Numerical Heat Transfer, Part B, 67: 1–25, 2015
Copyright © Taylor & Francis Group, LLC
ISSN: 1040-7790 print/1521-0626 online
DOI: 10.1080/10407790.2014.949522

ON A WAVENUMBER OPTIMIZED STREAMLINE UPWINDING METHOD FOR SOLVING STEADY INCOMPRESSIBLE NAVIER-STOKES EQUATIONS

Neo S. C. Kao[1], Tony W. H. Sheu[1,2,3], and S. F. Tsai[4]
[1]*Department of Engineering Science and Ocean Engineering, National Taiwan University, Taipei, Taiwan, Republic of China*
[2]*Center for Advanced Study in Theoretical Science (CASTS), National Taiwan University, Taipei, Taiwan, Republic of China*
[3]*Institute of Applied Mathematical Science, National Taiwan University, Taipei, Taiwan, Republic of China*
[4]*Department of Marine Engineering, National Taiwan Ocean University, Taipei, Taiwan, Republic of China*

A stabilized upwind finite-element model is developed to solve the three-dimensional incompressible steady Navier-Stokes equations. The test function is constructed to have a larger weight on the upstream side. This has been achieved by adding a stabilized term to the shape function so as to optimize the numerical wavenumber for convection terms. To avoid Lanczos or pivoting breakdown while solving the resulting unsymmetric and indefinite mixed finite-element matrix equations iteratively, the finite-element equation has been modified by pre-multiplying it with its transpose. The resulting normalized matrix equation becomes symmetric and positive-definite. We can therefore apply a computationally efficient conjugate gradient Krylov iterative solver to get an unconditionally convergent solution. However, the condition number of the new system becomes the square of the original unsymmetric indefinite system. To fully exploit excellent convergence nature of the conjugate gradient iterative solver, an element-by-element strategy is adopted to avoid assembling of all the stiffness matrices obtained at element level. We alleviate the drawback of slower convergence of the conjugate gradient method due to the increased condition number by preconditioning the positive-definite matrix. The resulting preconditioned matrix equation is solved in a matrix-free manner using the preconditioned conjugate gradient iterative solver. The developed finite-element code is first verified by solving a problem amenable to analytical solution. The benchmark lid-driven cavity problem is also solved in a cube for assessing the three chosen iterative solvers.

Received 16 April 2014; accepted 27 May 2014.

Address correspondence to Tony W. H. Sheu, Department of Engineering Science and Ocean Engineering, National Taiwan University, No. 1, Sec. 4, Roosevelt Road, Taipei, Taiwan 10617, Republic of China. E-mail: twhsheu@ntu.edu.tw

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/unhb.

NOMENCLATURE

$\underline{\underline{A}}$	global stiffness matrix	$N_i(i=1\text{–}27)$	tri-quadratic interpolation functions
$\underline{b}$	global right-hand side		
$\underline{\underline{B}}_e$	Eth element Boolean matrix	Re	Reynolds number ($\equiv \rho\, u_\infty L/\mu$)
B_i	biased stabilized term	u_∞	reference velocity
$\underline{\underline{B}}_J$	Jacobi preconditioner	δ^α	upwind parameter at center node
$\underline{\underline{B}}_P$	polynomial preconditioner	δ^β	upwind parameter at corner node
$\underline{\underline{B}}_{SP}$	scaling polynomial preconditioner	κ	wavenumber
h	$H/2$	μ	kinetic viscosity
H	element size	ρ	fluid density
$\mathscr{K}$	Krylov subspace	ξ_i, η_i, ζ_i	normalized coordinate
L	characteristic length	τ	stabilized parameter
$M_l(l=1\text{–}8)$	tri-linear interpolation functions	ω	scaling parameter
$\underline{n}$	unit outward normal vector		

1. INTRODUCTION

Simulation of incompressible viscous fluid flow by the finite-element method encounters several kinds of numerical oscillation. The first oscillation type results from the approximation of convective terms in flow equations. The second kind of numerical instability is attributed to the absence of a pressure term in the continuity equation. The incompressibility (or divergence-free) constraint condition on the equations of motion, which can be unconditionally satisfied by using a mixed finite-element model, is subject, however, to satisfaction of the Ladyzhenskaya-Babuška-Brezzi (LBB) stability condition. The necessity of imposing compatability condition to avoid numerical instability inhibits an arbitrary choice of finite-element basis functions. In other words, an element endowed with the LBB condition is required when adopting the mixed finite-element formulation.

The first origin leading to numerical oscillation has association with the local flow direction along which the Reynolds (or Peclet) number is very high. Finite-element solution can very often deteriorate due to the node-to-node oscillation when employing the Galerkin method. An artificial damping term needs to be added to the finite-element equation to eliminate this type of pathologic oscillation. The predicted solution very often becomes, however, overdiffusive along a direction orthogonal to the local streamline. The numerical accuracy deteriorates as well if the flow angle is not well aligned with the grid line. This motivates development of a new convectively more stable and accurate three-dimensional Petrov-Galerkin model in tri-quadratic elements. The concept of the streamline upwind Petrov-Galerkin (SUPG) [1] model will be adopted as our guideline to get a computationally more stable solution in tri-quadratic elements for the steady incompressible Navier-Stokes equations.

To improve the dispersive accuracy for a flow problem investigated at high Reynolds number, a new stabilized weighting function for the convection term is developed to yield an optimized numerical wavenumber. The upwinding finite-element equation developed for solving the steady Navier-Stokes equations at high Reynolds numbers is always unsymmetric. If a fluid flow under investigation is incompressible, its finite-element matrix equation becomes less diagonally dominant because of the presence of many diagonal zeros in the stiffness matrix. These diagonal zeros owing to the continuity equation can further increase the matrix condition number. A complex distribution of eigenvalues for the matrix equation free of symmetry can furthermore cause the equation to become indefinite. While use of direct solution solvers together with an efficient data storage strategy is suitable for performing the finite-element calculation of the two-dimensional problem, the storage demand becomes prohibitive in the course of fill-in for the three-dimensional analysis. Application of direct solvers may no longer be practical, calling for an iterative solver. In addition to the problems regarding the LBB constraint condition and the large-sized finite-element equations derived from the three-dimensional mixed formulation, matrix asymmetry and indefiniteness in the finite-element equation pose a grand computational challenge. Employment of a Gaussian-elimination-based direct solution solver has long been considered as a trivial means of circumventing these difficulties. In three-dimensional flow simulations, demand on the continued fill-in in the calculation of finite-element solution from a direct solver is infeasible even if one uses state-of- the-art computer technology. We have thus no choice but to turn to the iterative solution solvers.

Iterative methods developed for solving a large system of linear equations $\underline{\underline{A}}\underline{x} = \underline{b}$ avoid performing matrix–matrix operations that normally exist in direct solution solvers. One can multiply the vector $\underline{b}$ by the matrix $\underline{\underline{A}}$ and then deal with the resulting vector to get $\underline{\underline{A}}^2$ and so on. Such an iterative algorithm is referred to as the Krylov subspace method. This iterative method is among the most successful methods and is therefore adopted in this study. When applying the Krylov subspace iterative method, either an Arnolid algorithm or a Lanczos algorithm can be chosen to span the Krylov subspace by a series of orthogonal vectors. Depending on the way the basis is constructed, iterative solvers can be seperated into Arnoldi and Lanczos types. GMRES (Generalized Minimum RESidual) is the best-known Arnoldi-type Krylov subspace method. In the Lanczos class of Krylov subspace iterative methods, the BICGSTAB (BIConjugate Gradient STABilized), QMR (Quasi Minimal Residual), TFQMR (Transpose-Free QMR) [2], and MINRES (MINimal RESidual) methods are often referred to [3].

No Krylov subspace iterative solver can generate unconditional convergence of the unsymmetric finite-element matrix equations obtained from convection-dominated flow equations. In particular, this is true if the system involves a complex eigenvalue distribution and a large condition number. For these reasons, in this study the original unsymmetric indefinite finite-element matrix equations will be transformed to a symmetric positive-definite counterpart. The matrix equations will be normalized first, and one can then apply the conjugate gradient method to get the convergent solution from the symmetric and positive-definite normal matrix equation. When applying an iterative solver, a memory-efficient element-by-element concept [4] needs to be adopted to avoid storing the global matrix array. To fully exploit the element-by-element capability, the symmetric positive-definite matrix equations need to be preconditioned so that the condition number of the finite-element coefficient matrix can be reduced.

This article is organized as follows. In Section 2, the finite-element model is developed in tri-quadratic elements for suppressing the node-to-node checkerboarding of velocity solutions as well as the pressure oscillation. For enhancing convective stability for the case of high Reynolds number, the stabilized part of the weighting function will be applied mainly along the streamline direction. Within the resulting streamline upwind Petrov-Galerkin finite-element framework, the required amount of stabilization will be rigorously derived so as to get the best numerical wavenumber for the advection term which can, in turn, improve dispersive accuracy. The resulting three-dimensional unsymmetric and indefinite mixed finite-element matrix equations will then be solved iteratively in an element-by-element fashion. In Section 3, some iterative methods that have been widely referred to in the literature are first reviewed. We then present the CGNR iterative method in Section 4 and apply it to get the unconditionally convergent solution element by element from the matrix equations derived in Section 2. Since the matrix condition number will be increased substantially through the applied normalized procedure, preconditioning of the positive-definite matrix by the preconditioners described in Section 5 is needed. For the sake of comparison, two preconditioned iterative solvers, BICGSTAB and GMRES, are also assessed in Section 6. In Section 7, the proposed element-by-element finite-element model is first verified. The performance of the polynomial preconditioned iterative solver is then assessed through the calculation of three-dimensional lid-driven cavity flow at high Reynolds numbers. In Section 8, concluding remarks are drawn based on the simulated results.

