the B matrix takes the order of rows as x component before y component, G matrix takes same order as B matrix. Matrix $\widetilde{B}^T = (GB)^T$ denotes the divergence operator matrix in the transformed space. $\widetilde{B}$ matrix for the kth element involving the edge velocity node and the surrounding pressure nodes is given in (3.19). The submatrix $\widetilde{B}_k$ with rows corresponding to τ, η components of interior edge velocity, and columns corresponding to pressure nodes, is given as

$$\left[\widetilde{B}\right]_k = \begin{array}{c} \\ \tau \\ \eta \end{array} \begin{array}{c} \begin{array}{cccc} P_1 & P_2 & P_3 & P_4 \end{array} \\ \begin{bmatrix} 0 & h_1+h_2 & -(h_1+h_2) & 0 \\ d_2\sin\theta & 0 & 0 & -d_2\sin\theta \end{bmatrix} \end{array}, \tag{3.19}$$

where θ is the angle between $\tau - \eta$ direction. From equation(3.19), one can observe that transformed edge velocities have nonzeros in the $\widetilde{B}$ matrix for only those pressure nodes along their directions. Thus this transformation can simplify the structure of B matrix and make it much sparser.

After (τ, η) coordinate transformation using G matrix, the linearized system (1.7) becomes

$$\begin{bmatrix} GAG^T & GB \\ (GB)^T & 0 \end{bmatrix} \begin{bmatrix} G^{-T}\mathbf{u} \\ p \end{bmatrix} = \begin{bmatrix} G^T f \\ 0 \end{bmatrix}. \tag{3.20}$$

The procedure to solve the above system (3.20) is the following:

• Find the null space of $GB = \widetilde{B}$, i.e., $\widetilde{P} = Null(\widetilde{B}^T)$, which is equivalent to find the solenoidal functions. It is detailed from next paragraph.

• Suppose $\widetilde{\mathbf{u}} = \widetilde{P}x$. Solve for x from the transformed reduced system (3.21),

$$\widetilde{P}^T[GAG^T]\widetilde{P}x = \widetilde{P}^T[Gf], \tag{3.21}$$

and recover velocity as $\mathbf{u} = G^T\widetilde{\mathbf{u}} = G^T(\widetilde{P})x$.

Nodal-based solenoidal functions are constructed cluster by cluster, where a cluster is formed by assembling all the elements sharing a given interior pressure node. Figure 8 shows an example of the cluster formed around pressure node P_0 where v_0, v_{e1}, v_{e2}, v_{e3}, v_{e4}, $\cdots$ v_{ek} are velocity nodes inside the cluster formed by P_0, P_1, P_2,

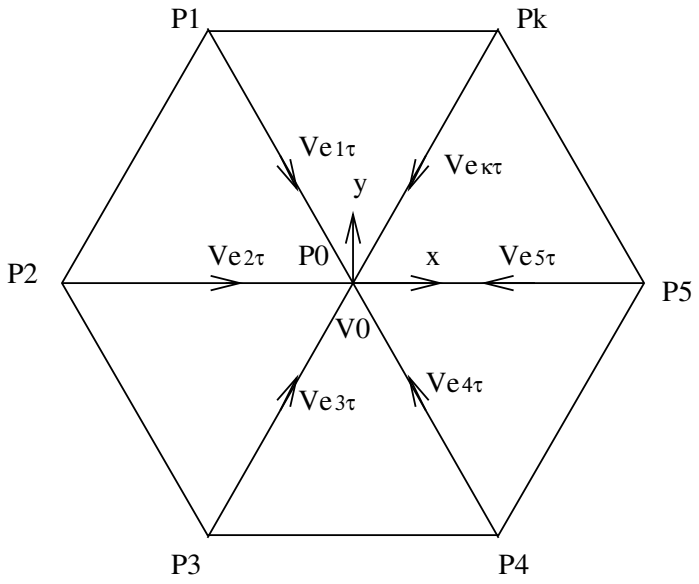

Fig. 8. Nodal-based solenoidal functions

P_3, P_4, P_5, $\cdots$ P_k pressure nodes. Nodal solenoidal flows consist of only τ component of velocity nodes v_{e1}, v_{e2}, v_{e3}, v_{e4}, $\cdots$ v_{ek}, and x, y components of v_0. Since the net flow into P_0 is zero, the nodal-based solenoidal flow can be generated. $\widetilde{B}$ matrix for the kth nodal cluster involving the edge velocity nodes and the surrounding pressure nodes is given in (3.22). The submatrix $\widetilde{B}_k$ with rows corresponding to x, y components of v_0 together with τ component of velocity along incoming edges, and columns corresponding to surrounding pressure nodes, is given as

