Parallel Program Complex "Express-3D" for 3D Flows Simulation on Hybrid Computer Systems

Alexander A. Davydov and Evgeny V. Shilnikov
Keldysh Institute of Applied Mathematics RAS
Moscow, Russia
e-mail: alexander.a.davydov@gmail.com, shiva@imamod.ru

***Abstract*—The paper presents a program complex for solving computational fluid dynamics problems, oriented on heterogeneous computer systems. Based on finite volumes method, an explicit difference scheme is constructed for Quasi Gas Dynamic equations system in 3D formulation on arbitrary hexagonal non-orthogonal structured index grid. The use of multi block grids is proposed. To improve the stability condition, the flux relaxation approach is used. The algorithm efficiency was verified on a set of test problems with wide variety of flow types. The speed-up for different computing units was investigated.**

Keywords-quasi gas dynamic equations; explicit scheme; finite volume method, nonorthogonal grid; hybrid supercomputer architecture.

I. INTRODUCTION

The development of modern hybrid computer systems is connected with massively parallel multicore processors with powerful accelerators, such as general purpose graphics processing units (GP-GPUs). They open possibilities for reducing the cost of a computer system per unit of performance and significantly reduce power consumption. These systems bring new opportunities to mathematical modeling and simulation. However, the difficulties in the efficient use of such hybrid systems are much greater than those in using conventional cluster-type high performance computers. Many of the existing sophisticated numerical methods are often not sufficient for modern high performance computer (HPC) systems. Such systems require software being created to take into account different types of processing units and a hybrid structure of memory. Experience shows that, for an effective application, it is preferable to apply algorithms as simple as possible from the logic point of view. In this regard very promising are the explicit schemes, which can be easily adapted to the computer systems with different architectures.

This paper presents further development of a program complex "Express-3D" [1], oriented on heterogeneous GPU-based computer systems. The program complex uses the explicit variant of kinetically consistent finite difference schemes based on quasi gas dynamic (QGD) equation system [2][3]. These schemes belong to the class of kinetic or Boltzmann schemes which are presently often used in the computational fluid dynamics (CFD) [4][5]. They are an effective approach to the numerical simulation of continuous media problems. The use of previous version of our program complex showed good results in solving a large number of gas dynamic problems. However, its use was limited by rectangular grids. Here, we present a new version of this program complex, which uses multi block, non orthogonal curvilinear structured hexahedral grids. Such grids give the opportunity to solve problems with complicated geometry and, on the other hand, the computational algorithm for structured grid is usually much simpler than for widely used unstructured tetrahedral grids.

In Section 2, the QGD equations system is presented and analyzed, the method of difference scheme construction is described, the stability conditions and the method of their improving are discussed. In Section 3, the results of a set of test problems simulation are presented. In Section 4, the parallel implementation of our program complex is described. The comparison is presented of speedups achieved on different GPU types. In Section 5, some conclusions are drawn based on our experience in using the program complex presented.

II. NUMERICAL METHOD

QGD equation system [2] differs from Navier-Stokes equations in some additional dissipative terms. These terms are small compared to the terms of natural viscosity and conductivity and equal to zero in flow regions where the solution satisfies the stationary Euler equations. They can be interpreted as efficient numerical stabilizers, which provide smoothness of the solution at distances of the order of mean free path. The successful use of kinetic schemes for solving a wide range of fluid dynamics problems shows that they describe viscous heat conducting flows, as well as the Navier–Stokes equations in the regions where the latter equations are applicable.

For a 3D ideal polytrophic gas flow, this system in traditional notation may be written in the form of conservation laws approximation, as follows:

$$\frac{\partial \mathbf{U}}{\partial t} - \left(\nabla \mathbf{W}_{QGD}\right)^T = 0 . \qquad (1)$$

Here $\mathbf{U} = \left(\rho, \rho u_1, \rho u_2, \rho u_3, E\right)^T$ – the vector of conservative variables, $E = \rho\left(\varepsilon + \mathbf{u}^2/2\right)$ – total energy,

$\mathbf{W}_{QGD}$ – is the matrix consisting of the conservative variables fluxes:

$$\mathbf{W}_{QGD} = \left(-\mathbf{j}_m, \Pi - \mathbf{j}_m \otimes \mathbf{u} - p\mathrm{I}, \Pi\mathbf{u} - \mathbf{q} - \mathbf{j}_m (E+p)/\rho\right). \quad (2)$$