2. FINITE-ELEMENT EQUATIONS WITH OPTIMIZED NUMERICAL WAVENUMBER

When investigating incompressible viscous flow problems, one can adopt any of three formulations: the primitive-variable, vorticity-vector potential, or vorticity–velocity formulation. Among them, the primitive-variable formulation is preferred because of the availability of rigorous closure boundary conditions. The steady incompressible Navier-Stokes equations in primitive-variable form, given as

$$\nabla \cdot \underline{u} = 0 \tag{1}$$

$$\underline{u} \cdot \nabla \underline{u} + \nabla p - \frac{1}{\mathrm{Re}} \nabla^2 \underline{u} = \underline{f} \tag{2}$$

will be solved in a three-dimensional domain Ω by the finite-element method. In this formulation, $\underline{u} = \{u,\ v,\ w\}$ and p are denoted as the velocity vector and pressure. The above elliptic differential equations will be solved subject to a Dirichlet-type condition $\underline{u} = \underline{g}$ along the boundary $\partial\Omega = \Gamma$. Note that the boundary vector $\underline{g}$ is constrained by $\int_\Gamma \underline{n} \cdot \underline{g} d\Gamma = 0$. The Reynolds number $\mathrm{Re} = \rho u_\infty L/\mu$ is the result of normalization of primitive variables. We denote S_u and S_p as the set of trial function spaces for the respective velocity and pressure:

$$S_u = \{u | u \in H^1(\Omega), u = g \text{ on } \Gamma\}$$

$$S_p = \{p | p \in L^2(\Omega)\}$$

where

$$L^2(\Omega) = \left\{ u | u \in \Omega, \int_\Omega u^2 \, d\Omega < \infty \right\}$$

$$H^1(\Omega) = \left\{ u | u \in \Omega, u, \frac{\partial u}{\partial x_k} \in L^2(\Omega) \right\}$$

In conjunction with trial function spaces, the weighting function spaces are also defined as

$$V_u = \{w | w \in H^1(\Omega), w = 0 \text{ on } \Gamma\} \tag{3}$$

$$V_p = S_p$$

Given the above functional spaces, the mixed variational problem characterizing Eqs. (1)–(2) and their associated boundary conditions for the chosen velocity–pressure pair of primitive variables $(\underline{u}, p) \in (S_u \times S_u) \times S_p$ will be dealt with. The following equations are solved to get the corresponding weak solutions of $\underline{u}$ and p:

$$\int_\Omega (\underline{u} \cdot \nabla)\underline{u} \cdot \underline{w}\, d\Omega + \frac{1}{\mathrm{Re}} \int_\Omega \nabla \underline{u} : \nabla \underline{w} d\,\Omega - \int_\Omega p \nabla \cdot \underline{w}\, d\Omega = \int_\Omega \underline{f} \cdot \underline{w}\, d\Omega \tag{4}$$

$$\int_\Omega (\nabla \cdot \underline{u}) q\, d\Omega = 0 \tag{5}$$

The above variational statement holds for all admissive functions $\underline{w} \in (V_u \times V_u)$ and $q \in S_p$.

In this article, univariant rather than multivariant finite-elements are adopted because of the programming simplicity. The primitive velocities are approximated using the following tri-quadratic basis functions N_i $(i = 1 - 27)$:

$$N_i = \left(\frac{3}{2}\bar{\xi}^2 + \frac{1}{2}\bar{\xi} + 1 + \xi^2 - \xi_i^2\right)\left(\frac{3}{2}\bar{\eta}^2 + \frac{1}{2}\bar{\eta} + 1 + \eta^2 - \eta_i^2\right)$$
$$\left(\frac{3}{2}\bar{\zeta}^2 + \frac{1}{2}\bar{\zeta} + 1 + \zeta^2 - \zeta_i^2\right) \tag{6}$$

In the above, ξ_i, η_i, and ζ_i denote the normalized coordinates for the i-th node and $\bar{\xi} = \xi\xi_i, \bar{\eta} = \eta\eta_i, \bar{\zeta} = \zeta\zeta_i$. To satisfy the compatability (or inf-sup, or LBB) condition, the pressure unknown is approximated by the eight nodal pressure values stored at element corners using the following basis functions M_l $(l = 1–8)$:

$$M_l = \frac{1}{8}(1 + \bar{\xi})(1 + \bar{\eta})(1 + \bar{\zeta}) \tag{7}$$

The quality of finite-element simulation of incompressible fluid flow in a convection-dominated flow depends on, among other factors, numerical accuracy and convective stability. One can easily get second-order spatial accuracy by applying the center-based Galerkin approach. The Galerkin model for the differential Equations (4)–(5) will in general lead to notorious velocity oscillations. To circumvent this numerical instability, a large number of upwinding finite-element models has been developed. In this study, we aim to develop a computationally accurate and numerically stable upwinding finite-element model within the context of Petrov-Galerkin models.

By substituting the expressions $\underline{u} = \sum_{j=1}^{27} \underline{u}_j N_j$ and $p = \sum_{l=1}^{8} p_l M_l$ for the respective $\underline{u}$ and p into the weighted residuals statement, an unsymmetric indefinite matrix system $\underline{\underline{A}}\,\underline{x} = \underline{b}$ can be derived, where $\underline{x} = \{u_j, v_j, w_j, p_l\}^T$:

$$\underline{\underline{A}} = \int_{\Omega^h} \begin{pmatrix} C_{ij} & 0 & 0 & -M_l \frac{\partial N_i}{\partial x_1} \\ 0 & C_{ij} & 0 & -M_l \frac{\partial N_i}{\partial x_2} \\ 0 & 0 & C_{ij} & -M_l \frac{\partial N_i}{\partial x_3} \\ M_l \frac{\partial N_j}{\partial x_1} & M_l \frac{\partial N_j}{\partial x_2} & M_l \frac{\partial N_j}{\partial x_3} & 0 \end{pmatrix} d\Omega^h \tag{8}$$

and

$$\underline{b} = -\int_{\Gamma^h_{out}} N_i \begin{pmatrix} pn_x - \frac{1}{\mathrm{Re}} \frac{\partial u_j}{\partial n} \\ pn_y - \frac{1}{\mathrm{Re}} \frac{\partial v_j}{\partial n} \\ pn_z - \frac{1}{\mathrm{Re}} \frac{\partial w_j}{\partial n} \\ 0 \end{pmatrix} d\Gamma^h \tag{9}$$

The components C_{ij} in (8) can be expressed using $B_i = \tau(N_m \tilde{u}_m)(\partial N_i / \partial x_k)$ as

$$C_{ij} = (N_i + B_i)(N_m \tilde{u}_m) \frac{\partial N_j}{\partial x_k} + \frac{1}{\mathrm{Re}} \frac{\partial N_i}{\partial x_k} \frac{\partial N_j}{\partial x_k} \tag{10}$$

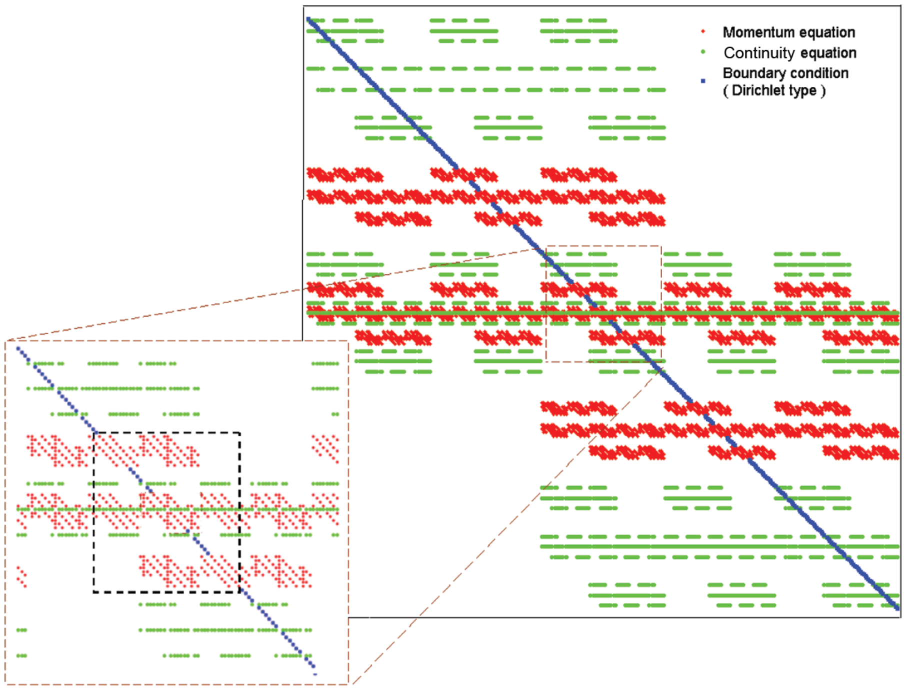
In each element, the unsymmetric matrix shown in (8) needs to be calculated sequentially. All these matrices are then mapped into their appropriate global row and column. Their assembly yields a large system of algebraic equations. We proceed to solve the resulting finite-element solutions from the sparse and unsymmetric matrix equation shown schematically in Figure 1. The number of diagonal zeros is the same as the number of continuity equations.

Our strategy of enhancing the numerical stability of the Galerkin formulation is to add a proper amount of artificial damping to the Galerkin formulation so as to suppress oscillatory velocities while maintaining the predicted accuracy. Like many artificial viscosity models, the predicted finite-element solution may be excessively smeared by the numerically introduced damping term. This type of numerical error is in particular serious as the velocity vector deviates very much from the grid line. The stabilized term B_i therefore needs to be added only to some proper upstream nodal points along the streamlines without sacrificing solution accuracy.

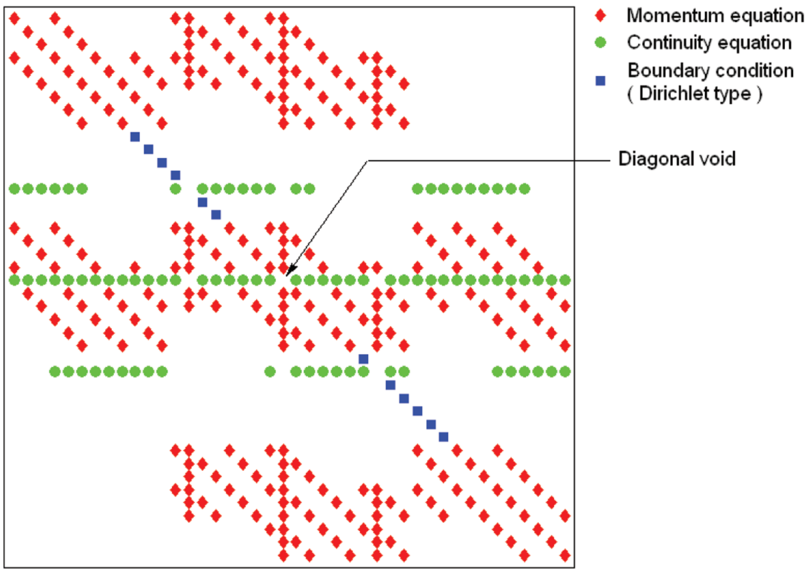
To enhance stability along the streamline in $B_i\ [\equiv \tau(N_m \tilde{u}_m)(\partial N_i / \partial x_k)]$, the coefficient τ given as

$$\tau = \frac{\delta^p \underline{u} H}{2 |\underline{u}|^2}$$

will be derived by minimizing the wavenumber error, for example, in a uniform mesh of size H. In the above relation for τ, the superscript p in δ^p is α or β, depending on its nodal classification in an element. Since the quadratic element is adopted, the upwinding coefficient δ^p will be determined at the center node δ^α and the corner



(*a*)



(*b*)

Figure 1. Illustration of a 402×402 finite-element matrix equation derived in the 2^3 tri-quadratic elements. The green, red, and blue markers represent the nonzero entries owing to the continuity equation, momentum equation, and boundary data, respectively. (*a*) Global matrix profile, (*b*) matrix profile in the dashed block of the matrix shown in color (*a*).

node δ^{β}. The coefficient δ^{p} is called *optimal* if the finite-element solution is nodally exact. In this study, we adopt the method of minimizing the numerical wavenumber error for the convection term, which is different from the way it was derived in [5].