$$\left[\widetilde{B}\right]_k = \begin{array}{c} \\ v_0(x) \\ v_0(y) \\ v_{e1\tau} \\ v_{e2\tau} \\ v_{e3\tau} \\ v_{e4\tau} \\ v_{e5\tau} \\ \vdots \\ v_{ek\tau} \end{array} \begin{array}{c} \begin{array}{cccccccc} P_0 & P_1 & P_2 & P_3 & P_4 & P_5 & \cdots & P_k \end{array} \\ \left[\begin{array}{cccccccc} \alpha & & & & W_1^T & & & \\ \beta & & & & W_2^T & & & \\ -t_1 & t_1 & & & & & & \\ -t_2 & & t_2 & & & & & \\ -t_3 & & & t_3 & & & & \\ -t_4 & & & & t_4 & & & \\ -t_5 & & & & & t_5 & & \\ \vdots & & & & & & \ddots & \\ -t_k & & & & & & & t_k \end{array}\right] \end{array} \begin{array}{l} \\ \leftarrow x - \text{function at } P_0 \\ \leftarrow y - \text{function at } P_0 \\ \\ \\ \\ \\ \\ \\ \\ \end{array} . \tag{3.22}$$

Since the influence of interior velocities outside the nodal cluster is zero, one can infer that the sum of all columns for any row is zero, which implies that the following equations are satisfied

$$\begin{aligned} \alpha + W_1^T \vec{e} &= 0, \\ \beta + W_2^T \vec{e} &= 0, \end{aligned} \tag{3.23}$$

where $\vec{e}$ refers to a vector of all ones. We can get the null space of $\left[\widetilde{B}\right]_k$ with two vectors

$$\left[\widetilde{P}\right]_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -W_1^T D^{-1} & -W_2^T D^{-1} \end{bmatrix}, \tag{3.24}$$

where $D = diag[t_1, t_2, \cdots, t_k]$. $\widetilde{P}$ has two columns involving x directional velocity of pressure node P_0 and y directional velocity of P_0.

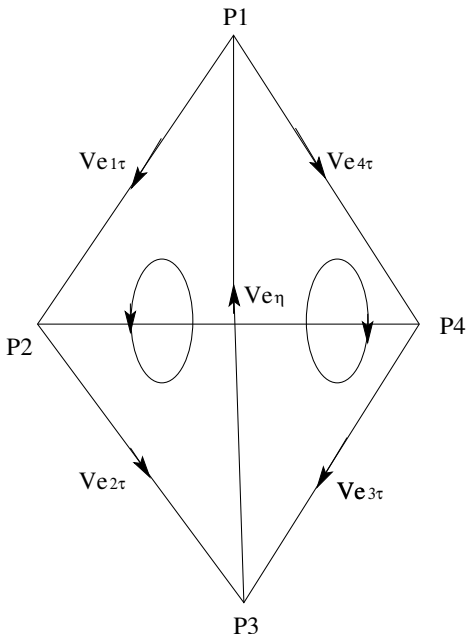

Fig. 9. Edge-based solenoidal functions

Edge-based solenoidal functions include five edge velocities in the edge solenoidal flow and four pressure nodes forming the element pair. Figure 9 shows an example of an element pair with the interior edge bounded by pressure nodes $[P_1\ P_2\ P_3\ P_4]$. Two solenoidal flows with opposite direction are generated. Both of these flows start from η component of interior edge velocity. $\widetilde{B}$ matrix for the kth interior edge involving the edge velocity nodes and the surrounding pressure nodes is given in (3.25). The submatrix $\widetilde{B}_k$ with rows corresponding to τ component of interior velocity nodes together with one η component of interior edge velocity, and columns corresponding to surrounding pressure nodes, is given as

$$\left[\widetilde{B}\right]_k = \begin{array}{c} \\ v_{e1\tau} \\ v_{e2\tau} \\ v_{e3\tau} \\ v_{e4\tau} \\ v_{e\eta} \end{array} \begin{array}{c} \begin{array}{cccc} P_1 & P_2 & P_3 & P_4 \end{array} \\ \begin{bmatrix} t_1 & -t_1 & 0 & 0 \\ 0 & t_2 & -t_2 & 0 \\ 0 & 0 & -t_3 & t_3 \\ t_4 & 0 & 0 & -t_4 \\ -t_5 & 0 & t_5 & 0 \end{bmatrix} \end{array}. \tag{3.25}$$

From Equation(3.25) one can observe that those transformed edge velocities have nonzeros in the $\left[\widetilde{B}\right]_k$ only for the pressure nodes along their direction. We can get the null space of $\left[\widetilde{B}\right]_k$

$$\left[\widetilde{P}\right]_k = \frac{t_5}{2}\begin{bmatrix} 1/t_1 & 1/t_2 & 1/t_3 & 1/t_4 & 2/t_5 \end{bmatrix}^T. \tag{3.26}$$

