Three-Dimensional Incompressible Navier-Stokes Solver Using Lower-Upper Symmetric-Gauss-Seidel Algorithm

Seokkwan Yoon* and Dochan Kwak†
NASA Ames Research Center, Moffett Field, California

Abstract

A NUMERICAL method based on the pseudocompressibility concept is developed for solving the three-dimensional incompressible Navier-Stokes equations using the lower-upper symmetric-Gauss-Seidel implicit scheme. Very high efficiency is achieved in a new flow solver, INS3D-LU code, by accomplishing the complete vectorizability of the algorithm on oblique planes of sweep in three dimensions.

Contents

A difficulty arises in solving the incompressible Navier-Stokes equations because the pressure and velocity fields are not directly coupled due to the lack of a pressure term in the continuity equation. Instead, the continuity equation imposes a divergence-free constraint on the velocity field. The lack of a time-derivative term in the continuity equation limits the straightforward application of time-marching numerical methods. One way to avoid this difficulty is the use of a streamfunction-vorticity formulation whose extension to three dimensions is not straightforward. The method using the Poisson equation for pressure and the fractional-step method, which have been used widely for three-dimensional flows, tend to be computationally expensive. An alternative way is to use the concept of pseudocompressibility.[1] Then the efficiency of the pseudocompressibility method depends on a time-marching scheme to integrate the resulting system of partial differential equations. Recently, an efficient implicit algorithm was derived for the compressible Euler and Navier-Stokes equations by Yoon and Jameson.[2] The lower-upper symmetric-Gauss-Seidel (LU-SGS) implicit method proved useful for hypersonic flows[3-5] as well as for transonic flows. In this paper, the LU-SGS scheme is developed for the pseudocompressibility formulation of the three-dimensional incompressible Navier-Stokes equations. The present algorithm offers additional advantages when solving the incompressible flow equations with fully implicit treatment of source terms.[6]

Let t be time; p, u, v, and w the pressure and velocity components in Cartesian coordinates (x,y,z); $\hat{Q}$ the vector of primitive variables; $\hat{E}$, $\hat{F}$, and $\hat{G}$ convective flux vectors; and $\hat{E}_v$, $\hat{F}_v$, and $\hat{G}_v$ the flux vectors for the viscous terms. Introduction of the pseudocompressibility after the coordinate transformation gives the incompressible Navier-Stokes equations for three-dimensional flow in generalized curvilinear coordinates (ξ,η,ζ)

$$\partial_t\hat{Q} + \partial_\xi(\hat{E}-\hat{E}_v) + \partial_\eta(\hat{F}-\hat{F}_v) + \partial_\zeta(\hat{G}-\hat{G}_v) = 0 \qquad (1)$$

where

$$\hat{Q} = h\begin{bmatrix} p \\ u \\ v \\ w \end{bmatrix}, \qquad \hat{E} = h\begin{bmatrix} \beta U \\ Uu + \xi_x p \\ Uv + \xi_y p \\ Uw + \xi_z p \end{bmatrix}$$

$$\hat{F} = h\begin{bmatrix} \beta V \\ Vu + \eta_x p \\ Vv + \eta_y p \\ Vw + \eta_z p \end{bmatrix}, \qquad \hat{G} = h\begin{bmatrix} \beta W \\ Wu + \zeta_x p \\ Wv + \zeta_y p \\ Ww + \zeta_z p \end{bmatrix} \qquad (2)$$

and where β is the pseudocompressibility parameter, U, V, and W are the contravariant velocity components, and the cell volume h is the determinant of the inverse of transformation Jacobian matrix. An unfactored implicit scheme can be obtained from a nonlinear implicit scheme by linearizing the flux vectors about the previous time step and dropping terms of the second and higher order:

$$\left[I + \alpha\Delta t(D_\xi\hat{A} + D_\eta\hat{B} + D_\zeta\hat{C})\right]\delta\hat{Q}$$
$$= -\Delta t\left[D_\xi(\hat{E}-\hat{E}_v) + D_\eta(\hat{F}-\hat{F}_v) + D_\zeta(\hat{G}-\hat{G}_v)\right] \qquad (3)$$

where I is the identity matrix, $\delta\hat{Q}$ is the correction $\hat{Q}^{n+1}-\hat{Q}^n$, where n denotes the time level, D_ξ, D_η, and D_ζ are difference operators that approximate ∂_ξ, ∂_η, and ∂_ζ, and $\hat{A}$, $\hat{B}$, and $\hat{C}$ are the Jacobian matrices of the convective flux vectors:

$$\hat{A} = \begin{bmatrix} 0 & \xi_x\beta & \xi_y\beta & \xi_z\beta \\ \xi_x & U+\xi_x u & \xi_y u & \xi_z u \\ \xi_y & \xi_x v & U+\xi_y v & \xi_z v \\ \xi_z & \xi_x w & \xi_y w & U+\xi_z w \end{bmatrix} \qquad (4)$$

Matrices $\hat{B}$ and $\hat{C}$ are similarly derived. For $\alpha = ½$, the scheme is second-order accurate in time. For other values of α, the time accuracy drops to first order.

