3D computation of unsteady flow past a sphere with a parallel finite element method

V. Kalro*, T. Tezduyar

Abstract

We present parallel computation of 3D, unsteady, incompressible flow past a sphere. The Navier-Stokes equations of incompressible flows are solved using a stabilized finite element formulation. Equal-order interpolation functions are used for velocity and pressure. The second-order accurate time-marching within the solution process is carried out in an implicit fashion. The coupled, nonlinear equations generated at each time step are solved using an element-vector-based iteration technique. The computed value of the primary frequency associated with vortex shedding is in close agreement with experimental measurements. The computation was performed on the Thinking Machines CM-5.

1. Introduction

With the emergence of high-speed multiprocessor architectures, and concurrent advances in high performance computing (HPC) algorithms, flow simulation is being increasingly utilized to study a wide range of problems in engineering and applied sciences. These include aerodynamic vehicle design, weather prediction, oceanography, chemical reactions and combustion phenomena, particle laden flows and many more. Flow simulation has come a long way from its nascent stages where it was merely used for qualitative predictions. Today, on the other hand, for example, the aerospace industry has adopted a two-pronged approach, where experimental investigations and flow simulation go hand-in-hand in tackling the challenging issues arising in the design of modern aircraft.

The programming styles on parallel architectures can be broadly classified into Single-Instruction-Multiple-Data (SIMD) (see e.g. [1–5]) and Multiple Instruction Multiple Data (MIMD) (see e.g. [6,7]) paradigms. While compiler directives enable data-parallelism in SIMD implementations and simplify the parallelization procedures, the use of message-passing libraries in MIMD models gives the user an explicit handle on inter-processor communications, sometimes resulting in more efficient implementations.

The computations presented here are based on a stabilized finite element formulation [8,9], which suppresses the numerical oscillations in the flow field. Typically, the causes of such oscillations are the flow having a high Reynolds number, presence of sharp boundary layers, and using certain combinations of interpolants for velocity and pressure variables.

The finite element formulation gives rise to a system of coupled, nonlinear equations. Due to the prohibitive size of such systems for large-scale simulations, the solution is obtained by iteration techniques which have high parallel efficiency. We use the GMRES [10] update technique, and compute the residuals based on matrix-free (element-vector-based) techniques.
Advanced computational methods and hardware now give us the capability for large-scale simulations of modestly high Reynolds number flows around complex shapes. Since it is possible to resolve the fine-scale flow features in detail not realized previously, flow simulation is also being used to study fundamental flow phenomena around simple geometries. Large-scale computations around circular cylinders were reported by the authors in an earlier article [11]. Recently, a novel capability for 3D simulation of a hundred particles in a fluid was reported by Johnson and Tezduyar [12]. In this paper we present 3D computation of unsteady incompressible flow around a sphere. Such flows are encountered in many engineering applications and natural phenomena. These include flow through porous media, sedimentation, interaction between droplets in the atmosphere, and several chemical processes. In our computations, we are able to capture the intricate vortex shedding mechanisms in the wake of the sphere.

The Navier–Stokes equations governing incompressible flows are reviewed in Section 2. The stabilized formulations, formation of discrete equation systems, and matrix-free strategies are outlined in Section 3. The data-parallel implementation on the Thinking Machines CM-5 is briefly described in Section 4. The numerical simulations are presented in Section 5. Closing remarks are put forth in Section 6.

2. Governing equations

Let \( \Omega \subset \mathbb{R}^{n_d} \) be the spatial domain of interest bounded by boundary \( \Gamma \). Here, \( n_{sd} \) is the number of spatial dimensions. The Navier–Stokes equations governing incompressible flows are

\[
\begin{align*}
\rho \left( \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} - f \right) - \nabla \cdot \boldsymbol{\sigma} &= 0 \quad \text{on } \Omega, \\
\nabla \cdot \mathbf{u} &= 0 \quad \text{on } \Omega.
\end{align*}
\]

Here, \( \rho, \mathbf{u}, f \) and \( \boldsymbol{\sigma} \) are the density, velocity, body force and the stress tensor, respectively. The stress tensor is written as the sum of its isotropic and deviatoric parts:

\[
\boldsymbol{\sigma} = -\rho \mathbf{I} + \mathbf{T},
\]

where

\[
\mathbf{T} = 2\mu \varepsilon(\mathbf{u}),
\]

\[
\varepsilon(\mathbf{u}) = \frac{1}{2} (\nabla \mathbf{u} + (\nabla \mathbf{u})^T).
\]