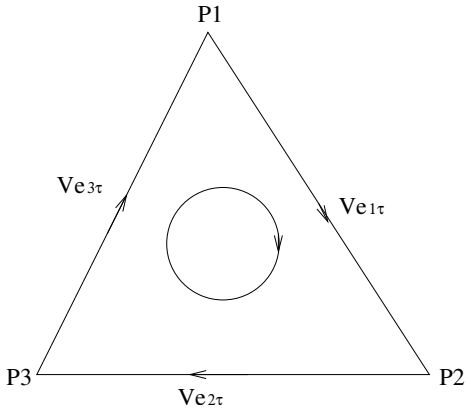

Fig. 10. Element-based solenoidal functions

Element solenoidal functions are generated using τ component of edge velocities. Figure 10 shows an example of an element with one solenoidal flow. The construction of element solenoidal functions is similar to that of cell flows generated in MAC scheme. $\widetilde{B}$ matrix for the kth element involving three edge velocity nodes and the

surrounding pressure nodes is given in (3.27). The submatrix $\widetilde{B}_k$ with rows corresponding to τ component of interior velocity nodes, and columns corresponding to surrounding pressure nodes, is given as

$$\left[\widetilde{B}\right]_k = \begin{array}{c} \\ v_{e1\tau} \\ v_{e2\tau} \\ v_{e3\tau} \end{array} \begin{array}{c} \begin{array}{ccc} P_1 & P_2 & P_3 \end{array} \\ \begin{bmatrix} t_1 & -t_1 & 0 \\ 0 & t_2 & -t_2 \\ -t_3 & 0 & t_3 \end{bmatrix} \end{array}. \tag{3.27}$$

From Equation(3.27) we can observe that edge tangential velocity has nonzeros in $\left[\widetilde{B}\right]_k$ matrix for only those pressure nodes that share the edge.

We can compare the elemental solenoidal flow with that in MAC scheme to find the similarity. The null space of $\left[\widetilde{B}\right]_k$ becomes

$$\left[\widetilde{P}\right]_k = \begin{bmatrix} 1/t_1 & 1/t_2 & 1/t_3 \end{bmatrix}^T. \tag{3.28}$$

Until now we generated localized null space vector, the generated $[\widetilde{P}]_k$ represents a solenoidal function local to flow in either a nodal cluster, an interior edge or an element k. Using the above approach, local solenoidal flows can be constructed directly from the matrix $\widetilde{B}^T$. When we expand vector $[\widetilde{P}]_k$ to a length $2m + 2e$ vector $\widetilde{P}_i$ by adding zeros, it satisfies the divergence free constraint $\widetilde{B}^T\widetilde{P}_i = 0$. The set of vector $\widetilde{P}_i$ forms the discrete solenoidal basis $\widetilde{P}$. The global null space P can be recovered as $P = G^{-1}\widetilde{P}$.

B. Preconditioning the Reduced System

In order to better understand the structure of $\widetilde{P}$, we rearrange the solenoidal functions according to the order of nodal cluster, edge solenoidal flows and element solenoidal flows. Then $\widetilde{P}$, the null space of the global $\widetilde{B}^T$, has the structure shown in equation (3.29). From the structure of $\widetilde{P}$, we can see that columns of nodal solenoidal functions $\widetilde{P_m}$, edge solenoidal functions $\widetilde{P_e}$ and element solenoidal functions $\widetilde{P_t}$ are mutually independent.

$$\widetilde{P} = \left[\widetilde{P_m}, \widetilde{P_e}, \widetilde{P_t}\right] \tag{3.29}$$

Table II. Structure of null space $\widetilde{P}$

	Nodal Solenoidal Functions $\widetilde{P_m}$	Edge Solenoidal Functions $\widetilde{P_e}$	Element Soelnoidal Functions $\widetilde{P_t}$
$2\tilde{m}$ node velocity	$I_{2\tilde{m}}$	0	0
η- edge velocity	0	$I_{\tilde{e}}$	0
τ- edge velocity	$\widetilde{P}_{13}$	$\widetilde{P}_{23}$	$\widetilde{P}_{33}$

$\widetilde{P}$ has full column rank. We can confirm it by checking the dimension of null space. The size of $\widetilde{P}$ is equal to $2\tilde{m} + \tilde{e} + \tilde{t}$, where $\tilde{m}$ is the number of pressure nodes, $\tilde{e}$ is the number of edges and $\tilde{t}$ is the number of elements. The size of $\tilde{B}$ is

$(2\tilde{m} + 2\tilde{e}) \times \tilde{m}$, so that the dimensions of null space becomes $2\tilde{m} + 2\tilde{e} - \tilde{m} + 1 = 2\tilde{m} + \tilde{e} + (\tilde{m} + \tilde{t} - 1) - \tilde{m} + 1 = 2\tilde{m} + \tilde{e} + \tilde{t}$, which is equal to the size of $\tilde{P}$.

