

PARALLEL PROGRAM COMPLEX FOR UNSTEADY FLOW SIMULATION*

Eugene V. Shilnikov*

** Institute for Mathematical Modeling RAS
125047, Miusskaya sq. 4A, Moscow, Russia
Email: shiva@imamod.ru*

Key words: viscous gas flow, kinetic schemes, locally refined grids, parallel program complex, geometry parallelism principle, scalability.

Abstract. A parallel program complex for 3D viscous gas flow simulation is presented. This complex is based on explicit finite difference schemes, which are constructed as an approximation of conservation laws (control volume method) and oriented on use of locally refined grids. Special algorithm and utility for nested grid partitioning was created. The principle of program construction permits to introduce new types of boundary conditions and change as finite difference scheme as governing equation system. Introducing new face types and writing new subroutines for flux calculations may reach it. The scalability of program complex was investigated on 2D and 3D subsonic and supersonic problems. The calculations were held on different cluster type multiprocessor computer systems.

1. INTRODUCTION

Essentially unsteady and turbulent regimes of viscous gas flows have received increasing attention from researches, motivated in part from the importance of unsteadiness in industrial problems arising in turbomachinery and aeronautics. Unsteady flow phenomena which occur frequently behind relatively slender, bluff structures are of great practical interest. In the case of symmetric geometry at relatively small Reynolds numbers numerical simulation based on 2D unsteady Navier - Stokes equations is quite successful. At high Reynolds numbers, which are more relevant in practice, 3D stochastic turbulent fluctuations are superimposed on the quasi-periodic 2D unsteady motion. So, numerical simulation must be three-dimensional even for simple flow geometry. The numerical simulation of a detailed structure of unsteady viscous compressible 3D flows with high Reynolds numbers is possible only by use of high performance parallel computer systems. This demands the development of the specialized parallel software. This software must have a good scalability (with respect as to the number of processors as to the problem size), portability and robustness.

* This work was supported by Russian Foundation for Basic Research (grants No. 06-01-00187, 05-07-90230, 05-01-00510)

2. CHOICE OF NUMERICAL METHOD

Use of parallel computer systems with distributed memory architecture determines the choice of numerical method. The opinion is widely spread that we have to use only implicit schemes for viscous gas flow simulation because of their good stability properties. It is quite right for stationary or slow flows. In the case of essentially unsteady flow we have to receive detailed information about high frequency oscillations of gas dynamic parameters. This fact limits the time step acceptable by the accuracy requirements. For many interesting problems these limitations are very strong, although they may not ensure the stability of explicit scheme. On the other hand each time step of implicit schemes usually demands much more calculations. Besides, these schemes are difficult for parallel implementation from the viewpoint of efficiency. All these factors together may neutralize the advantages of implicit schemes. So for such problems the explicit difference schemes seem to be preferable because of their simplicity. Our program complex is based on explicit finite difference schemes, which are constructed as an approximation of conservation laws (control volume method). The explicit kinetically consistent finite difference (KCFD) schemes¹ were selected for realization. The variant of KCFD schemes named Corrected KCFD scheme may be considered as an approximation of Navier - Stokes equations with some additional terms. For 3D case in Cartesian coordinates it can be written in the following form

$$\frac{\hat{\rho} - \rho}{\Delta t} + (\rho u_j)_{x_j}^{\circ} = \left(\tau_j (\rho u_j^2 + p)_{x_j}^- \right)_{x_j} \quad (1)$$

$$\frac{\hat{\rho} u_i - \rho u_i}{\Delta t} + (\rho u_i u_j + p \delta_{ij})_{x_j}^{\circ} = \left((\mu u_i)_{x_j}^- (1 + \frac{\delta_{ij}}{3}) \right)_{x_j} + \left(\tau_j (\rho u_i u_j^2 + 3 p u_i \delta_{ij})_{x_j}^- \right)_{x_j} \quad (2)$$

$$\begin{aligned} \frac{\hat{E} - E}{\Delta t} + (u_j (E + p))_{x_j}^{\circ} &= \left(\frac{\mu}{2} \left(\mathbf{u}^2 + \frac{u_j^2}{3} \right)_{x_j}^- \right)_{x_j} + \frac{\gamma}{\gamma - 1} \left(\kappa \left(\frac{p}{\rho} \right)_{x_j}^- \right)_{x_j} + \\ &+ \left(\tau_j (u_j^2 (E + 2.5 p))_{x_j}^- \right)_{x_j} + \left(\tau_j \frac{p}{\rho(\gamma - 1)} p_{x_j}^- \right)_{x_j} \end{aligned} \quad (3)$$