As an example, consider the term ϕ_x in one dimension. By substituting $\phi = \sum_j N_j \phi_j$ and the weighting function $w_i = N_i + \tau u(\partial N_i/\partial x)$, where $N_i (i = 1, 2, 3)$ are quadratic basis functions, into the weighted-residuals formulation for ϕ_x, the discretized equations at the center and corner nodes are derived respectively as

$$\phi_x|_{\text{center}} \approx \frac{1}{h}\left(a_1\phi_{i-1} + a_2\phi_i + a_3\phi_{i+1}\right) \tag{11}$$

$$\phi_x|_{\text{corner}} \approx \frac{1}{h}\left(b_1\phi_{i-2} + b_2\phi_{i-1} + b_3\phi_i + b_4\phi_{i+1} + b_5\phi_{i+2}\right) \tag{12}$$

In the above, $a_1 = -\frac{1}{2} - \delta^{\alpha}, a_2 = 2\delta^{\alpha}, a_3 = \frac{1}{2} + \delta^{\alpha}, b_1 = \frac{1}{4} + \frac{1}{4}\delta^{\beta}, b_2 = -1 - 2\delta^{\beta}, b_3 = \frac{7}{2}\delta^{\beta}, b_4 = 1 - 2\delta^{\beta}$, and $b_5 = -\frac{1}{4} + \frac{1}{4}\delta^{\beta}$. As in the work of Tam and Webb [6], the Fourier transform $\tilde{\phi}(\alpha) = (1/2\pi)\int_{-\infty}^{\infty} \phi(x)\exp(-\mathbf{i}\kappa x)\,dx$ and its inverse $\phi(x) = \int_{-\infty}^{\infty} \tilde{\phi}(\kappa)\exp(\mathbf{i}\kappa x)\,d\kappa$ will be applied, where $\mathbf{i} = \sqrt{-1}$. By conducting Fourier transform on each term shown in Eqs. (11)–(12), the numerical wavenumber κ at the center node can be derived as $\kappa \approx (-\mathbf{i}/h)[a_1\exp(-\mathbf{i}\kappa h) + a_2 + a_3\exp(\mathbf{i}\kappa h)]$. The exact wavenumber $\tilde{\kappa}$ is regarded as the right-hand side of the above relation, thereby leading to

$$\tilde{\kappa} = \frac{-\mathbf{i}}{h}[a_1\exp(-\mathbf{i}\kappa h) + a_2 + a_3\exp(\mathbf{i}\kappa h)] \tag{13}$$

To make κ be close to $\tilde{\kappa}$, the integral quantity $E(\kappa)$ defined below should be a very small and positive value:

$$E(\kappa) = \int_{-\pi/2}^{\pi/2} |\kappa h - \tilde{\kappa}h|^2\, d(\kappa h) \tag{14}$$

To make E a minimum, the limiting condition $\partial E/\partial a_1$ is enforced in this study to get the coefficient δ^{α}. The coefficient δ^{β} can be derived similarly. The derived upwinding coefficients δ are summarized below.

$$\delta^{\alpha} = \frac{1}{2} \quad \text{at center node}$$

$$\delta^{\beta} = \frac{8 - 3\pi}{-22 + 6\pi} \quad \text{at corner node}$$

Determination of the stabilized coefficient τ in the multidimensional equation follows the guideline of performing operator splitting approximation to get the following artificial viscosity:

$$\tau = \frac{\delta_{\xi} V_{\xi} h_{\xi} + \delta_{\eta} V_{\eta} h_{\eta} + \delta_{\zeta} V_{\zeta} h_{\zeta}}{2 V_j V_j} \tag{15}$$

where $V_{Y_i} = \hat{\mathbf{e}}_{Y_i} \cdot \underline{u}$, and $(Y_1, Y_2, Y_3) = (\bar{\xi}, \bar{\eta}, \bar{\zeta})$.

3. ITERATIVE SOLUTION SOLVERS

Two popular classes of Krylov subspace methods developed to cope with the problem of matrix asymmetry are known as the GMRES [7] and BICGSTAB iterative solvers [8]. The GMRES method minimizes the residual over every vector in the Krylov subspace, within which the orthonormal basis is generated by the Arnoldi procedure. The storage requirement for the GMRES method increases accordingly as the iteration proceeds. To alleviate this drawback, a restart procedure has been employed to reduce the memory requirement and the computational cost of GMRES.

One can also apply a Lanczos procedure to solve the unsymmetric matrix equation, by which a nonorthogonal basis in Krylov subspace is built [8]. The BICGSTAB method of Van der Vorst is a variant of the biconjugate (BICG) method [8]. It was proposed mainly for getting a faster and smoother converged solution for the unsymmetric linear system of equations. In Lanczos context, the variants of BICG such as the conjugate gradient squared (CGS) [9] and the quasi-minimal residual (QMR) [10] are the alternatives that can be applied to resolve the difficulty regarding the matrix asymmetry. Product methods have dual orthogonal vector sets, and they are distinguished by their way of combining the construction polynomials. The QMR method of Freund and Nachtigal [11] can be applied to get rid of irregular convergence behavior, but this method still suffers from the necessity of transposing the matrix equation. On the contrary, CGS method developed on the basis of the BICG method has no need to perform transpose operations. This method, however, inhibits a rather irregular convergence of the solution because it accommodates the contraction polynomial of the BICG. Besides the transpose-free version of the QMR solver, the BICGSTAB solver is also known to be able to eliminate the oscillatory convergence behavior of the solution encountered in the CGS solver through the introduction of the product of polynomial terms presented in [8]. The BICGSTAB method is not effective to resolve irregular convergence problems when the eigenspectrum has imaginary components [12]. This situation happens quite often in advection-dominated cases [13]. Elimination of this irregular convergence problem motivated the development of the BICGSTAB2 [13] and BICGSTAB (l) [14] solvers, where l denotes the degree of polynomial.

4. ELEMENT-BY-ELEMENT ITERATIVE SOLVER

One can adopt the Gaussian elimination method together with efficient data storage to get the two-dimensional solution directly from the finite-element matrix equations. Calculation of the three-dimensional finite-element flow solution directly from the Gaussian elimination method may no longer be practical because of the substantial increase of the degrees of freedom and bandwidth. In order to reduce the memory demand and save computation time, an iterative solver will be chosen to get the finite-element solutions in three-dimensions.

Iterative solvers can be implemented together with the element-by-element (EBE) strategy [4] to avoid assembling element matrices. The reason for not necessarily forming a global system of equations from all element matrices $\underline{\underline{A}}_e$ and the element solution vectors $\underline{x}_e$ is the following. By applying the Boolean connectivity matrix $\underline{\underline{B}}_e$, which maps the entries of every element matrix $\underline{\underline{A}}_e$ into a global n -by- n matrix $\underline{\underline{A}}$, the global matrix–vector product $\underline{\underline{A}}\ \underline{x}$ can be formulated at element level by means of $\underline{\underline{A}}\underline{x} = \left(\sum_{e=1}^{n} \underline{\underline{B}}_e^T \underline{\underline{A}}_e \underline{\underline{B}}_e\right)\underline{x} = \sum_{e=1}^{n} \underline{\underline{B}}_e^T \left(\underline{\underline{A}}_e \underline{x}_e\right)$. A global matrix–vector product can be, as a result, represented by the operations carried out at element level for $\underline{\underline{A}}_e \underline{x}_e$. Any iterative method applied in conjunction with the element-by-element strategy has, therefore, a much lower memory demand. Because of this advantage, a large three-dimensional flow simulation can be carried out on a relatively modest computer. As a result, the EBE concept has been incorporated into the solution procedure when solving the linear algebraic equations by the iterative solver described below.

The Krylov subspace method will be chosen in this study to solve the sparse matrix equations iteratively. The finite-element solution is updated from an initial guess solution $\underline{x}_0$ through $\underline{x}_0 \in \underline{x}_0 + \underline{\mathcal{K}}_k$. The kth Krylov subspace $\underline{\mathcal{K}}_k$ is the span of $\{\underline{r}_0, \underline{\underline{A}}\ \underline{r}_0, \ldots, \underline{\underline{A}}^{k-1}\underline{r}_0\}$, where $\underline{\underline{A}}$ is the matrix equation and $\underline{r}_0\left(\equiv \underline{b} - \underline{\underline{A}}\ \underline{x}_0\right)$ is the initial residual vector. In this study the orthogonal residual (OR) method, which is one of the Krylov methods, is chosen. This method iterates the solution by the orthogonal means $(\underline{r}_k)^T \underline{\underline{V}}_k = 0$ between the residual vector $\underline{r}_k$ and the matrix $\underline{\underline{V}}_k$, whose columns are the basis vectors of the Krylov subspace $\underline{\mathcal{K}}_k$.

The orthogonal residual method may break down for a non-SPD (symmetric positive definite) matrix. As a result, the unsymmetric indefinite finite-element matrix equation shown in (8) is transformed first to its equivalent symmetric positive-definite matrix form. The conjugate gradient normal residual (CGNR) method is adopted to multiply first the matrix system $\underline{\underline{A}}\underline{x} = \underline{b}$ by $\underline{\underline{A}}^T$ to get the normal equation $\underline{\underline{A}}^T\underline{\underline{A}}\underline{x} = \underline{\underline{A}}^T\underline{b}$. The resulting matrix $\underline{\underline{A}}^T\underline{\underline{A}}$ becomes symmetric and positive-definite. The conjugate gradient (CG) Krylov iterative method of Hestenes and Stiefel [15] can then be applied. The convergence bound of the resulting symmetric positive-definite linear system depends on the condition number of the matrix equation.