From Table II, we can observe that $\widetilde{P}_m$ and $\widetilde{P}_e$ are well conditioned since they have the identity matrix on the diagonal. Preconditioning would be more effective when it transforms $\widetilde{P}_t$ to a well-conditioned matrix. Preconditioner for the reduced system (1.11), which is used to precondition matrix $P^T AP$, is given by

$$G_p = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & P_t^T P_t \end{bmatrix}. \tag{3.30}$$

Compared to $P1$ *iso* - $P1$ case given in [14], the reduced system (1.11) is preconditioned by the following

$$G_p = \left[\frac{1}{\Delta t} M + \frac{1}{Re} P^T P\right] \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & P_t^T P_t \end{bmatrix}, \tag{3.31}$$

and solved under various physical conditions by changing the ratio $\kappa = h^2 Re/\Delta t$. Experimental results shows that when $\kappa > 10$ the preconditioning is near optimal which assures stable convergence, regardless of parameters such as the mesh size, Reynolds number, and the time step.

CHAPTER IV

NUMERICAL EXPERIMENTS

Generating a finite element triangulation is done by constructing initial grid and refinement of the mesh using *'pdetool'* of commercial software Matlab. The refinement technique is to take the three midpoints of a triangle, thus creating four smaller triangles from a larger triangle. We make use of this technique to get the information for $P2/P1$ triangular element. *'pdetool'* has the export function to get the data structure of p, e, t, in which p has the coordinates of the nodal points, e has the information for the edges on the boundary, and t has the information of three corner nodes per element. In order to evaluate quadratic velocity polynomial, nodes on the midpoints need to be determined. A special data structure *'nodes'* is used to store all the six nodes per element during preprocessing. The structure of global null space matrix $\widetilde{P}$ in Table II indicates that three groups of cluster information is needed in order to construct local solenoidal basis. During the preprocessing, we use a set of vectors to get the elemental cluster information, which are generated for each pressure node with anti-clock wise direction. We then get the groups of cluster information for node cluster, edge cluster and element cluster using this elemental cluster information.

We use $P2/P1$ mixed finite element approximation and present results for three standard test flow problems. We solve the linear system that arises at each discrete time interval using CG method [8] with the preconditioner P as defined in Table II. To show the robustness of the solver, both the iteration counts and execution time required for the tolerance to be satisfied on a given mesh are reported. In order to demonstrate that the solution obtained from the proposed solenoidal basis method is exact for quadratic polynomial velocity, we compare with one known exact solution problem. In addition to that, we test the driven cavity flow for both Stokes

problem (4.1) and generalized Stokes problem (1.6) to illustrate the performance and the effectiveness of the preconditioner. In these experiments, CG iterations are terminated when the relative residual is reduced by 10^{-3}.

A. Stokes Problem

The steady Stokes problem with Dirichlet conditions on the boundary for incompressible fluids satisfies

$$\begin{aligned} -\frac{1}{Re}\nabla^2\mathbf{u} + \nabla p &= \mathbf{f} \qquad in \quad \Omega, \\ \nabla \cdot \mathbf{u} &= 0 \qquad in \quad \Omega, \\ \mathbf{u} &= \mathbf{g} \qquad on \quad \partial\Omega. \end{aligned} \tag{4.1}$$

The first problem to be considered is Stokes problem applied to flow on the unit domain $\Omega = (0,1) \times (0,1)$. Applying the constraint $\int_\Omega p dx = 0$,

$$\mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}, \text{and the boundary conditions } \mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x^2 \\ -2xy \end{bmatrix} \tag{4.2}$$

on $\partial\Omega$, the exact solution of system (4.1) is given as

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x^2 \\ -2xy \end{bmatrix}, \qquad p = x. \tag{4.3}$$

We test the problem with the grid in Figure 11. The approximate solution of u, v are shown by the meshes in Figure 12 and Figure 13. Comparing the approximate solution with exact solution, it is seen that the error in u, v velocity for Stokes problem reduces to 10^{-6} when the relative residual is reduced by 10^{-6}, which verifies the correctness of the solenoidal basis method.