Here $i, j = 1, 2, 3$, summation by index j is supposed, ρ – density, p – pressure, u_i – three velocity components, $E = \frac{p}{\rho(\gamma - 1)} + \frac{\mathbf{u}^2}{2}$ – total energy, $\mathbf{u}^2 = u_1^2 + u_2^2 + u_3^2$, γ – specific ratio, μ – viscosity, κ – conductivity, A_x, A_x^-, A_x^+ – central, backward and forward finite difference first order spatial derivatives, Δt – time step, \hat{A} – variable at the next time moment. All gas dynamic variables are defined in the cell centers. Additional dissipative terms in these equations contain factor $\tau_j = \Delta x_j / (2U)$, where Δx_j – spatial step along j coordinate and $U = \sqrt{\gamma p / \rho + \mathbf{u}^2}$.

KCFD schemes belong to the class of kinetic schemes. Nowadays the kinetic or Boltzmann schemes are very popular in computational fluid dynamics²⁻⁴. The kinetic schemes differ from the other algorithms primarily in the fact that the basis for their derivation is the discrete models for the one-particle distribution function.

Schematically, the process of deriving the kinetic schemes can be represented as the following chain: the Boltzmann equation – the discrete model for one-particle distribution function – the averaging of the discrete models with the collision vector components. As a result we obtain the discrete equations for gas dynamic parameters. Traditionally, another logical chain is used: the Boltzmann equation – the averaging of the Boltzmann equation with the collision vector components and the gas dynamics equations as a result – the discretization of gas dynamics equations. Compared with the traditional algorithms, the processes of averaging and discretization are interchanged in the case of kinetic schemes. The basic assumptions used for KCFD schemes construction are that one particle distribution function (and the macroscopic gas dynamic parameters too) have small variations on the distances compatible with the average free path length l and the distribution function has Maxwellian form just after molecular collisions. Proceeding from these assumptions we may suppose that one particle distribution function has a Maxwellian form constant on cells of size equal to the free path length. The dissipative terms of KCFD schemes are formed by two different "half-Maxwellian" distribution functions taken from both sides of a cell face. They can be interpreted as efficient numerical stabilizers which provide smoothness of the solution on the distance l . It is interesting to remark that analogous stabilizers were obtained independently in³. KCFD schemes have Courant-like stability condition ($\Delta t \sim \Delta x_{\min}$) giving the opportunity to use very fine meshes to study the fine flow structure. The successful experience in solving various gas dynamic problems by means of KCFD schemes showed that they describe viscous heat conducting flows as good as schemes for Navier - Stokes equations, where the latter are applicable.

Modeling some problems of mathematical physics demand high accuracy resolution of the solution particularities in small regions. Such situation may occur, for example, in detonation or combustion processes, solitons motion, flow around a body with shape singularities etc. This problem may be avoided by use of unstructured meshes, but the convenience and simplicity of difference schemes on regular grids enforced us to use multiblock grids where different subregions have their own grids. A variant of such approach, the use of nested (or locally refined) grids is chosen. The main idea is to divide one or more cells of regular (maybe rough) grid into some amount of small cells. These cells are rectangular too. If these new cells are not sufficiently small, we may repeat the refining procedure and divide some of them into more small cells and so on. We use explicit difference schemes, which are constructed as an approximation of conservation laws. So, the main problem is approximation of fluxes between cells of different sizes. Construction of this approximation and parallel realization of explicit finite difference schemes for the numerical solution of gas dynamic problems on nested grids were discussed in⁵.