Direct inversion of a large block banded matrix of the unfactored scheme Eq. (3) is impractical in three dimensions because of rapid increase of operation count with the number of mesh points and large storage requirement. To alleviate this difficulty, much research work has been focused on indirect methods. MacCormack[7] introduced back-and-forth symmetric sweeps into the line Gauss-Seidel relaxation method. The line symmetric-Gauss-Seidel (SGS) relaxation method achieved fast convergence rates for certain problems by taking large time steps. However, the line Gauss-Seidel method is not vectorizable because of its recursive nature, resulting in a large CPU time per iteration. Although the situation can be relieved somewhat by storing all the Jacobian matrices in the domain at the expense of large storage requirements, the structure of algorithm that does not fit into the architecture of current and

Received May 22, 1989; synoptic received March 22, 1990. Full paper available from National Technical Information Service, Springfield, VA 22151, by title, at the standard price (available on request).
Copyright © 1990 by the American Institute of Aeronautics and Astronautics, Inc. No copyright is asserted in the United States under Title 17, U.S. Code. The U.S. Government has a royalty-free license to exercise all rights under the copyright claimed herein for Governmental purposes. All other rights are reserved by the copyright owner.

*Senior Scientist, MCAT Institute. Member AIAA.
†Research Scientist. Associate Fellow AIAA.

future supercomputers let many researchers stay with the approximate factorization methods. One of the popular approximate factorization methods is the Alternating Direction Implicit (ADI) scheme whose large factorization error further limits the time step, which is already restricted by the stability. It has been shown that the factorization error can be reduced by LU factorization.[8] The LU-SGS implicit factorization scheme can be derived by combining the advantages of LU factorization and SGS relaxation. The scheme resembles the LU factorization scheme. However, diagonal preconditioning increases the robustness, while scalar diagonal inversion is possible without using a diagonalization procedure. The scheme is also similar to the line SGS relaxation scheme with single subiteration. However, no additional relaxation or factorization is required on planes of sweep. Moreover, both central and upwind difference schemes can be used without losing the diagonal dominance. The LU-SGS scheme can be written as

$$LD^{-1}U\delta\hat{Q} = -\Delta t R \tag{5}$$

where

$$L = I + \alpha\Delta t(D_\xi^- \hat{A}^+ + D_\eta^- \hat{B}^+ + D_\zeta^- \hat{C}^+ - \hat{A}^- - \hat{B}^- - \hat{C}^-)$$

$$D = I + \alpha\Delta t(\hat{A}^+ - \hat{A}^- + \hat{B}^+ - \hat{B}^- + \hat{C}^+ - \hat{C}^-)$$

$$U = I + \alpha\Delta t(D_\xi^+ \hat{A}^- + D_\eta^+ \hat{B}^- + D_\zeta^+ \hat{C}^- + \hat{A}^+ + \hat{B}^+ + \hat{C}^+)$$

$$R = D_\xi(\hat{E} - \hat{E}_v) + D_\eta(\hat{F} - \hat{F}_v) + D_\zeta(\hat{G} - \hat{G}_v) \tag{6}$$

where D_ξ^- and D_ξ^+ denote backward and forward difference operators. In the framework of the LU-SGS algorithm, a variety of schemes can be developed by different choices of numerical dissipation models and Jacobian matrices of the flux vectors.[3] For example,

$$\hat{A}^\pm = \hat{T}_\xi \Lambda_\xi^\pm \hat{T}_\xi^{-1} \tag{7}$$

where $\hat{T}_\xi$ and $\hat{T}_\xi^{-1}$ are similarity transformation matrices of eigenvectors. Another possibility is to construct the Jacobian matrices of the flux vectors approximately to yield diagonal dominance:

$$\hat{A}^\pm = ½\left[\hat{A} \pm \rho(\hat{A})I\right] \tag{8}$$

and

$$\rho(\hat{A}) = \kappa \max\left[|\lambda(\hat{A})|\right] \tag{9}$$

where $\lambda(\hat{A})$ represent eigenvalues of Jacobian matrix $\hat{A}$ and κ is a constant that is greater than or equal to 1. The diagonal matrix of eigenvalues is