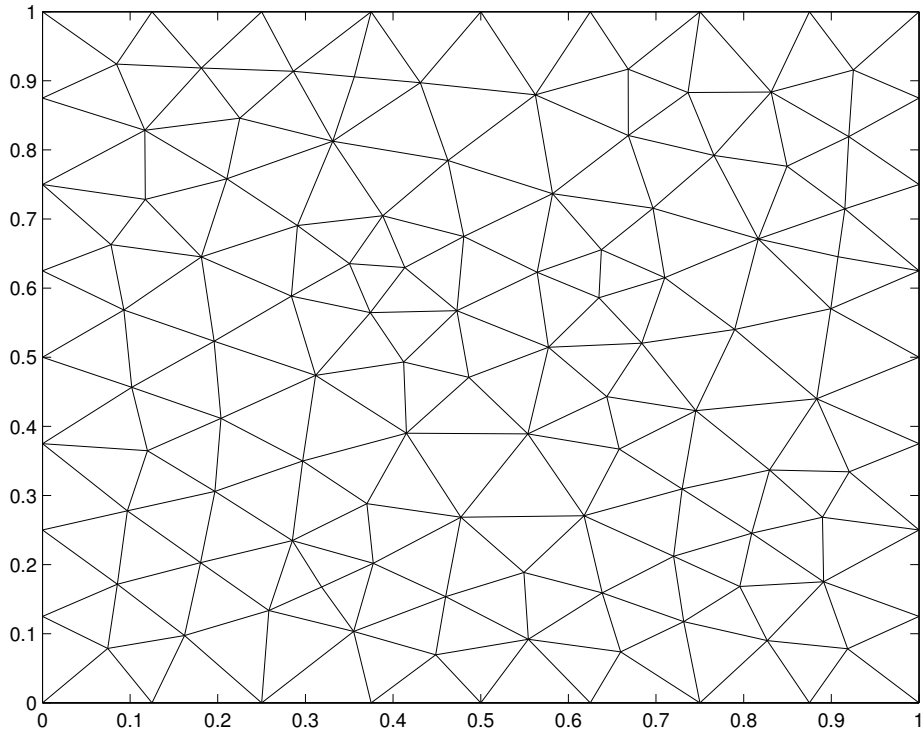

Fig. 11. Finite element mesh with mesh width parameter as 1/8

B. Stokes Problem: Driven Cavity Flow

We consider the Stokes problem, associated with a standard driven cavity flow problem defined on a unit domain $\Omega = (0,1) \times (0,1)$, see Figure 14. The associated boundary condition is given by

$$\mathbf{u}(\partial\Omega, t) = \begin{cases} (1,0) & y = 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

The matrix A consists of only Laplace operator part $A = \frac{1}{Re}L$, which is symmetric positive definite matrix. The Stokes problem is solved with the grid given in Figure 11, and the solution of velocity plot is given in Figure 15. Figure 16 shows the contours for the magnitude of the velocity.