3. PARALLEL IMPLEMENTATION

Parallel realization is based on geometry parallelism principle. Each processor makes calculations in its own separate subdomain. It has to exchange data with processors working in adjacent subregions. Only information from boundary cells of each subregion is to be sent, and the exchange is local. In the case when the difference scheme is written in the form of conservation laws, the approximation of gas dynamic equations comes to approximation of conservative variables (density ρ , momentum ρU and total energy E) fluxes through cell faces. In order to reach the algorithm homogeneity the boundary conditions of different types (no slip, symmetry,

impermeability, inlet, outlet conditions etc.) are also written as fluxes of conservative variables through region bound. Each cell face is supplied by an attribute indicating its type: inner face, various boundary faces, face between cells of different size (result of local mesh refining), ghost face i.e. face between cells laying outside the computational region. This attribute determines which subroutine must calculate fluxes through the face. The face attributes as well as description of problem geometry and grid information are contained in a special text file, which is prepared by sequential preprocessing utility.

Another utility divides 3D computational region with rectangular bounds (in i-j-k space) into required number of rectangular subdomains according to multistep algorithm described in⁵. It is based on the utility contained in the previous version of our program package⁶. It divides 3D box-shaped region into required number of rectangular subdomains with approximately equal volumes. This utility was modified in order to take into account local grid refinement (which may be rather deep). The computational region is expanded up to parallelepiped. Each cell of original regular mesh is ascribed a weight 1. Additional cells get zero weights and their faces get "ghost" attribute. Zero weights are also ascribed to ghost cells which are inside computational region (cells in the solid body, for example). If a cell was divided into N small cells during mesh refining, it is ascribed a weight N . Now we may calculate a number of cells which must be in each subregion. It is equal to sum of all weights divided by number of processors. After it the computational region is divided into needed number of subdomains with approximately equal total weights. If the mesh refinement is too deep, some of original cells weights may be too large for one subdomain. Meeting such cell we treat it separately using the same algorithm. After allotting a needed number of small cells to a subdomain we subtract this number from the weight of treated large cell. This procedure is repeated until the residual weight becomes less than number needed for one processor.

Note that some subdomains may be overlapping. In this situation their common faces get "ghost" attribute in one of the regions. The permissibility of overlapping subregions is useful for achieving equal number of cells when all domains have box shape. As a result this utility creates a text file describing 3D subregions in terms of grid node numbers, list of neighbors for each subregion and information needed for organizing of inter processor communications.

According to this the main computational module of parallel program is uniform for all processors. Each processor calculates fluxes through all faces in its subregion. Having equal number of faces in each subdomain the homogeneity of algorithm automatically provides load balancing of processors. For regular grid equal number of cells provides equal number of faces except faces between cells of different size. The number of such faces, which bound the zones of local mesh refining, is not very large. Some imbalance is also possible because different face types demand different amount of calculations. However, large number of cells in subdomains leads to leveling these differences. At last, small differences among weights of subregions are possible. It is unavoidable because the result of division of total weight by number of processors is not always integer. These differences do not affect efficiency dramatically if the maximum weight is close to average one.

Note, that such program construction permits to introduce new types of boundary conditions and change as finite difference scheme as governing equation system (changing the coordinate system for example).

4. PROGRAM PACKAGE TESTING

A program complex was tested on a problem of supersonic ($M_\infty = 2$) and subsonic ($M_\infty = 0.135$) viscous compressible gas flow around square cylinder at $Re = 20000$. The inlet of the computational region was located at $x/D = -5$ and outlet at $x/D = 15$, where $D = 1$ is cylinder diameter, the origin of the coordinate system was in the center of the cylinder front surface. The length of cylinder was equal to $4D$. The height and width of the computational region were equal to $20D$. Locally refining computational grid was used. The original coarse grid with maximum cell size equal to 0.2 was refined near cylinder walls so that the minimum cell size was 0.001. Total amount of grid cells was about 8000000. The no slip conditions were posed on cylinder walls. The far field boundary conditions based on splitting of the convective fluxes by Steger – Warming⁷ were used.