I - is the unity matrix, vectors of mass flux ($\mathbf{j}_m$), heat flux (**q**) and viscous stress tensor (Π) are defined as follows:

$$j_{mi} = \rho(u_i - w_i), \quad w_i = \frac{\tau}{\rho}\left(\frac{\partial}{\partial x_j}\rho u_j u_i + \frac{\partial}{\partial x_i} p\right), \quad (3)$$

$$q_i = q_i^{NS} - \tau\rho u_i u_j \left(\frac{\partial}{\partial x_j}\varepsilon + p\frac{\partial}{\partial x_j}\frac{1}{\rho}\right), \; q_i^{NS} = -\kappa\frac{\partial}{\partial x_i}T, \quad (4)$$

$$\Pi_{ij} = \mu\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} - \frac{2}{3}\delta_{ij}\frac{\partial u_k}{\partial x_k}\right) + \\ + \tau\rho u_i\left(u_k\frac{\partial u_j}{\partial x_k} + \frac{1}{\rho}\frac{\partial p}{\partial x_j}\right) + \tau\delta_{ij}\left(u_k\frac{\partial p}{\partial x_k} + \gamma p\frac{\partial u_k}{\partial x_k}\right). \quad (5)$$

Closing equations are:

$$p = \rho\varepsilon(\gamma - 1), \quad T = \frac{p}{\rho R}, \quad \tau = \frac{\mu}{p\mathrm{Sc}}, \quad (6)$$

$$\mu = \mu_0 \frac{C + T_0}{C + T}\left(\frac{T}{T_0}\right)^{3/2}, \quad \kappa = \mu\frac{\gamma R}{(\gamma - 1)\,\mathrm{Pr}} \quad (7)$$

Here γ – specific ratio, Pr and Sc – Prandtl and Schmidt numbers, τ is a relaxation parameter having a dimension of time, summation is implied over repeated indices. Parameters in the Sutherland formula for viscosity may be taken from the tables. For example, for nitrogen C = 111 K, T_0 = 300.55 K, μ_0 = 17.81 μPa·c.

In order to achieve computational stability, an item proportional to the spatial step size is usually added to the expression for relaxation parameter in (6)

$$\tau = \frac{\mu}{p\mathrm{Sc}} + \alpha\frac{h}{c}, \quad (8)$$

where c – is a local speed of sound, α – is a number of the order of unity, which is adjusted experimentally. Note that for high Reynolds numbers and not very fine grids the first term in (8) is much less than the second one and may be neglected.

When the flow of non viscid gas is simulated, we have $\mu = 0$ and $\tau = \alpha h/c$. So, all dissipative terms in QGD equations system are artificial regularizers, corresponding to the artificial viscosity $\mu_{art} = \tau \cdot p \cdot Sc$. In the case of viscous flows, equations contain both terms with natural viscosity and terms with artificial one. However, this combined viscosity may be insufficient for the computational stability when hypersonic flows with strong shock waves are simulated. In this case we correct the natural viscosity by some artificial additive proportional to the spatial step size. It means that the natural viscosity μ is replaced in all Navier-Stokes terms by corrected value $\mu' = \tau \cdot p \cdot Sc$, where τ is taken from (8).

To construct difference scheme we use the control volume method. Let some regular index grid be done, which consists of arbitrary convex hexahedrons in the computational domain. Let all gas dynamic parameters be addressed to cell centers and equal to $\bar{\mathbf{U}} = V^{-1}\int_V \mathbf{U}dV$, where V is a cell volume (we will omit the dash over variables in the following text). Integrating (1) over cell volume we obtain integral form of conservation laws. Replacing time derivative by finite difference, we have the following expression for conservative variables at the next time level (here summation is held over cell faces S_l with normal vectors $\mathbf{n}_l$):

$$\hat{\mathbf{U}} = \mathbf{U} + \frac{\Delta t}{V}\sum_{l=1}^{6}\int_{S_l}(\mathbf{W}^T, \mathbf{n}_l)dS \quad (9)$$