The CGNR Krylov method changes the original unsymmetric matrix to its symmetric counterpart, and use of this method in turn alters the distribution of eigenvalues. More precisely, the nature of the matrix is changed from an indefinite matrix featuring imaginary eigenvalues to a definite matrix containing only real eigenvalues. However, use of this technique increases the condition number of the system. Note that the condition number of $\underline{\underline{A}}^T\underline{\underline{A}}$ is the square of the condition number of the original matrix $\underline{\underline{A}}$. Because of the increase of condition number in the normal equation $\underline{\underline{A}}^T\underline{\underline{A}}\underline{x} = \underline{\underline{A}}^T\underline{b}$, the convergence of CG iteration becomes far too slow. To overcome this drawback, the matrix equation needs to be preconditioned using a properly chosen preconditioner, prior to application of the CG iterative solver.

To make full use of the EBE advantage, one should choose a proper preconditioner and then multiply it with the normal matrix equation from right, left, or both sides to reduce the condition number. While an efficient preconditioner can be derived by the methods of successive overrelaxation [16], incomplete Cholesky factorization [17], polynomial expansion [18], and multilevel multigrid [19], they all require constructing their own global preconditioners. Such a requirement imposes a heavy demand on the storage capacity. Therefore, one should choose an EBE-based preconditioner. Besides the simplest EBE-based Jacobi preconditioner, one can also adopt other more complex EBE-based preconditioners developed, for example, from the approximate Cholesky factorization [20] on LU splitting technique [21]. Other applicable preconditioners can be found in [22].

Having normalized the mixed finite-element matrix equations, we can now apply the conjugate gradient iterative solver to get the finite-element solution from the resulting symmetric positive-definite matrix equation. In order to reduce the memory requirement while solving the three-dimensional incompressible viscous equations, an element-by-element strategy is adopted, and it is implemented together with the conjugate gradient iterative solver. The element-by-element CGNR iterative solver applied in the current study is as described below:

Algorithm 1. The element-by-element CGNR solver for $\underline{\underline{A}}^T \underline{\underline{A}} \underline{x} = \underline{\underline{A}}^T \underline{b}$

```
Starting from an initial guess $\underline{x}_0$
Compute $\underline{x}_0 = \sum_e \underline{\underline{A}}_e \underline{x}_0, \underline{x}''_0 = \sum_e \underline{\underline{A}}_e^T \underline{x}'_0, \; \underline{b}' = \sum_e \underline{\underline{A}}_e^T \underline{b} \leftarrow$ EBE procedure
Compute the initial residual $\underline{r}_0 = \underline{b}' - \underline{x}''_0$
$\underline{p}_0 = \underline{r}_0$
For $j = 1, 2, \ldots,$
  $\underline{p}'_{j-1} = \sum_e \left( \underline{\underline{A}}_e \underline{p}_{j-1} \right) \leftarrow$ EBE procedure
  $\underline{p}''_{j-1} = \sum_e \left( \underline{\underline{A}}_e^T \underline{p}'_{j-1} \right) \leftarrow$ EBE procedure
  $\alpha_{j-1} = \left( \underline{p}_{j-1}, \underline{r}_{j-1} \right) \Big/ \left( \underline{p}_{j-1}, \underline{p}''_{j-1} \right)$
  $\underline{x}_j = \underline{x}_{j-1} + \alpha_{j-1} \underline{p}_{j-1}$
  $\underline{r}_j = \underline{r}_{j-1} - \alpha_{j-1} \underline{p}''_{j-1}$
  Convergence check
$\beta_{j-1} = (\underline{r}_j, \underline{r}_j) / (\underline{r}_{j-1}, \underline{r}_{j-1})$
$\underline{p}_j = \underline{r}_j + \beta_{j-1} \underline{p}_{j-1}$
End
```

5. IMPLEMENTATION OF POLYNOMIAL PRECONDITIONERS

To reduce the matrix condition number without deterioration of prediction accuracy, the finite-element matrix equation derived in Section 2 is preconditioned prior to calculation of the solution through iterative steps. In this study, two preconditioners belonging to the class of polynomial preconditioners will be chosen.

Within the CGNR solution context, the normalized matrix $\underline{\underline{A}}^T \underline{\underline{A}}$ is preconditioned first. This symmetric positive-definite matrix can be rewritten as $\underline{\underline{A}}^T \underline{\underline{A}} = \underline{\underline{D}} \left[\underline{\underline{I}} - \left(\underline{\underline{I}} - \underline{\underline{D}}^{-1} \underline{\underline{A}}^T \underline{\underline{A}} \right) \right]$ or $\left(\underline{\underline{A}}^T \underline{\underline{A}} \right)^{-1} = \left[\underline{\underline{I}} - \left(\underline{\underline{I}} - \underline{\underline{D}}^{-1} \underline{\underline{A}}^T \underline{\underline{A}} \right)^{-1} \right] \underline{\underline{D}}^{-1}$, where $\underline{\underline{D}}$ denotes the diagonal matrix of $\underline{\underline{A}}^T \underline{\underline{A}}$. Approximation of the matrix $(\underline{\underline{A}}^T \underline{\underline{A}})^{-1}$ starts with defining the matrix given by $\underline{\underline{G}} = \underline{\underline{I}} - \underline{\underline{D}}^{-1} \underline{\underline{A}}^T \underline{\underline{A}}$. The matrix $\underline{\underline{A}}^T \underline{\underline{A}}$ can then be expressed in terms of the n-by-n matrix $\underline{\underline{G}}$ as $(\underline{\underline{A}}^T \underline{\underline{A}})^{-1} = (\underline{\underline{I}} - \underline{\underline{G}})^{-1} \underline{\underline{D}}^{-1}$. Given this matrix $\underline{\underline{G}}$, one can rewrite $(\underline{\underline{I}} - \underline{\underline{G}})^{-1}$ analytically in a form of binomial series as $(\underline{\underline{I}} - \underline{\underline{G}})^{-1} = \sum_{k=0}^{\infty} \underline{\underline{G}}^k$, provided that the spectral radius of $\underline{\underline{G}}$ or $\rho(\underline{\underline{G}})$ has a value between -1 and 1. The inverse of the symmetric positive-definite matrix $\underline{\underline{A}}^T \underline{\underline{A}}$ can be expressed analytically terms of the following infinite series of matrix summation:

$$\left(\underline{\underline{A}}^T\underline{\underline{A}}\right)^{-1} = \sum_{k=0}^{\infty} \underline{\underline{G}}^k \underline{\underline{D}}^{-1} \tag{16}$$

Approximation of $\left(\underline{\underline{A}}^T\underline{\underline{A}}\right)^{-1}$ by choosing $k = 0$ and $k = 1$ leads respectively to $\underline{\underline{D}}^{-1}$ and $\underline{\underline{GD}}^{-1}$, which correspond to preconditioning the matrix equation $\underline{\underline{A}}^T\underline{\underline{A}}$ by employing the currently employed Jacobi preconditioner and polynomial preconditioner, respectively.

6. ELEMENT-BY-ELEMENT PRECONDITIONED ITERATIVE SOLVERS

While finite-element matrix equations become positive-definite after normalizing the original mixed finite-element matrix equations, the resulting matrix condition number increases substantially. To accelerate convergence during calculation of the finite-element solution iteratively from the CGNR matrix, two preconditioners described in the previous section are adopted. The first preconditioner is known as the Jacobi preconditioner $\underline{\underline{M}}_J$, which is nothing but the matrix containing only the diagonal part of the matrix $\underline{\underline{A}}^T\underline{\underline{A}}$, i.e.,

$$\underline{\underline{M}}_J = \mathrm{diag}(\underline{\underline{A}}^T\underline{\underline{A}}) \tag{17}$$

The second preconditioner chosen in this study is the polynomial preconditioner $\underline{\underline{M}}_P$ given below:

$$\underline{\underline{M}}_P = 2\underline{\underline{D}}^{-1} - \underline{\underline{D}}^{-1}\underline{\underline{A}}^T\underline{\underline{AD}}^{-1} \tag{18}$$

Algorithm 2. The preconditioned CGNR solver for $\underline{\underline{A}}^T\underline{\underline{A}}\underline{x} = \underline{\underline{A}}^T\underline{b}$

```
Starting from an initial guess $\underline{x}_0$
Compute $\underline{x}'_0 = \sum_e \underline{\underline{A}}_e \underline{x}_0$, $\underline{x}''_0 = \sum_e \underline{\underline{A}}_e^T \underline{x}'_0$, $\underline{b}' = \sum_e \underline{\underline{A}}_e^T \underline{b} \leftarrow$ **EBE procedure**
Compute the initial residual $\underline{r}_0 = \underline{b}' - \underline{x}''_0$
$\underline{z}_0 = \underline{\underline{M}}^{-1}\underline{r}_0 \leftarrow \underline{\underline{M}} = \underline{\underline{M}}_J$ or $\underline{\underline{M}}_P$
$\underline{p}_0 = \underline{z}_0$
**For** $j = 1, 2, \ldots,$
  $\underline{p}'_{j-1} = \sum_e \left(\underline{\underline{A}}_e \underline{p}_{j-1}\right) \leftarrow$ **EBE procedure**
  $\underline{p}''_{j-1} = \sum_e \left(\underline{\underline{A}}_e^T \underline{p}'_{j-1}\right) \leftarrow$ **EBE procedure**
  $\alpha_{j-1} = \left(\underline{p}_{j-1}, \underline{r}_{j-1}\right) / \left(\underline{p}_{j-1}, \underline{p}''_{j-1}\right)$
  $\underline{x}_j = \underline{x}_{j-1} + \alpha_{j-1}\underline{p}_{j-1}$
  $\underline{r}_j = \underline{r}_{j-1} - \alpha_{j-1}\underline{p}''_{j-1}$
**Convergence check**
  $\underline{z}_j = \underline{\underline{M}}^{-1}\underline{r}_j \leftarrow \underline{\underline{M}} = \underline{\underline{M}}_J$ or $\underline{\underline{M}} = \underline{\underline{M}}_P$
  $\beta_{j-1} = (\underline{z}_j, \underline{r}_j)/(\underline{r}_{j-1}, \underline{r}_{j-1})$
  $\underline{p}_j = \underline{z}_j + \beta_{j-1}\underline{p}_{j-1}$
**End**
```

Prior to applying the above element-by-element preconditioned iterative solver to solve the large-size matrix equation, it is instructive first to perform calculation in the smallest set of 2^3 tri-quadratic elements. The polynomial preconditioner $\underline{\underline{M}}_P$ can be employed, provided that the spectral radius of the matrix G is less than 1. For this reason, the eigenvalues of $\underline{\underline{G}}$ are plotted in Figure 2, which shows that not all the eigenvalues of $\underline{\underline{G}}$ lie in the unit circle. This means that the spectral radius is greater than 1, implying that iterative solvers cannot be applied to get a convergent solution.