Here, \( \rho \) and \( \mu \) are the pressure and viscosity, and \( \mathbf{I} \) is the identity tensor. Both the Dirichlet- and Neumann-type boundary conditions are accounted for, represented as

\[
\begin{align*}
\mathbf{u} &= \mathbf{g} \quad \text{on } \Gamma_x, \\
\mathbf{n} \cdot \mathbf{\sigma} &= \mathbf{h} \quad \text{on } \Gamma_h.
\end{align*}
\]

Here, \( \Gamma_x \) and \( \Gamma_h \) are complementary subsets of the boundary \( \Gamma \), \( \mathbf{n} \) is the unit normal vector at the boundary, and \( \mathbf{g} \) and \( \mathbf{h} \) are given functions. The initial condition on the velocity is specified as

\[
\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0 \quad \text{on } \Omega,
\]

where \( \mathbf{u}_0 \) is divergence free.

3. Finite element methodology

3.1. Stabilized semi-discrete formulation

The domain \( \Omega \) is discretized into subdomains \( \Omega^e, e = 1, 2, \ldots, n_{el} \), where \( n_{el} \) is the number of elements. Based on this discretization, the finite element trial function spaces \( \mathcal{V}_u^e \) and \( \mathcal{V}_p^e \), and weighting function spaces \( \mathcal{V}_h^e \) and \( \mathcal{V}_h^p \) are defined for the velocity and pressure. The selected function spaces account for the Dirichlet
boundary conditions, and are subsets of \([H^{1b}(\Omega)]^n\) and \(H^{1b}(\Omega)\), where \(H^{1b}(\Omega)\) is the finite-dimensional function space over \(\Omega\). The stabilized finite element formulation of Eqs. (1)-(2) is written as follows: find \(u^h \in \mathcal{V}_u^h\) and \(p^h \in \mathcal{V}_p^h\), such that \(\forall \, w^h \in \mathcal{V}_w^h, \, q^h \in \mathcal{V}_q^h\)

\[
\int_{\Omega} w^h \cdot \left( \frac{\partial u^h}{\partial t} + u^h \cdot \nabla u^h - f \right) \, dt + \int_{\Gamma} \mathcal{E}(w^h) : \sigma(p^h, u^h) \, dl + \int_{\Omega} q^h \nabla \cdot u^h \, dt + \sum_{i=1}^{n} \int_{\Gamma} \left( \tau_{\text{SUPG}} \rho u^h \cdot \nabla w^h + \tau_{\psi_{PSG}} \rho \nabla q^h \right) \cdot [L(u, p) - f] \, dl \\
+ \sum_{i=1}^{n} \int_{\Gamma} \delta \nabla \cdot w^h \rho \nabla \cdot u^h \, dl \\
- \int_{\Gamma} w^h \cdot h^h \, dl
\]

where

\[
L(w, q) = \rho \left( \frac{\partial w^h}{\partial t} + u^h \cdot \nabla w^h \right) - \nabla \cdot \sigma(q^h, w^h).
\]

**REMARKS**

(1) The first three terms in Eq. (9), together with the right-hand side, constitute the standard Galerkin formulation of the problem.

(2) The first series of element-level integrals are the SUPG and PSPG stabilization terms added to the variational formulations, where, \(\tau_{\psi_{PSG}}\) in our computations is selected to be of the same form as \(\tau_{\text{SUPG}}\). The second series of element-level integrals provide numerical stability at high Reynolds numbers. This is a least-squares term based on the continuity equation. Details on the stabilization coefficients are discussed in [9].

(3) Since the stabilization terms are weighted residuals, they maintain consistency of the formulation. In Eq. (9), we use the same class of interpolation functions for \(u^h, w^h, p^h\) and \(q^h\). In this paper we report results from linear, 3D interpolation functions.

The implicit, second-order accurate trapezoidal rule is used for time-marching. At each time step, as we march from level \(n\) to level \(n+1\), we need to solve a nonlinear equation system:

\[
R(d_{n+1}) = 0,
\]

where \(d_{n+1}\) is the nodal values of the unknowns \((u, p)\) corresponding to level \(n+1\).