The calculated flow pictures are typical for the flow types under consideration and are similar to experimental data. Supersonic flow is stationary in our case. The calculated pressure distribution in symmetry plane $z=0$ for this flow is presented on Fig. 1. The subsonic flow is much more complicated and interesting. The symmetry is broken due to vortex shedding from four cylinder back edges and the flow becomes quasi-periodic. The stream traces near the cylinder in the same symmetry plane for different flow phases are shown in Fig. 2.

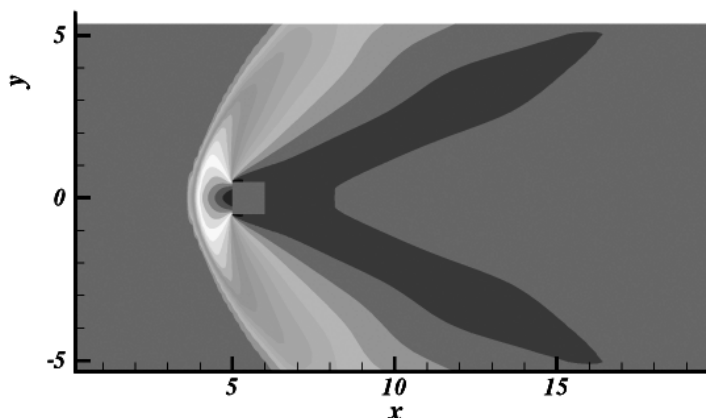


Figure 1. Pressure distribution for supersonic flow

The comparison of our results for subsonic problem with experimental^{8,9} and calculated¹⁰ (with different turbulence models in 2D problem formulation) results were held. Integral force characteristics such as mean value of drag coefficient ($\overline{c_D}$), amplitude of middle range oscillations of lift coefficient (c'_L) and Strouhal number (Sh) corresponding to them are summarized in Table 1. You can see a good agreement.

	$\overline{c_D}$	c'_L	Sh
Experimental results ⁹	2.05 – 2.19	—	0.135 – 0.139
Calculated results ¹⁰	1.56 – 2.11	0.30 – 1.18	0.129 – 0.146
Our results	2.09	0.37	0.137

Table 1. Force coefficients and Strouhal numbers

It is necessary to note, that our results were obtained without introducing any turbulence model.

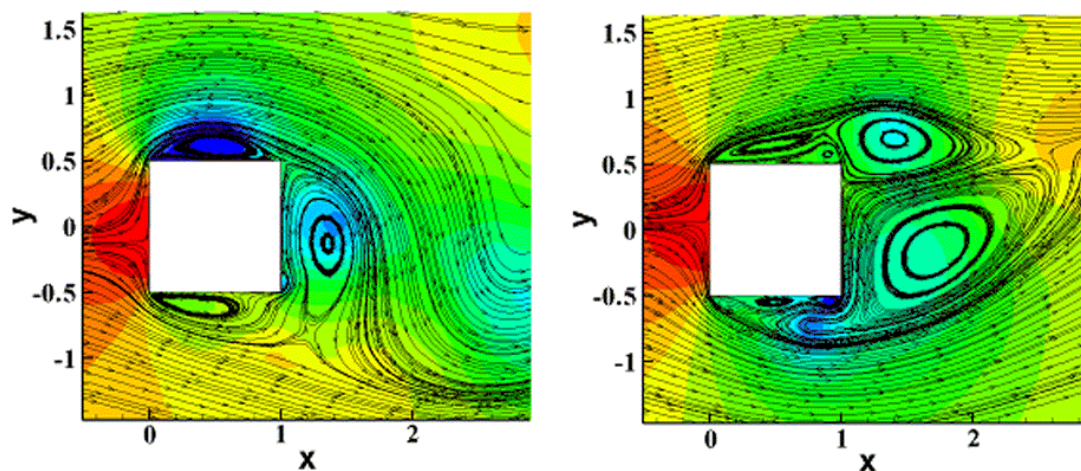


Figure 2. Stream traces in $z = 0$ section for subsonic flow

5. EFFICIENCY INVESTIGATION

The scalability of program complex was investigated on different cluster type multiprocessor computer systems (768-processor MCS-1000M computer system equipped with 667MHz 64-bit 21164 EV67 Alpha processors and 906-processor MCS-15000 computer system equipped with 2.2GHz PPC970FX processors). The parallelization efficiency was measured for different number of processors. The results can be found in Table 2.