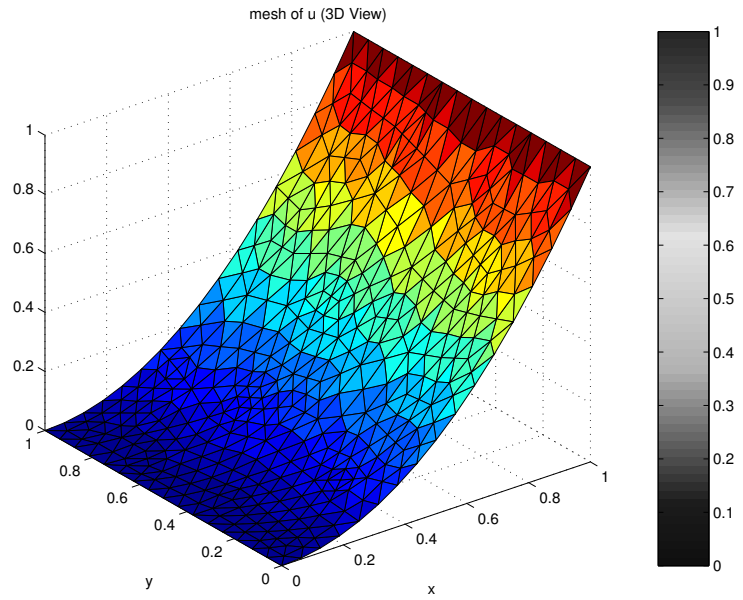

Fig. 12. Solution of velocity u

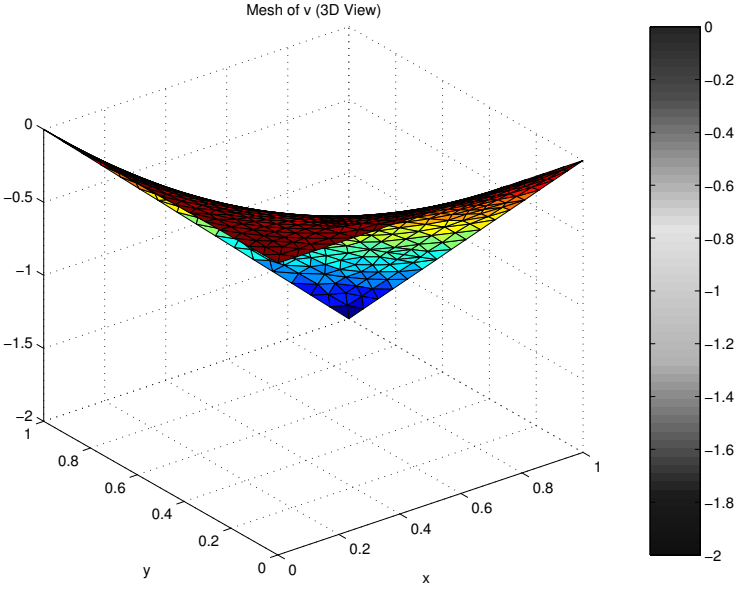

Fig. 13. Solution of velocity v

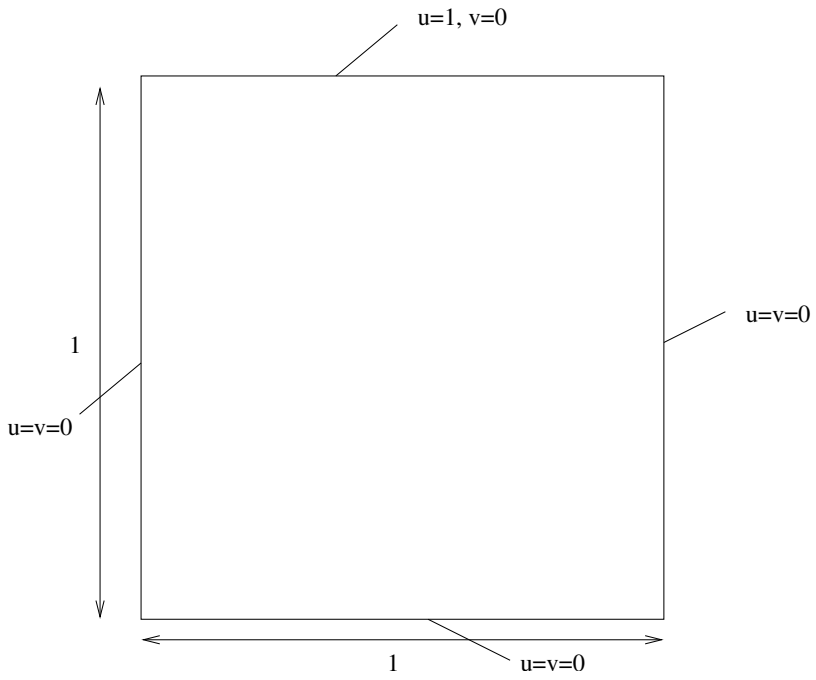

Fig. 14. The geometry and the boundary conditions of the driven cavity flow test problem

C. Generalized Stokes Problem: Driven Cavity Flow

Table III. Problem sizes for the generalized Stokes problem

Mesh	Pressure	Velocity	Solenoidal functions
h = 1/8	123	914	603
h = 1/16	469	3618	3362
h = 1/32	1807	14194	12003

The generalized Stokes problem is considered with same domain, boundary and initial conditions given above. The unsteady Stokes problem is solved with the grid

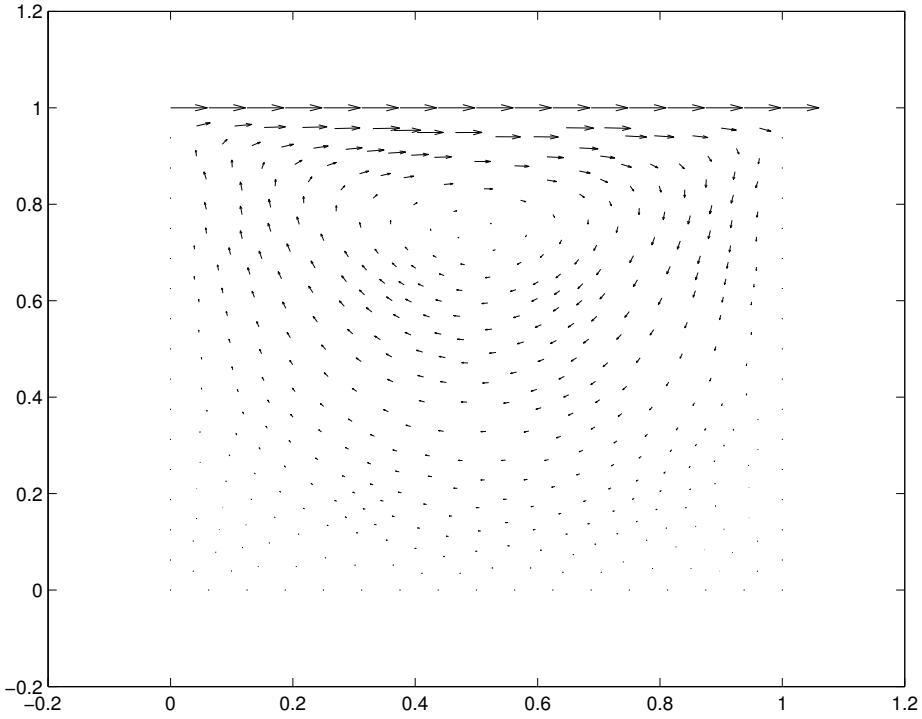

Fig. 15. Solution for the driven cavity problem

in Figure 11. The matrix A takes the following form

$$A = \frac{1}{\Delta t} M + \frac{1}{Re} L. \tag{4.5}$$

The set of experiments shows the behaviour of the solenoidal basis method for different Reynolds number and different time interval Δt. Reynolds number varies from 100 to 1000 and time interval Δt varies from 0.001 to 10. Table III shows the number for unknowns of pressure nodes and velocity nodes for the meshes used in the experiments. The number of solenoidal functions represents the size of the reduced system that is solved by the preconditioned method.