To make the spectral radius of $\underline{\underline{G}}$ smaller than 1, the so-called scaling parameter ω is introduced for the approximation of $\underline{\underline{A}}^T\underline{\underline{A}}$. In the derivation, we start with $\omega\underline{\underline{A}}^T\underline{\underline{A}}$ instead of $\underline{\underline{A}}^T\underline{\underline{A}}$. Following the same derivation procedure described in Section 4, the inverse of $\underline{\underline{A}}^T\underline{\underline{A}}$ can be derived as $\omega(\underline{\underline{I}} - \underline{\underline{G}}_S^{-1})\underline{\underline{D}}^{-1}$, where $\underline{\underline{G}}_S = \underline{\underline{I}} - \omega\underline{\underline{D}}^{-1}\underline{\underline{A}}^T\underline{\underline{A}}$. By employing the theorem of binomial series, the scaling polynomial preconditioner is derived as

$$\underline{\underline{M}}_{SP} = \omega\left(\underline{\underline{I}} - \underline{\underline{G}}_S\right)\underline{\underline{D}}^{-1} = 2\omega\underline{\underline{D}}^{-1} - \omega^2\underline{\underline{D}}^{-1}\underline{\underline{A}}^T\underline{\underline{A}}\underline{\underline{D}}^{-1} \tag{19}$$

We determine the user's specified scaling parameter ω in such a way that the spectral radius of $\underline{\underline{G}}_S$ should be less than unity. For choosing a proper range of ω, we perform as before the finite-element calculation in a domain of 2^3 tri-quadratic elements. In Figure 3, the spectral radius ρ is plotted with respect to the scaling parameter ω for the matrix equation $\underline{\underline{G}}_S$. It can be seen that the spectral radius is less than 1 in the range of $0 \leq \omega \leq 0.1$. In this study, we fix $\omega = 0.05$ in the rest of our calculations. For completeness, the eigenvalues calculated at $\omega = 0.05$ are also plotted in Figure 4.

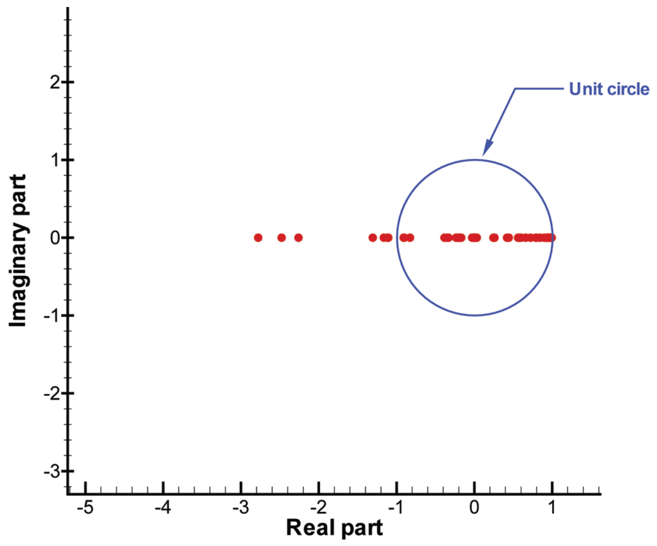


Figure 2. Computed eigenvalues from the matrix equation $\underline{\underline{I}} - \underline{\underline{D}}^{-1}\underline{\underline{A}}^T\underline{\underline{A}}$ in a domain of 2^3 elements.

For the sake of comparison, two highly recommended BICG-based iterative methods are also applied. Since BICGSTAB typically has a much smoother convergence than CGS [23], the following element-by-element BICGSTAB method described in [24] will be used for the assessment purpose. We aim to show that the norm of the BICGSTAB residual still oscillates considerably when solving the investigated three-dimensional lid-driven cavity problem.

Algorithm 3. The preconditioned BICGSTAB solver for $\underline{\underline{A}}\underline{x} = \underline{b}$

```
Starting from an initial guess $\underline{x}_0$
Compute the initial residual vector $\underline{r}_0 = \underline{b} - \underline{\underline{A}}\underline{x}_0$. Choose $\underline{r}'$, such that $(\underline{r}', \underline{r}_0) \neq 0$
For $j = 1, 2, \ldots,$
  $\rho_{j-1} = (\underline{r}', \underline{r}_{j-1})$
  if $\rho_{j-1} < \varepsilon_1$ (near breakdown)
  if $j = 1$
    $\underline{p}_j = \underline{r}_{j-1}$
  else
    $\beta_{j-1} = (\rho_{j-1}/\rho_{j-2})/(\alpha_{j-1}/\omega_{j-1})$
    $\underline{p}_j = \underline{r}_{j-1} + \beta_{j-1}\left(\underline{p}_{j-1} - \omega_{j-1}\underline{v}_{j-1}\right)$
  end if
  $\underline{p}'_j = \underline{\underline{M}}^{-1}\underline{p}_j$
  $\underline{v}_j = \sum_e\left(\underline{\underline{A}}_e\underline{p}'_j\right) \leftarrow$ EBE procedure
  $\alpha_j = \rho_{j-1}\Big/\left(\underline{r}', \underline{v}_j\right)$
  if $(\underline{r}', \underline{v}_j) < \varepsilon_2$ (near breakdown)
```

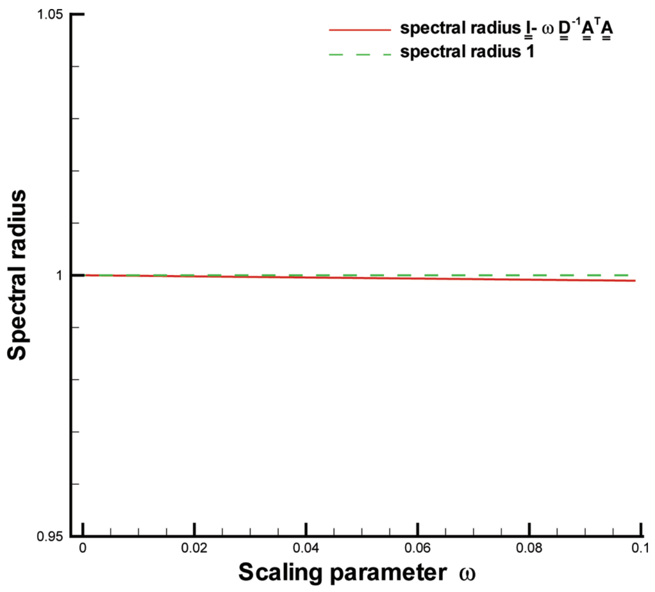


Figure 3. Predicted spectral radius versus scaling parameter ω.

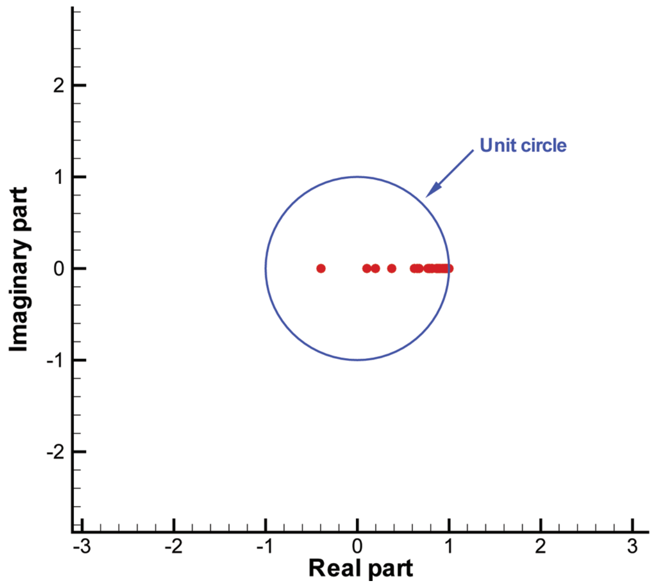


Figure 4. Computed eigenvalues for the matrix equation $\underline{\underline{I}} - \omega \underline{\underline{D}}^{-1} \underline{\underline{A}}^T \underline{\underline{A}}$ in a domain of 2^3 elements.

```
    $\underline{s}_j = \underline{r}_{j-1} - \alpha_j \underline{v}_j$
    $\underline{s}'_j = \underline{\underline{M}}^{-1} \underline{s}_j$
    $t = \sum_e \left( \underline{\underline{A}}_e \underline{s}'_j \right)$ ← **EBE procedure**
    **if** $||\underline{s}_j||_2 < \varepsilon$
        $\omega_j = 0$
    **else**
        $\omega_j = (t, s)/(t, t)$
    **end if**
    $\underline{x}_j = \underline{x}_{j-1} + \alpha_j \underline{p}'_j + \omega_j \underline{s}'_j$
    $\underline{r}_j = \underline{s}_j - \omega_j t$
    Convergence check
**End**
```

The other well-known iterative solver, GMRES, is also applied in conjunction with the Jacobi preconditioner. The GMRES iterative solver implemented in an element-by-element fashion in [24] is given below.