To calculate the integrals in (9), we suppose the gas dynamic values be constant on the cell faces. These values may be calculated by linear interpolation between values in the centers of the cells adjacent to the face. The approximation of spatial derivatives in the fluxes of conservative variables is also based on the finite volume method. The control volume is constructed connected with each face of grid cells. Let this volume be the octahedron with the vertices in the four centers of the face edges and two centers of the adjacent cells. The values of gas dynamic variables in the edge center may be calculated as a quarter of the sum of their values in the centers of four adjacent cells. Consider the vector-function $\mathbf{A} = (f(x,y,z), 0, 0)$ and integrate its divergence over the volume Ω of the octahedron. By use of Gauss-Ostrogradsky formula, we have:

$$\int_\Omega div\mathbf{A}dV = \int_{\partial\Omega}(\mathbf{A}, \mathbf{n})dS = \sum_{m=1}^{8} n_{mx}\int_{S_m} fdS. \quad (10)$$

If we suppose the function *f(x,y,z)* is linear at each octahedron face, we may calculate last integral. It is equal to the product of the face square and the arithmetic mean of function values in the face vertices. On the other hand, the volume integral in (10) is equal to the product of the octahedron volume by some average value of the derivative $\partial f/\partial x$. Addressing this value to the cell face center we obtain it after algebraic transformations as a linear combination of the function values in six octahedron vertices. The coefficients of this combination only depend on

 ISBN: 978-1-61208-506-7

the grid points coordinates. Derivatives $\partial f/\partial y$ and $\partial f/\partial z$ may be expressed similarly. As a result, we have a 19-point stencil for the space approximation.

Usually, the explicit schemes impose stringent stability limitations on a time step, especially when the parabolic equations are solved, $\Delta t \sim h^2$, which is not appropriate for fine grids used in HPC calculation. Theoretical investigation of QGD based explicit schemes showed that, for numerical modeling of inviscid gas flows (when all dissipative terms in (1) – (7) are the artificial regularizers), the stability condition has a Courant type $\Delta t \sim h$, and the time step may be defined by the formula

$$\Delta t = \beta \cdot \min_i \frac{h_i}{c_i + |u_i|}, \qquad (11)$$

where h_i, c_i, u_i are the spatial step, local speed of sound and gas velocity in *i*-th grid cell, β is a coefficient (Courant number), which does not depend on spatial step size. In the case of viscous gas flow simulation the situation is more complicated. Computational experience shows that these schemes have Courant-like stability condition with β close to unity for high and medium Mach number ($Ma \geq 0.3$) flow simulation, giving the opportunity to use very fine meshes to study the fine flow structure. However, with the Mach number diminishing the stability condition tends to $\Delta t \sim h^2$, which is typical for parabolic equations. A similar situation takes place in the case of hypersonic flow simulation ($Ma > 5$), when Courant number acceptable for calculation stability does not exceed 0.1.

To improve this situation we used the flux relaxation approach, proposed in [6]. The main idea of this method is a statement that the fluxes of conservative variables at any time moment cannot achieve new values instantly. They have to relax to them, starting from the previous values with some characteristic time of flux relaxation τ_f. Thus, the system (1) is transformed to

$$\frac{\partial}{\partial t}\mathbf{U} = (\nabla \mathbf{W})^T, \quad \tau_f \frac{\partial}{\partial t}\mathbf{W} = \mathbf{W}_{QGD} - \mathbf{W}. \qquad (12)$$

For small values of τ_f or for slow processes (12) practically coincides with (1). But the new system has a hyperbolic type and, consequently, the Courant stability condition for the explicit schemes. So, we may construct finite difference scheme based on system (12) and use τ_f as a regularizing parameter. This parameter must be large enough to provide the best stability and small enough to receive the solution close to the solution of system (1). In [7], such transformation was investigated on the sample of heat conductivity equation. It was proved that, if the solution of the hyperbolic problem has no sufficient oscillations (second time derivative is not very large), then difference between solutions of parabolic and hyperbolic problems is small for small values of τ_f. Note that introducing the relaxation of fluxes, we remove the global defect of parabolic equations – infinite speed of disturbances propagation.

The second equation in (12) is the first order linear ODE, so, its solution on a time step Δt may be written, as follows:

$$\hat{\mathbf{W}} = \mathbf{W}D + \mathbf{W}_{QGD}(1-D), \;\; D = \exp\left(-\Delta t / \tau_f\right). \qquad (13)$$

So, a time step consists of three parts. At first, using the known values of gas dynamic parameters we calculate fluxes $\mathbf{W}_{QGD}$. Then we find a time step from (11) and, according to (13), correct fluxes values and find $\hat{\mathbf{W}}$. At last, we calculate new values of density, velocity components and energy from (9).