Convergence of the iterative method is sensitive to the variation of coefficient in matrix A, which depends on Δt and Re. The effect of time interval Δt is given in Table

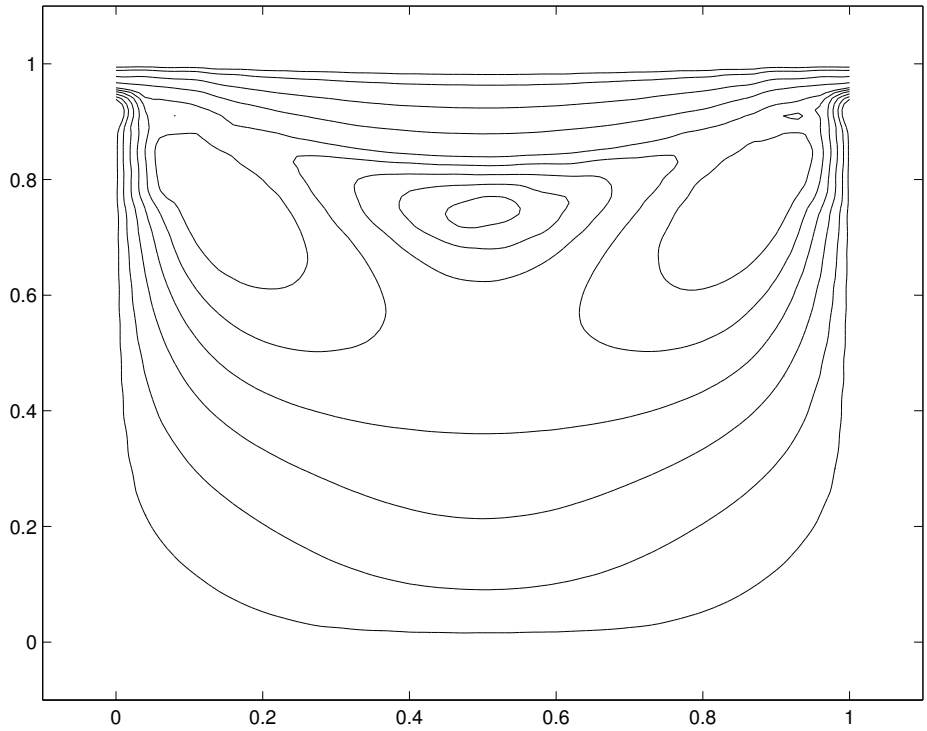

Fig. 16. The contours for the magnitude of the velocity for the driven cavity problem

IV. It can be seen that as Δt gets smaller, the iteration counts tend to be relatively stable regardless of mesh refinement. Moreover the maximum count becomes larger as Δt increases. In order to verify the performance of preconditioned solenoidal method, both preconditioned system and unpreconditioned system are solved. The maximum iteration counts are given in Table V, the execution time is given in Table VI. The effectiveness of the preconditioned solenoidal basis method is illustrated via several instances of Re for $P2/P1$ case with $\Delta t = 0.01$. Compared the preconditioned with unpreconditioned systems, it can be easily observed that the iteration counts of preconditioned system are far less than that of the unpreconditioned ones. Compared to unpreconditioned system, the preconditioned system is solved in much less time. It can be noticed that when $Re/\Delta t > 10^4$, the iteration count converges to a stable

Table IV. Effect of time interval for the preconditioned solenoidal basis method in P2/P1 case with $Re = 100$

Δt	h = 1/8	h = 1/16	h = 1/32
0.001	75	47	48
0.01	41	50	54
0.1	47	105	190
1	107	219	333
10	160	323	420

value regardless of mesh refinement, which indicates that the preconditioner is optimal for this case. The same result does not apply for the case when $Re/\Delta t < 10^4$. The preconditioner is not optimal for that system even though it is optimal for any system on uniform mesh using MAC scheme.