Algorithm 4. The preconditioned GMRES solver for $\underline{\underline{A}}\underline{x} = \underline{b}$

```
Starting from an initial guess $\underline{x}_0$
**For** $j = 1, 2, \ldots,$
    $\underline{r}_j = \underline{b} - \underline{\underline{A}} \underline{x}_0$ ← **EBE procedure**
    **Solve** $\underline{r}_j = \underline{\underline{M}}^{-1} \underline{r}_j$
    $\underline{v}_1 = \underline{r}_j / ||\underline{r}_j||_2$
    $s := ||\underline{r}_j||$
```

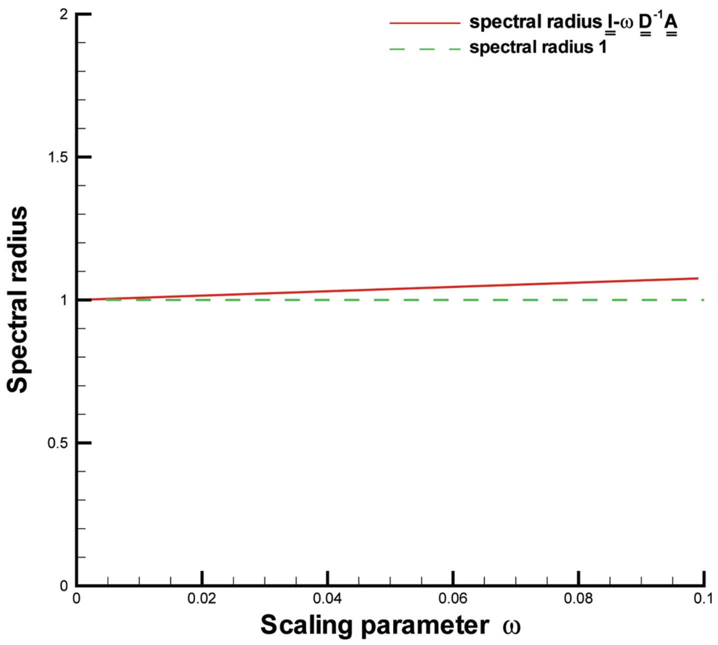


Figure 5. Predicted spectral radius versus scaling parameter ω.

```
for $i = 1, 2, 3, \ldots, m$
  $\underline{z} = \underline{\underline{A}}\underline{v}_j \leftarrow$ EBE procedure
  Solve $\underline{w} = \underline{\underline{M}}^{-1}\underline{z}$
```

Table 1. Computed L_2 error norms for the problem given in Section 6.1

		Number of elements		
Solver		4^3	8^3	16^3
Frontal	u	1.5322×10^{-3}	8.2419×10^{-4}	—
	v	8.9738×10^{-3}	5.4890×10^{-3}	—
	w	9.3387×10^{-3}	6.0440×10^{-3}	—
	p	3.1943×10^{-2}	2.0691×10^{-2}	—
BICGSTAB	u	1.4609×10^{-3}	8.2539×10^{-4}	4.1915×10^{-4}
	v	8.9158×10^{-3}	5.5351×10^{-3}	2.7610×10^{-3}
	w	9.4792×10^{-3}	6.0045×10^{-3}	3.0656×10^{-3}
	p	3.0718×10^{-2}	1.4773×10^{-2}	6.6294×10^{-3}
GMRES	u	1.4609×10^{-3}	8.2539×10^{-4}	4.1915×10^{-4}
	v	8.9158×10^{-3}	5.5351×10^{-3}	2.7613×10^{-3}
	w	9.4792×10^{-3}	6.0045×10^{-3}	3.0656×10^{-3}
	p	3.0737×10^{-2}	1.4814×10^{-3}	6.6954×10^{-3}
CGNR	u	1.4609×10^{-3}	8.2540×10^{-4}	4.1913×10^{-4}
	v	8.9158×10^{-3}	5.5351×10^{-3}	2.7610×10^{-3}
	w	9.4792×10^{-3}	6.0045×10^{-3}	3.0661×10^{-3}
	p	3.5336×10^{-2}	1.4813×10^{-2}	6.6612×10^{-3}

In this table, "—" means that the solution is not computable.

Table 2. Total CPU times needed for the execution of one entire calculation and CPU times spent on the three chosen iterative solvers for the calculations involving 16^3 elements

Solver	Total CPU time (s)	CPU time (s) and percentage in (%) for iterative solver
BICGSTAB	4,608.6	4,527.5 (98.24%)
GMRES	11,459.1	11,257.4 (98.23%)
CGNR	3,015.2	2,967.4 (98.71%)

```
for $k = 1,\ldots, i$
  $h_{k,i} = (\underline{w}, \underline{v}_k)$                                   475
  $\underline{w} = \underline{w} - h_{k,i}\underline{v}_k$
end
$h_{i+1,i} = ||\underline{w}||$
$\underline{v} = \underline{w}/h_{i+1,i}$
Apply $J_1, \ldots, J_{i-1}$ on $(h_{1,i}), \ldots, (h_{i+1,i})$                   480
Construct $j_i$, acting on the $i$th and $(i+1)$th components of $h_{.,i}$
  such that the $(i+1)$th component of $J_i h_{.,i}$ has the value of 0
$s := J_i s$
```

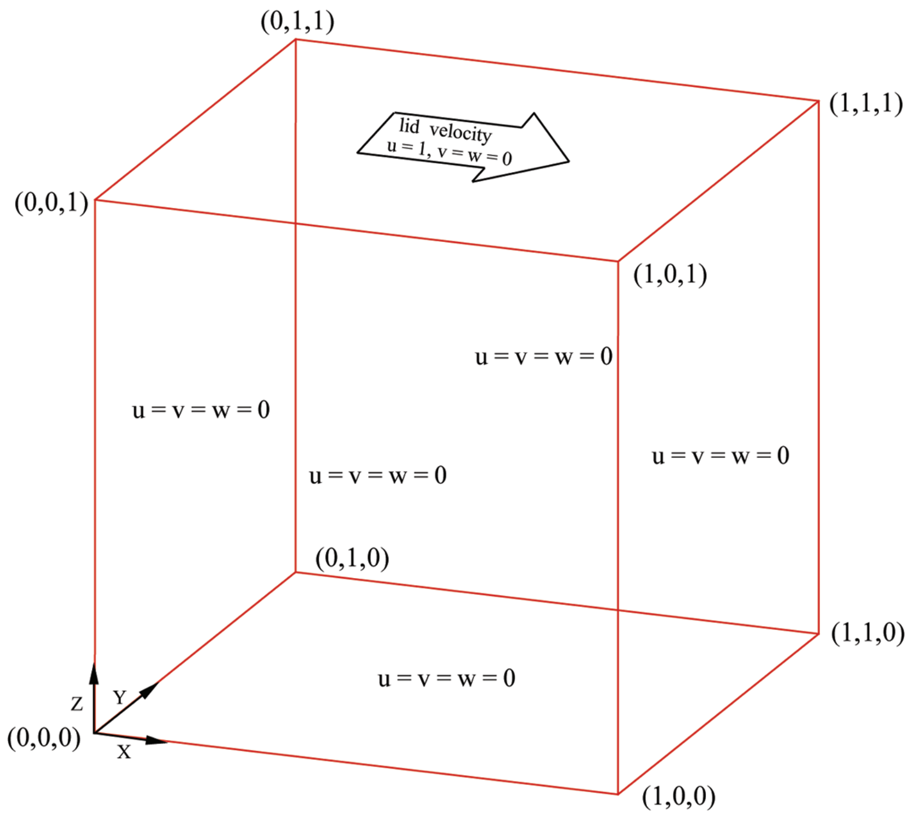
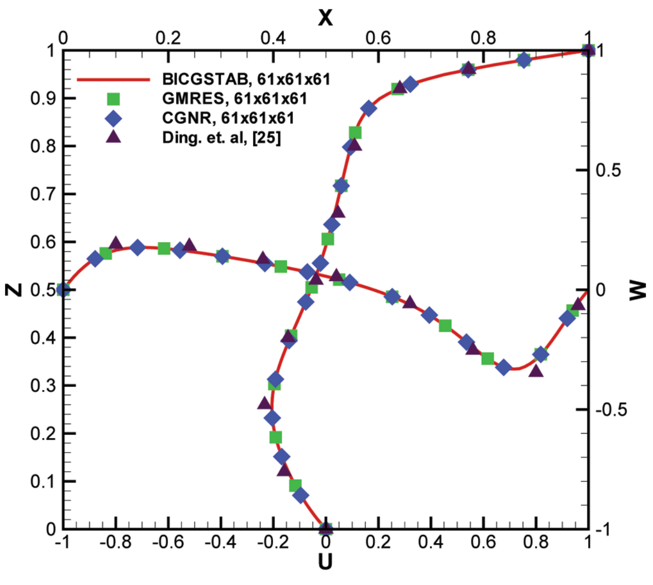
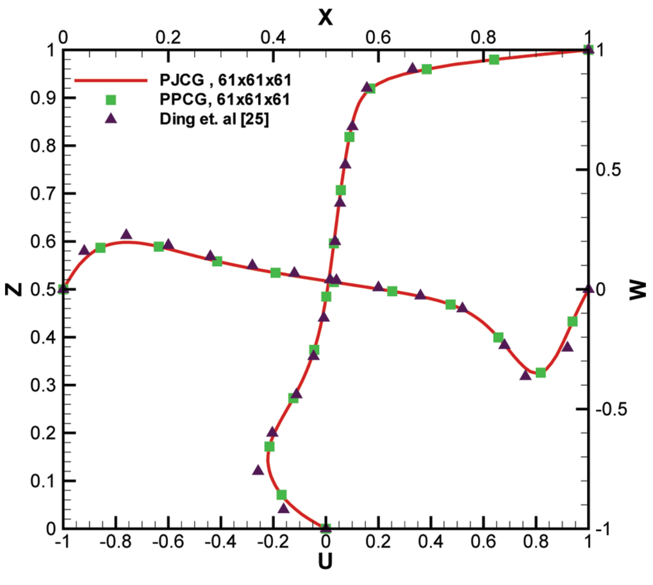


Figure 6. Illustration of the three-dimensional lid-driven cavity problem.

```
  if $s(i+1)$ becomes small enough, then UPDATE ($\tilde{x}$, $i$) and quit
end
UPDATE ($\underset{\sim}{x}, m$)
End
```



(*a*)



(*b*)

Figure 7. Comparison of the predicted velocity profiles at the mid-plane $y = 0.5$. (*a*) $\mathrm{Re} = 400$, (*b*) $\mathrm{Re} = 1{,}000$.