$$\hat{\Lambda}(\hat{A}) = \begin{bmatrix} U & 0 & 0 & 0 \\ 0 & U & 0 & 0 \\ 0 & 0 & U + C_\xi & 0 \\ 0 & 0 & 0 & U - C_\xi \end{bmatrix} \tag{10}$$

where C_ξ is the pseudospeed of sound:

$$C_\xi = \sqrt{U^2 + \beta(\xi_x^2 + \xi_y^2 + \xi_z^2)} \tag{11}$$

It is interesting to note that the need for block inversions can be eliminated if we use approximate Jacobian matrices of Eq. (8). The use of true Jacobian matrices of Eq. (7), which might lead to a faster convergence rate, requires block diagonal inversions and hence doubles the computational work per iteration. Another interesting feature of the present algorithm is that the scheme is completely vectorizable on $i + j + k =$ const oblique planes of sweep. The plane is illustrated in Fig. 1. To achieve this potential, the INS3D-LU code is written using two-dimensional arrays in three dimensions, that is,

Table 1 Performance on Cray YMP (CPU seconds per grid point per iteration)

Code	Algorithm	CPU, μs	MFLOPS	Note
INS3D-LU	LU-SGS	7	150	
INS3D	ADI	30	55	diagonal
INS3D	ADI	81	35	block
INS3D-UP	Line GS	90	55	

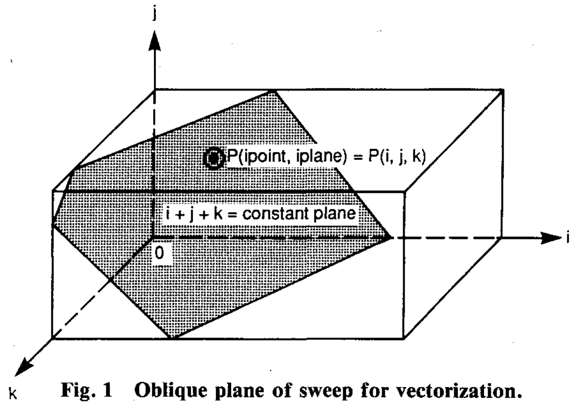

Fig. 1 Oblique plane of sweep for vectorization.

$$\hat{Q}(ipoint, iplane) = \hat{Q}(i,j,k) \tag{12}$$

where *i plane* is the serial number of plane of sweep, and *i point* is the address on that plane. The present algorithm may also be amenable to massively parallel processing. Complete vectorizability of the algorithm is demonstrated by the computing times per grid point per iteration of the code on single processor of Cray YMP computer. CPU seconds as well as MFLOPS are compared to existing incompressible Navier-Stokes codes in Table 1. As the table shows, the code needs only 7 μs for the full Navier-Stokes equations in three-dimensional generalized coordinates. Excellent agreement between numerical solutions and the experimental data for a square duct with a 90-deg bend was observed.[6] The robustness of the new method and the code is demonstrated with the application to the liquid-oxygen turbopump inducer flow of the Space Shuttle main engine.[6]

References

[1]Chorin, A. J., "A Numerical Method for Solving Incompressible Viscous Flow Problems," *Journal of Computational Physics,* Vol. 2, 1967, pp. 12–26.

[2]Yoon, S., and Jameson, A., "Lower-Upper Symmetric-Gauss-Seidel Method for the Euler and Navier-Stokes Equations," *AIAA Journal,* Vol. 26, Sept. 1988, pp. 1025–1026.

[3]Yoon, S., and Kwak, D., "Artificial Dissipation Models for Hypersonic Internal Flow," AIAA Paper 88-3277, July 1988.

[4]Park, C., and Yoon, S., "A Fully-Coupled Implicit Method for Thermo-Chemical Nonequilibrium Air at Sub-Orbital Flight Speeds," AIAA Paper 89-1974, June 1989.

[5]Shuen, J. S., and Yoon, S., "Numerical Study of Chemically Reacting Flows Using an LU-SSOR Scheme," *AIAA Journal,* Vol. 27, Dec. 1989, pp. 1752–1760.

[6]Yoon, S., Kwak, D., and Chang, L., "LU-SGS Implicit Algorithm for Three-Dimensional Incompressible Navier-Stokes Equations with Source Term," AIAA Paper 89-1964-CP, June 1989.

[7]MacCormack, R. W., "Current Status of Numerical Solutions of the Navier-Stokes Equations," AIAA Paper 85-0032, Jan. 1985.

[8]Jameson, A., and Yoon, S., "Lower-Upper Implicit Schemes with Multiple Grids for the Euler Equations," *AIAA Journal,* Vol. 25, July 1987, pp. 929–935.