Table V. Iteration counts for the preconditioned solenoidal basis method for P2/P1 case with $\Delta t = 0.01$

Preconditioned			
Re	h = 1/8	h = 1/16	h = 1/32
100	41	50	54
1000	75	47	48
Unpreconditioned			
Re	h = 1/8	h = 1/16	h = 1/32
100	102	289	733
1000	114	194	449

Table VI. Execution time for the preconditioned solenoidal basis method for P2/P1 case with $\Delta t = 0.01$

Preconditioned			
Re	h = 1/8	h = 1/16	h = 1/32
100	0.15	1.00	6.40
1000	0.26	0.93	5.82
Unpreconditioned			
Re	h = 1/8	h = 1/16	h = 1/32
100	0.34	4.59	54.75
1000	0.39	3.09	32.29

CHAPTER V

CONCLUSIONS

In this thesis we developed an extension of the preconditioned solenoidal basis technique on an unstructured mesh for solving the linear system arising from the finite element discretization of Navier-Stokes equations. A localized algebraic scheme was outlined to compute discrete local solenoidal flows using $P2/P1$ triangular element. A preconditioner was presented after analyzing the structure of the reduced system. Benchmark simulations were conducted to illustrate the effectiveness of the proposed technique on an unstructured mesh, which shows the following:

- The velocity system can be solved by solenoidal basis method with high precision.
- Preconditioner for unstructured 2D mesh is more effective because the preconditioner takes care of ill-conditioned part of global solenoidal basis. Preconditioned system outperforms unpreconditioned system in terms of both iteration counts and execution time.

REFERENCES

[1] O. Axelsson and P. Vassilevski, "Algebraic Multilevel Preconditioning Methods 2", *SIAM J.Numer.Anal.*, vol.27, pp.1569-1590, 1990.

[2] O. Axelsson and P. Vassilevski, "Algebraic Multilevel Preconditioning Methods 1", *Numer.Math.*, vol.56, pp.157-177, 1999.

[3] R. E. Bank, B. D. Welfert, and H. Yserentant, "A Class of Iterative Methods for Solving Saddle Point Problems", *Numer.Math.*, vol.56, pp.645-666, 1990.

[4] T. J. Chung, *Finite Element Analysis in Fluid Dynamics*, McGraw-Hill, Inc., 1978.

[5] G. R. Cowper, "Gaussian Quadrature Formulas for Triangles", *International Journal for Numerical Methods in Engineering*, vol.7, pp.405-408, 1973.

[6] N. Dyn and W. E. Ferguson, "The Numerical Solution of Equality-Constrained Quadratic Programming Problems", *Math.Comp.*, vol.41, pp.165-170, 1983.

[7] M. D. Gunzburger, *Finite Element Methods for Viscous Incompressible Flows*, Academic Press, New York, 1989.

[8] M. R. Hestenes and E. Stiefel, "Methods of Conjugate Gradients for Solving Linear Systems", *J. Res. Nat. Bur. Stand*, vol.49, pp.409-436, 1952.

[9] D. Peaceman and A. Rachford, "The Numerical Solution of Parabolic and Elliptic Differential Equations", *SIAM J.*, vol.3, pp.28-41, 1955.

[10] T. Rusten and R. Winther, "A Preconditioned Iterative Method for Saddle-Point Problems", *SIAM J.Matrix Anal.Appl.*, vol.13, pp.887-904, 1992.

[11] Y. Saad and M. H. Schultz, "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems ", *SIAM J. Sci. Stat. Comput*, vol.7, pp.856-869, 1986.

[12] Y. D. Saad, "ILUT: A Dual Threshold Incomplete ILU Factorization", Technical Report 92-83, University of Minnesota Super Computer Institute, Minneapolis, MN, 1992.

[13] S. R. Sambavaram and V. Sarin, "A Parallel Solenoidal Basis Method for Incompressible Fluid Flow Problems", *Parallel Computational Fluid Dynamics*, pp.309-314, 2002.

[14] S. R. Sambavaram, "High Performance Parallel Algorithms for Incompressible Flows", master's thesis, Texas A&M University, December 2002.

[15] A. H. Sameh and V. Sarin, "Hybrid Parallel Linear System Solvers", *International Journal of Computational Fluid Dynamics*, vol.12, pp.213-223, 1999.

[16] V. Sarin and A. H. Sameh, "An Efficient Iterative Method for the Generalized Stokes Problem", *SIAM Journal of Scientific Computing*, vol.19, no.1, pp.206-226, 1998.

[17] V. Sarin, "Parallel Linear Solvers for Incompressible Fluid Problems", *Proceedings of the SIAM Parallel Processing Conference*, Portsmouth, VA, March 2001.

VITA

Name	Xue Wang
Permanent Address	C/o Dr.Vivek Sarin
	Department of Computer Science
	TAMUS 3112
	Texas A&M University
	College Station, TX 77843-3112
Education	B.E., Marine and Ocean Engineering
	South China University of Technology, China, 1996.
	M.E., Naval Architecture and Ocean Engineering
	Osaka University, Japan, 2000.
	M.S., Computer Science
	Texas A&M University, U.S.A., 2004.