Table 3. Finite-element results computed from three iterative solvers for the problem investigated in a domain of 21^3 nodal points

Solver	Reynolds number	CPU time (s)	No. of outer iterations	No. of inner iterations at the *n*th outer iteration
BICGSTAB	100	682.7	8	1,465 ($n=4$)
	400	×	×	Breaks down
	1,000	×	×	Breaks down
GMRES	100	300.2	6	430 ($n=3$)
	400	1,238.4	8	1,350 ($n=4$)
	1,000	2,623.3	9	2,120 ($n=5$)
CGNR	100	795.4	8	2,073 ($n=4$)
	400	862.2	8	2,788 ($n=4$)
	1,000	1,937.7	10	5,463 ($n=5$)

The notation "×" means that the solution is not computable.

```
The UPDATE ($\tilde{x}$, $i$) procedure is as follows.
    Compute $\underline{y}$ from $\underline{\underline{H}}\underline{y} = \underline{s}$
    in which the upper $i \times i$ triangular part of $\underline{\underline{H}}$ has $h_{i,j}$ as its elements.
    $\underline{s}$ is the first $i$ components of $s$
$\underline{x} = \underline{x}_0 + \underline{y}_1\underline{v}_1 + \underline{y}_2\underline{v}_2 + \ldots + \underline{y}_i\underline{v}_i$
$s_{i+1} = ||\underline{b} - \underline{\underline{A}}\tilde{\underline{x}}|| \leftarrow$ EBE procedure
If $\tilde{\underline{x}}$ is accurate enough, then terminate the calculation
else $\underline{x}_0 = \tilde{\underline{x}}$
```

It is worth noting that the polynomial preconditioner is not applicable to indefinite matrix calculation using either the GMRES or the BICGSTAB solver. Hence only the Jacobi preconditioner will be implemented in the GMRES and BICGSTAB iterative solvers. As before, the finite-element calculation in 2^3 tri-quadratic elements is carried out. One can clearly see from Figure 5 that the spectral radius can by no means be less than unity.

Table 4. Finite-element results computed from three iterative solvers for the problem investigated in a domain of 41^3 nodal points

Solver	Reynolds number	CPU time (s)	No. of outer iterations	No. of inner iterations at the *n*th outer iteration
BICGSTAB	100	5,853.1	7	3,040 ($n=3$)
	400	30,622.5	8	5,585 ($n=4$)
	1,000	×	×	Breaks down
GMRES	100	4,652.1	6	850 ($n=3$)
	400	12,468.4	7	1,760 ($n=4$)
	1,000	×	×	Breaks down
CGNR	100	4,384.9	6	3,460 ($n=3$)
	400	9,490.4	7	4,428 ($n=4$)
	1,000	29,845.3	11	7,611 ($n=6$)

The notation "×" means that the solution is not computable.

Table 5. Finite-element results computed from three iterative solvers for the problem investigated in a domain of 61^3 nodal points

Solver	Reynolds number	CPU time (s)	No. of outer iterations	No. of inner iterations at the *n*th outer iteration
BICGSTAB	100	57,630.5	6	3,090 ($n=3$)
	400	172,855.2	8	9,995 ($n=4$)
	1,000	×	×	Breaks down
GMRES	100	71,598.2	6	2,840 ($n=3$)
	400	151,312.7	7	4,070 ($n=4$)
	1,000	×	×	Breaks down
CGNR	100	46,097.8	6	4,281 ($n=3$)
	400	88,192.9	8	6,001 ($n=4$)
	1,000	272,705.7	12	10,657 ($n=6$)

The notation "×" means that the solution is not computable.

7. NUMERICAL RESULTS

7.1. Verification Study

The first step toward verification of the proposed incompressible Navier-Stokes finite-element model is to solve equations amenable to analytical solution. The 505
problem under investigation is defined in a hexahedron of length 1. Along the cube surfaces, the nodal velocities are analytically prescribed by $u = \frac{1}{2}(y^2 + z^2), v = -z$, and $w = y$. The corresponding exact pressure takes the form

$$p = \frac{1}{2}\left(y^2 + z^2\right) + \frac{2}{\mathrm{Re}}x$$

In this finite-element verification study, which involves a total number of N unknowns, the memory is estimated to be $O(N^{4/3})$ using the Frontal solver. Such

Table 6. Finite-element results computed from three iterative solvers for the problem investigated in a domain of 21^3 nodal points

Solver	Reynolds number	CPU time (s)	No. of outer iterations	No. of inner iterations at the *n*th outer iteration
PBICGSTAB	100	294.6	6	1,400 ($n=3$)
	400	3,852.5	8	1,870 ($n=4$)
	1,000	×	×	Breaks down
PGMRES	100	254.1	6	300 ($n=3$)
	400	1,033.5	8	930 ($n=4$)
	1,000	2,094.9	9	1,470 ($n=5$)
PJCG	100	93.2	6	364 ($n=3$)
	400	362.5	8	1,230 ($n=4$)
	1,000	1,466.9	15	2631 ($n=8$)
PPCG	100	107.5	6	361 ($n=3$)
	400	425.4	8	1,229 ($n=4$)
	1,000	1,051.4	9	2,600 ($n=5$)

The notation "×" means that the solution is not computable. PJCG, CG iterative solver used together with Jacobi preconditioner; PPCG, CG iterative solver used together with polynomial preconditioner.

Table 7. Finite-element results computed from three iterative solvers for the problem investigated in a domain of 41^3 nodal points

Solver	Reynolds number	CPU time (s)	No. of outer iterations	No. of inner iterations at the *n*th outer iteration
PBICGSTAB	100	5,510.3	6	2,600 ($n=3$)
	400	10,297.1	8	4,870 ($n=4$)
	1,000	×	×	Breaks down
PGMRES	100	3,935.1	6	600 ($n=3$)
	400	10,537.0	7	1,610 ($n=4$)
	1,000	×	×	Breaks down
PJCG	100	1,918.3	6	986 ($n=3$)
	400	4,248.3	7	1,960 ($n=4$)
	1,000	13,167.7	9	5,017 ($n=5$)
PPCG	100	2,711.5	6	978 ($n=3$)
	400	5,799.5	7	1,498 ($n=4$)
	1,000	12,230.8	9	3,301 ($n=5$)

The notation "×" means that the solution is not computable.

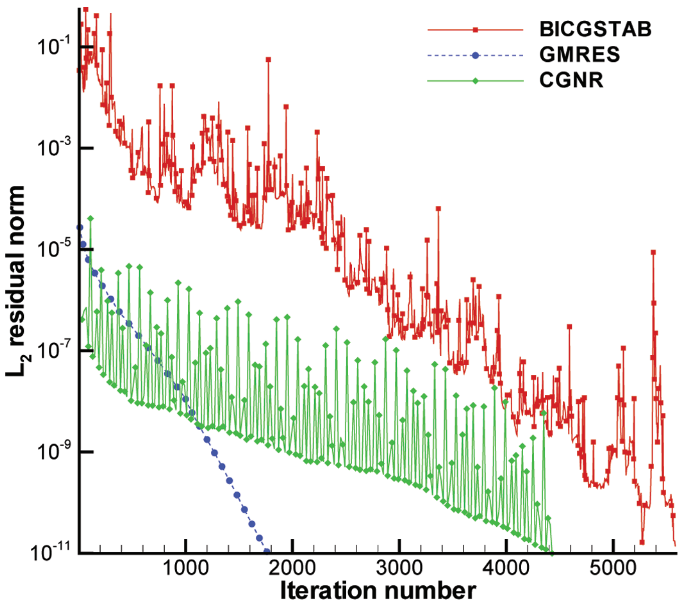
a large memory demand prohibits us from getting an accurate solution effectively using the direct solver.

Three uniform meshes have been used to perform the convergence test. Table 1 tabulates the computed L_2 errors, from which the iterative solutions are seen to be compatible with the Frontal direct solutions. To illustrate the error-reduction history, the L_2 residuals are plotted with respect to the iteration number. The CPU times are summarized in Table 2. From these results, we can conclude that the direct Frontal solver is preferable to iterative solvers when solving a smaller-size problem. Much of the CPU time is consumed in the inner iteration. This points out the difficulty of solving unsymmetric indefinite matrix equations.

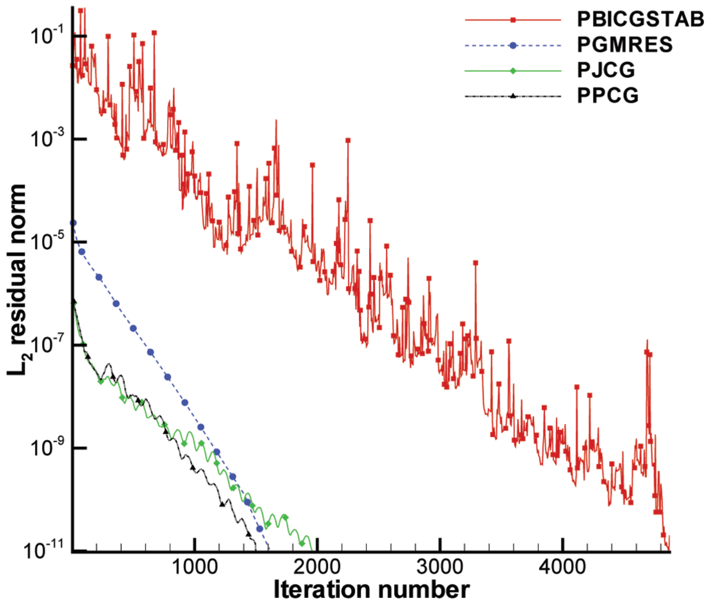
Table 8. Finite-element results computed from three iterative solvers for the problem investigated in a domain of 61^3 nodal points

Solver	Reynolds number	CPU time (s)	No. of outer iterations	No. of inner iterations at the *n*th outer iteration
PBICGSTAB	100	49,805.3	7	5,455 ($n=4$)
	400	98,022.3	8	6,615 ($n=4$)
	1,000	×	×	Breaks down
PGMRES	100	36,839.9	6	930 ($n=3$)
	400	82,338.9	7	1,670 ($n=4$)
	1,000	×	×	Breaks down
PJCG	100	25,153.4	6	1,996 ($n=3$)
	400	46,770.4	10	2,149 ($n=5$)
	1000	172,560.3	15	6,218 ($n=7$)
PPCG	100	32,717.1	6	2,395 ($n=3$)
	400	51,790.1	7	2,960 ($n=4$)
	1,000	133,386.8	11	5,329 ($n=5$)

The notation "×" means that the solution is not computable.



(*a*)



(*b*)

Figure 8. Residual reduction plots in the inner iteration (at the fourth outer iteration) using the investigated iterative solvers to solve the Navier-Stokes equations at Re = 400 in 41^3 nodal points. (*a*) Nonpreconditioned solvers; (*b*) preconditioned solvers.

7.2. Lid-Driven Cavity Flow in a Cube

The lid-driven cavity problem presented schematically in Figure 6 is considered for assessment of the three iterative solvers investigated. To begin with, the predicted and referenced [25] mid-line velocity profiles are plotted in Figure 7. This shows a good match of the two solutions. Since one of our objectives is to explain the necessity of normalizing the matrix equation, the computed results using the BICG-STAB and GMRES solvers for the original matrix and the CGNR solver for the normal matrix equation are also tabulated in Tables 3–5. Unlike the other solvers, steady solution can be obtained by the CGNR solver for each case. The necessity of performing matrix normalization is therefore amply demonstrated.