III. TEST PROBLEMS SIMULATION

The program complex was tested on a set of problems including subsonic, supersonic and hypersonic flows. Some test results were presented in [8]. Here, we present some new simulations.

The first test problem is a nitrogen flow simulation over 2D edge compression corner described in [9]. This 2D problem was solved by means of 3D program with small number of cells in the third direction (*y*). The free stream parameters were $Ma_\infty = 9.22$, $Re_\infty = 47 \cdot 10^6$ per meter, $T_\infty = 64.5$ K, stagnation temperature $T_s = 1070$ K, without boundary layer tripping ahead of the flip. The compression corner angles were taken as 15° and 38°. The distance between the left border and angle edge is 76 cm. The boundary conditions are: constant inflow parameters on the left bound, no slip conditions on the bottom, zero normal derivatives on the upper and right bounds, and periodic boundary conditions in the third direction.

The calculation results are presented in Fig. 1 and Fig. 2. Fig. 1 demonstrates stream lines on the background of the pressure distributions in the central section of the computational region. For the case of 15° angle we have an attached flow with the shock wave attached to the corner edge. If the angle is equal to 38°, the separated flow is formed with a vortex and a chock wave in front of it. It is in agreement with the experimental results [9]. The calculated maximum values of pressure in flow field also coincide with experimental data.

Fig. 2 presents the comparison of calculated wall pressure distribution in the middle section with the experimental results. The calculations were held with different values of α coefficient in (8). One can see that the results obtained with $\alpha = 0.2$ are better than for $\alpha = 0.5$. This effect is natural because the coefficient α is responsible for the artificial members in equations. Further diminishing of this coefficient leads to the computational instability.

 ISBN: 978-1-61208-506-7

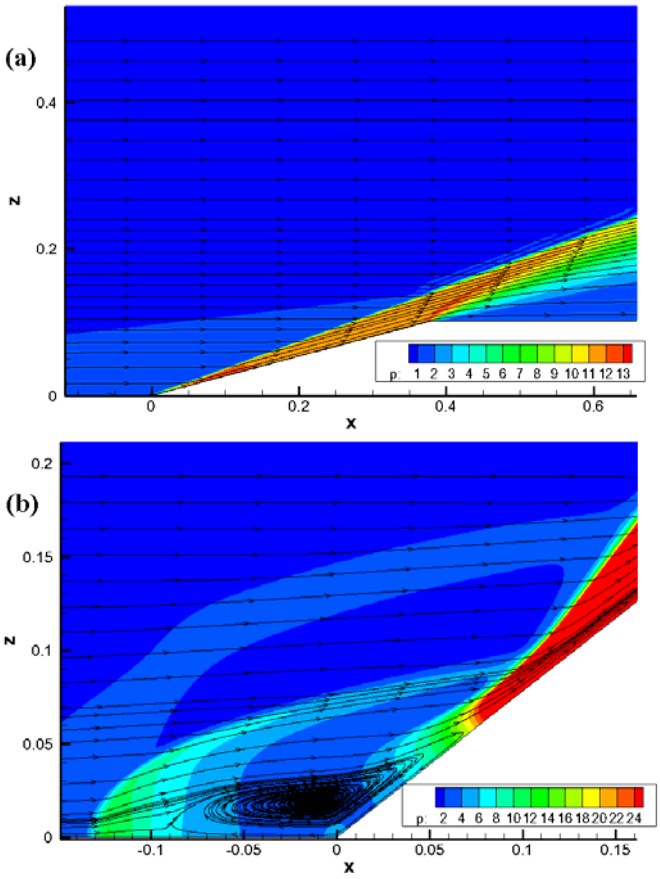

Figure 1. Stream traces near the compression corner with angle of 15° (a) and of 38° (b)

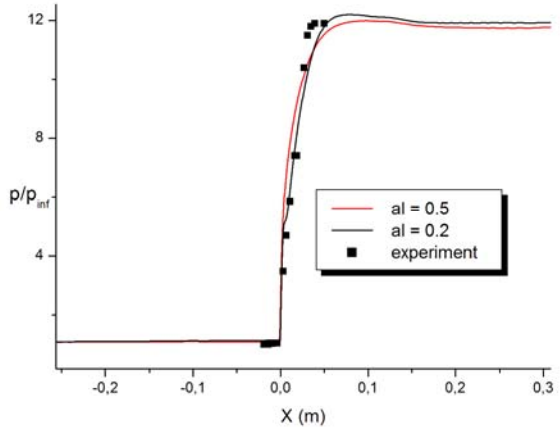

Figure 2. Wall pressure distribution for the corner angle of 15°.