Number of processors	1	2	10	40	160	320	600
MVS – 1000M	—	100	97	92.8	79.5	70.1	61.7
MVS – 15000	100	98.1	96.2	92	79.2	67.4	57.3

Table 2. Efficiency (%) dependence on the number of processors

Because of lack of memory such a large problem can't be solved on one processor of MCS-1000M system. That's why the efficiency for this system was computed with respect to the calculations held on 2 processors.

The scaling with respect to the problem size was also inspected. Our grid was doubled in each direction, so total number of cells became 8 times as large as previous and achieved 6×10^7 . For the number of processors $N < 100$, when the problem is sufficiently large (more than 100000 cells per processor), the efficiency practically doesn't depend on the problem size. Further increasing of N leads to the greater efficiency for greater problem. For example for $N = 600$ we have 74% efficiency for "large" problem in contrast with 57.3% for "small" one. These results are quite natural for the explicit finite difference schemes.

10. REFERENCES

- [1] O.C. Zienkiewicz and R.L. Taylor, *The finite element method*, McGraw Hill, Vol. I., 1989, Vol. II, (1991).
- [2] S. Idelsohn and E. Oñate, “Finite element and finite volumes. Two good friends”, *Int. J. Num. Meth. Engng*, 37, 3323-3341 (1994).
- [1] B.N. Chetverushkin. “On improvement of gas flow description via kinetically-consistent difference schemes”. In: B.N. Chetverushkin et al (Eds): *Experimental modeling and computation in flow, turbulence and combustion*, Wiley, Vol. 2, 27 – 37, (1997).
- [2] B. Perthame. “The kinetic approach to the system of conservation laws. Recent advances in partial differential equations”, *Res. Appl. Math.* Masson, Paris, 30, (1992).
- [3] E. Oñate and M. Manzan. *Stabilization techniques for finite element analysis for convective-diffusion problem*. Publication CIMNE, 183, (2000).
- [4] S. Succi. *The lattice Boltzmann equations for fluid dynamics and beyond*. Oxford, Clarendon press. (2001).
- [5] E.V. Shilnikov. “Viscous gas flow simulation on nested grids using multiprocessor computer systems”. In: *Proceedings of Parallel Computational Fluid Dynamics Conference (Moscow, Russia, 2003)*, 110 – 115, Elsevier Science BV, (2004).
- [6] E.V. Shilnikov, M.A. Shoomkov. “Parallel Program Package for 3D Unsteady Flows Simulation”. In: *Parallel Computing. Advances and current Issues. Proceedings of international conference ParCo2001*. G.R. Joubert et al. (eds.), Imperial College Press, London, UK, 238 – 245, (2002).
- [7] C. Hirsh: *Numerical computation of internal and external flows*, Vol. 1: Fundamentals of numerical discretization. John Wiley & Sons, A Wiley Interscience Publication,. (1988).
- [8] D.A. Lin and W. Rody. “The flapping shear layer formed by flow separation from the forward corner of a square cylinder”, *J. Fluid Mech.*, Vol. 267, 353 – 376, (1994).
- [9] D.A. Lin, S. Einav, W. Rody, J.-H. Park. “A laser-Doppler velocimetry study of ensemble-averaged characteristics of the turbulent near wake of a square cylinder”, *J. Fluid Mech.*, Vol. 304, 285 – 319, (1995).
- [10] G. Bosch, W.Rodi. “Simulation of vortex shedding past a square cylinder with different turbulence models”, *Int. J. Numer. Meth. Fluids*, Vol. 28, 601 – 616, (1998).