We solve Eq. (11) with Newton–Raphson iterations:

\[
J^t_{n+1} \, \Delta d_{n+1} = R(d_{n+1}^t),
\]

where \(d_{n+1}^t\) is the \(k\)th iteration value of \(d_{n+1}\), \(J^t_{n+1}\) is the approximate derivative of \(R\) with respect to \(d_{n+1}\), evaluated at \(d_{n+1}^k\), and \(\Delta d_{n+1}^t\) is the correction computed for \(d_{n+1}^t\). The derivative of \(R\) with respect to \(d_{n+1}^k\) is approximate in the sense that some of the parameters, such as the stabilization coefficients, are frozen. From numerical experiments it was observed that such freezing does not significantly affect the convergence of these iterations.

### 3.2. Element-vector-based iteration strategies

The prohibitive costs associated with direct solution techniques in large-scale simulations make it necessary to use iterative methods. In iterative techniques, we start with an approximate guess to the solution, and systematically search for updates to the solution. In our computations we use the GMRES [10] update technique. The essential component of this method is the formation of the Krylov vectors. Subsequent vectors are recursively generated by taking matrix–vector products of the matrix \(J\) and the current vector:

\[
v_{j+1} = J_{n+1}^t v_j,
\]

where \(v_j\) is the initial guess, and \(J_{n+1}^t\) is the transpose of the Jacobian matrix. The Krylov vectors are used to approximate the solution in the Krylov subspace, which is defined as the space spanned by the Krylov vectors. The GMRES algorithm aims to find the vector in the Krylov subspace that minimizes the residual of the linear system.
Various schemes for evaluating matrix-vector products in parallel have been proposed. These include element-matrix-based schemes, sparse-storage-based schemes and element-vector-based schemes, listed in decreasing order of memory requirements.

Element-matrix-based schemes [1] are straightforward and require storing the element-level components of $J_f^{k+1}$. The global-level matrix-vector product is computed by first localizing $u_j$ to the element-level, then computing element-level matrix-vector products, and finally assembling the global vector $u_{j+1}$.

Sparse-storage-based schemes [7] involve setting up sparse data-structures for partitions of elements. Each processor is assigned an element partition. The sparse, partition-level components of $J_f^{k+1}$ are computed and stored on the corresponding processor. Here, the computation of the matrix-vector products involves localizing $u_j$ to partition-level vectors, followed by sparse partition-level matrix-vector products, and finally assembling the global vector $u_{j+1}$.

In element-vector-based schemes, the global matrix is not stored at any level. The matrix-vector products are computed by using the following expression:

$$J_f^{k+1} u_j = \lim_{\varepsilon \to 0} \frac{R(d_{n+1}^{k+1}) - R(d_{n+1}^{k} + \varepsilon u_j)}{\varepsilon}.$$  \hspace{1cm} (14)

Techniques for choosing an optimum $\varepsilon$ are discussed by Johan [3]. The matrix-vector products can in some cases be evaluated analytically without using $\varepsilon$ (and this is how we do it in computations reported here), subjected to ‘freezing’ restrictions described earlier.

4. Parallel implementation

The methodology presented in the previous section is implemented on the Thinking Machines CM-5. In this section we summarize the basic components of the data-parallel implementation. For a detailed description, the reader is referred to earlier work [13]. The bulk of the computational cost is spent on element vector evaluations. This task is perfectly parallelizable and easily achieved by allocating equal-sized element clusters to each processor. However, the communication involved in constructing global vectors is directly proportional to the number of mesh nodes on the cluster boundaries. For this purpose, partitioning techniques [3,14] are used to form clusters which minimize the number of boundary nodes. Each processor is also assigned a set of global nodes; a majority of these are interior nodes of the corresponding element partition. The boundary nodes are shared amongst common partitions. In this way communication bottlenecks are averted, resulting in a scalable, load-balanced computation.

5. 3D flow past a sphere at $Re = 400$

The wake of a sphere exhibits many interesting flow structures. The flow is axisymmetric and attached at $Re < 20$. The flow separates at $Re \sim 24$ giving rise to a cylindrical vortex surface. This surface is unstable and involutes to form a vortex ring behind the sphere [15]. It was reported [16] that the axisymmetric ring configuration loses its stability in the range $120 < Re < 300$. According to Tanneda [17], the instability is initiated via low-frequency pulsations aft of the ring at $Re \sim 130$. At $Re \sim 300$, the backflow in the wake region just aft of the sphere (arising from the inductive effect of the vortex ring) causes the rings to detach partially and form vortex loops. Magarvey and Maclatchy [18] give a good description and visualization of the vortex formation. These vortex loops (hairpin vortices) are periodically shed with uniform strength and frequency. The Strouhal number corresponding to the shedding frequency is reported [15] to be in the range of 0.120–0.160. In experimental investigations, it was observed that the plane of shedding rotates (and in some cases oscillates intermittently) about the streamwise axis.