While matrix normalization can be applied to get the steady-state solution using the CGNR solver, it will also bring in two drawbacks. One drawback is related to the increased condition number, and the other difficulty involves performing additional matrix–vector multiplication. Since the convergence behavior of the Krylov subspace solver is very sensitive to the condition number, a larger condition number makes the CGNR solver converge slowly or even diverge. In Tables 3–5, the inner iteration number for the CGNR solver is greater than those of the BICGSTAB and GMRES solvers. To reduce condition number and improve convergence, we adopt the Jacobi and polynomial preconditioners. The numerical results of CGNR and CGNR used together with the Jacobi preconditioner and polynomial preconditioner are shown in Tables 6–8. For completeness, the results obtained from the BICGSTAB and GMRES iterative solvers along with the use of Jacobi preconditioner are also shown

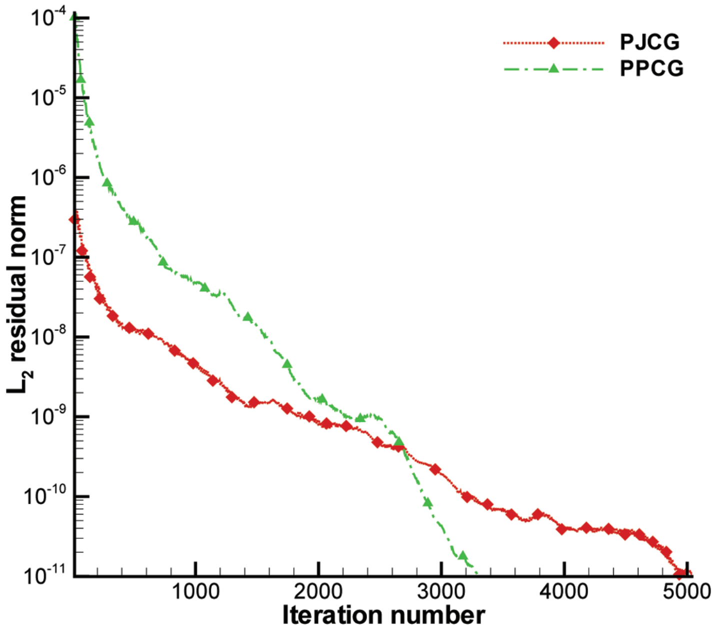


Figure 9. Residual reduction plots in the inner iteration (at the fifth outer iteration) using the investigated preconditioned iterative solvers to solve the Navier-Stokes equations at Re = 1,000 in 41^3 nodal points.

in these tables. One can clearly see from Figure 8 that for the case Re = 400, the number of inner iterations for the solver used together with the preconditioner is much smaller compared to the original CGNR solver. As the Reynolds number is increased to 1,000, one can see from Figure 9 that use of the polynomial preconditioner outperforms the Jacobi preconditioner in reducing the iteration number within the context of the proposed CGNR iterative solver.

8. CONCLUDING REMARKS

The current finite-element calculation of three-dimensional incompressible Navier-Stokes equations has been carried out in tri-quadratic elements. The basis functions for the primitive variables $\underline{u}$ and p satisfy the compatability condition. To enhance numerical stability in association with the predicted velocity, a stabilization term is added along the streamline direction within the Petorv-Galerkin finite-element context. To get better dispersive accuracy, a proper upwinding coefficient is rigorously derived so that the difference between the numerical and exact wavenumbers for the first-order derivative terms is minimized. An element-by-element strategy is implemented in the currently chosen CGNR iterative solver for reducing the required memory. Prior to calculation of the solution from the finite-element matrix equation, which has been transformed to become symmetric and positive-definite through the normalization of the original unsymmetric and indefinite mixed Petrov-Galerkin finite-element equations, we precondition the CGNR equations. This procedure is essential to reduce the spectral radius of the normalized positive-definite finite-element equations. Both the Jacobi and polynomial preconditioners are investigated. The performance of three element-by-element iterative solvers are also assessed. It is concluded that the CGNR iterative solver applied together with the polynomial preconditioner is superior to the preconditioned GMRES and BICGSTAB iterative solvers.

FUNDING

Financial support from the National Science Council under Grants NSC99-2221-E-002-225-MY3 and TIMS Project 101R891002 is gratefully acknowledged.

REFERENCES

1. A. Brooks and T. J. R. Hughes, Streamline Upwind/Petrov-Galerkin Formulation for Convection Dominated Flow with Particular Emphasis on the Incompressible Navier-Stokes Equations, *Comput. Meth. Appl. Mech. Eng.*, vol. 32, pp. 199–259, 1982.
2. R. W. Freund, Transpose-Free Quasi-Minimal Residual Methods for Non-Hermitian Linear Systems, in G. H. Golub, A. Greenbaum, and M. Luskin (eds.), *Recent Advances in Iterative Methods*, IMA Vol. Math. Appl., vol. 60, pp. 69–94, 1994.
3. V. Simoncini and D. B. Szyld, Recent Computational Developments in Krylov Subspace Method for Linear Systems, *Numer. Linear Algebr.*, vol. 14, pp. 1–59, 2007.
4. T. J. R. Hughes, I. Levit, and J. M. Winget, An Element-by-Element Solution Algorithm for Problems of Structural and Solid Mechanics, *Comput. Meth. Appl. Mech. Eng.*, vol. 36, pp. 241–254, 1983.
5. R. Codina, E. Oñate, and M. Cervera, The Intrinsic Time for the Streamline Upwind/Petrov-Galerkin Formulation Using the Quadratic Elements, *Comput. Meth. Appl. Mech. Eng.*, vol. 94, pp. 239–262, 1992.

6. C. K. W. Tam and J. C. Webb, Dispersion-Relation-Preserving Finite Difference Schemes for Computational Acoustics, *J. Comput. Phys.*, vol. 107, pp. 262–281, 1993.
7. Y. Saad and M. H. Schultz, GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems, *SIAM J. Sci. Comput.*, vol. 7, pp. 856–869, 1986.
8. H. A. Van der Vorst, Bi-CGSTAB: A Fast and Smoothly Converging Variants of Bi-CG for the Solution of Non-Symmetric Linear Systems, *SIAM J. Sci. Comput.*, vol. 13, pp. 631–644, 1992.
9. P. Sonneveld, CGS: A Fast Lanczos-Type Solver for Nonsymmetric Linear Systems, *SIAM J. Sci. Comput.*, vol. 10, pp. 36–52, 1989.
10. R. W. Freund and M. Hochbruck, On the Use of Two QMR Algorithms for Solving Singular Systems and Applications in Markov Chain Modeling, *Numer. Linear Algebr.*, vol. 1, pp. 403–420, 1994.
11. R. W. Freund and N. M. Nachtigal, OMR: A Quasi-Minimal Residual Methods for Non-Hermitian Linear Systems, *Numer. Math.*, vol. 60, pp. 315–339, 1991.
12. C. Brezinski, M. D. Zaglia, and H. Sadok, Breakdowns in the Implementation of the Lanczos Method for Solving Linear Systems, *Comput. Math. Appl.*, vol. 33, pp. 31–44, 1997.
13. M. H. Gutknecht, Variants of BICGSTAB for Matrices with Complex Specturm, *SIAM J. Sci. Comput.*, vol. 14, pp. 1020–1033, 1993.
14. G. L. G. Sleijpen and D. R. Fokkema, BICGSTAB (*l*) for Linear Equations Involving Unsymmetric Matrices with Complex Spectrum, *Electron. Trans. Numer. Anal.*, vol. 1, pp. 11–32, 1993.
15. M. R. Hestenes and E. Stiefel, Methods of Conjugate Gradients for Solving Linear Systems, *J. Res. Natl. Inst. Stan.*, vol. 49, pp. 409–436, 1952.
16. P. Concus, G. H. Golub, and D. P. O'Leary, A Generalized Conjugate Gradient Method for the Numerical Solutions of the Elliptic Partial Differential Equations, in J. R. Bunch and D. J. Rose (eds.), *Sparse Matrix Computations*, Academic Press, New York, 1976.
17. M. A. Ajiz and A. Jennings, A Robust Incomplete Cholesky Conjugate Gradient Algorithm, *Int. J. Numer. Meth. Eng.*, vol. 20, pp. 949–966, 1984.
18. Y. Saad, Practical Use of Polynomial Preconditioning for the Conjugate Gradient Method, *SIAM J. Sci. Comput.*, vol. 6, pp. 865–881, 1985.
19. V. E. Bulgakov and G. Kuhn, High-Performance Multi-Level Iterative Aggregation Solver for Large Finite-Element Structure Analysis Problems, *Int. J. Numer. Meth. Eng.*, vol. 38, pp. 3529–3544, 1995.
20. J. M. Winget and T. J. R. Hughes, Solution Algorithm for Non-Linear Transient Heat Conduction Analysis Employing Element-by-Element Iterative Strategies, *Comput. Meth. Appl. Mech. Eng.*, vol. 52, pp. 811–815, 1985.
21. B. Nour-Omid and B. N. Parlett, Element Preconditioning Using Splitting Technique, *SIAM J. Sci. Stat. Comput.*, vol. 6, pp. 761–770, 1985.
22. F. H. Lee, K. K. Phoon, K. C. Lim, and S. H. Chan, Performance of Jacobi Precondtioning in Krylov Subspace Solutiom of Finite Element Equations, *Int. J. Numer. Anal. Meth.*, vol. 26, pp. 341–372, 2002.
23. R. W. Freund, G. H. Golub, and N. M. Nacgtigal, Recent Advances in Lanczos-Based Iterative Methods for Nonsymmetic Linear Systems, in M. Y. Hussaini, A. Kumar, and M. D. Salas, (eds.), *Algorithm Trends in Computational Fluid Dynamics*, Springer-Verlag Press, Berlin, 1993.
24. C. C. Fang and T. W. H. Sheu, Two Element-by-Element Iterative Solutions for Shallow Water Equations, *SIAM J. Sci. Comput.*, vol. 22, pp. 2075–2092, 2001.
25. H. Ding, C. Shu, K. S. Yeo, and D. Xu, Numerical Computaion of Three-Dimensional Incompressible Viscous Flows in the Primitive Variable Form by Local Multiquadratic Differential Quadrature Method, *Comput. Meth. Appl. Mech. Eng.*, vol. 195, pp. 516–533, 2006.