With the radius of the sphere taken as the unit length, the computational domain extends from 15 units
upstream to 30 units downstream. The mesh consists of 43,282 nodes and 258,569 elements (see Fig. 1), and results in 162,096 unknowns. The outer boundary has a circular cross-section with a diameter of 14 units.

The boundary conditions consist of uniform inflow velocity (set to 1.0 unit), zero-crossflow-velocity and zero-shear-stress at the lateral boundaries, a traction-free outflow boundary, and no slip on the sphere. The unstructured tetrahedral mesh was generated using the software developed by Johnson [19].

The time step was set to 0.1, the number of nonlinear iterations to 2, and the Krylov space size in GMRES iterations to 5, with no restarts. This simulation required ~3.65 seconds per time step on a 256-PN CM-5.

It is observed that hairpin vortices are shed periodically. The time-averaged drag coefficient is ~0.619 and compares well with the reported [20] experimental value. We observe that the drag and lift oscillate with the same frequency (see Fig. 2).

Visualization of the vorticity magnitude confirms the presence of the involuted vortex surface (see Fig. 3). A pair of counter-rotating vortex trails appear from the involution and fuse into a loop. These loops are shed periodically.

Since vortex tubes cannot terminate abruptly in the flow field (vorticity is a solenoidal quantity by definition), downstream of the sphere these loops are attached to each other. The computed Strouhal number based on the
Fig. 3. Flow past a sphere at Re = 400. Magnitude of the vorticity at different instants illuminates the mechanism through which vortex tubes develop, stretch and finally break away.

Fig. 4. Flow past a sphere at Re = 400. Time-histories of the drag (left) and side force (right) coefficients at advanced stages of the simulation. (Plots are shown for a time interval spanning 1200 units.)
Fig. 5. Flow past a sphere at Re = 400. Time-histories of the moment coefficients (x component on the left, y and z components on the right) at advanced stages of the simulation. (Plots are shown for a time interval spanning 1200 units.)

Fig. 6. Flow past a sphere at Re = 400. Images show the flow vectors in a cross-section perpendicular to the flow direction at x = 0.0, 0.6, 1.0, 1.2, 1.4 and 1.9. The sequence goes left to right, top to bottom.
vortex shedding frequency is \(-0.131\). Visualization of the vorticity magnitude reveals that the plane in which these vortices are shed undergoes rotation.

Without a sufficiently long-duration computation, it may appear that the rotation does not follow a regular pattern. However, with our computations being carried out over several shedding periods, we observe that the rotation is periodic (see Figs. 4 and 5). The Strouhal number corresponding to this frequency is \(-0.0067\). To our knowledge, a measurement of this frequency has not been previously reported.

Fig. 6 shows the flow vectors in the crossflow plane at various streamwise locations at some instant (the sphere is located at \(x = 0.0\)). At this instant, until about \(x = 0.8\), the flow field on this plane exhibits a nearly uniaxial symmetry. The instantaneous shedding plane is perpendicular to the axis of this symmetry. This uniaxial symmetry is lost beyond \(x = 0.8\), and the appearance of streamwise vortices at \(x = 1.9\) is seen. Fig. 7 shows the velocity vectors in 4 streamwise planes. The successive planes are 45° apart. Image 4 lies in the plane of shedding, and the wake structure is nearly symmetric in this plane. Also note the presence of the strong backflow regions which extends \(\sim 1.5\) diameters downstream of the sphere. The flow separates at an angle \(\sim 65°\) measured from the streamwise axis. Fig. 8 shows the instantaneous stream tubes in mutually perpendicular planes.

![Image of flow past a sphere at Re = 400.](image)

**Fig. 7.** Flow past a sphere at Re = 400. Images show the flow vectors in cross-sections parallel to the flow direction. The normals between successive frames are 45° apart.
6. Concluding remarks

With parallel 3D computations, we were able to capture the vortex-shedding mechanisms in the wake of a sphere. The primary frequency corresponding to vortex shedding, and the time averaged drag coefficient are in close agreement with experiment. Extended time-integration reveals that the plane of shedding undergoes periodic rotation about an axis aligned to the free-stream.
Acknowledgment

This research was sponsored by ARO under grant DAAH04-93-G-0514, by ARPA under NIST contract 60NANB2D1272, and by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAH04-95-2-0003/contract number DAAH04-95-C-0008, the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred. CRAY C90 time was provided in part by the University of Minnesota Supercomputer Institute.

References