The next problem was a subsonic (Ma = 0.1) flow around a hill. The Reynolds number was taken to be Re = 10^4. For such parameters a flow must be turbulent. A complicated 3D flow arises behind the hill with the separations and reattachments. Volume stream traces on the background of a density distribution are presented in Fig. 3.

Note that, according to [3], boundary conditions for subsonic flow differ from conditions for supersonic ones. This difference concerns only the inflow and outflow conditions for pressure. We state constant pressure $p = p_\infty$ at the right bound (outflow) and $\partial p/\partial n = 0$ at the left bound (inflow).

The third problem was a simulation of the wind load on a launch vehicle standing on the launch pedestal. Unlike previous problems, complicated geometry of this problem forced us to use multi block grid. The computational region was divided into a number of blocks with different index grids in each one. The grid vertices of neighbor blocks coincide on their boundary surface. The volume stream traces and pressure distribution on the vehicle surface in a stationary flow are presented in Fig. 4. The ascending vortices are clearly visible near pods.

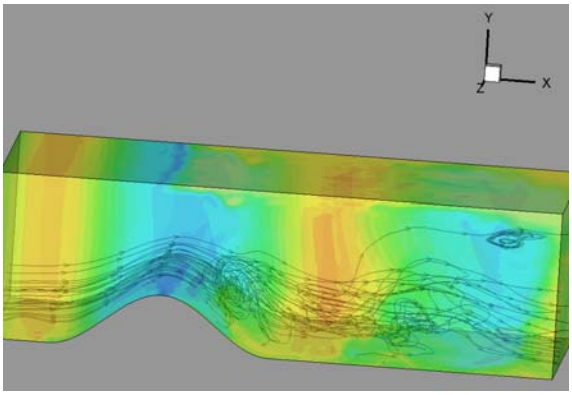

Figure 3. Volume instant stream traces behind the hill.

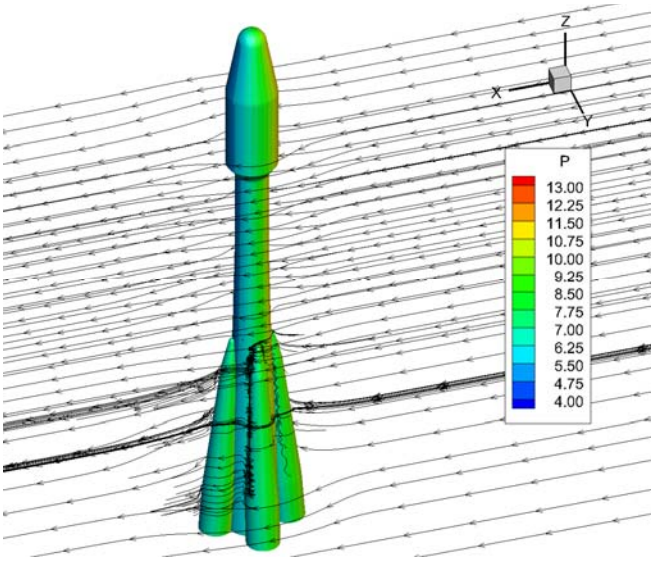

Figure 4. Flow around standing Soyuz launch vehicle.

 ISBN: 978-1-61208-506-7

IV. PARALLEL IMPLEMENTATION

There are two levels of parallelism in CFD program complex "Express-3D" [1][6]:

- The first level — geometrical decomposition. The computation domain is divided into blocks with the indexed grid in each. Blocks are distributed on MPI (Shmem) to processes so that to provide load balancing of all processes. Each MPI (Shmem) process can process one or several blocks.
- The second level — small-grain parallelism. Each mesh cell is processed by one Compute Unified Device Architecture (CUDA) stream [10].

Each face and edge have additional layers of ghost cells for organization of data exchange between blocks. We also add some additional arrays for sending and receiving data for each face and edge. The depth of additional arrays and ghost layers depends on the scheme stencil.

The orientation of faces of neighbor blocks may be arbitrary. Therefore, they are reordered before data transmission. All transferred values are packed into one array for each face and each edge.

We will call “computing process” one of the following tasks:

- The CPU core task — if calculation goes without use of GPU
- The GPU + CPU core task — if calculation goes on GPU under management of CPU

A Shmem (MPI) process corresponds to each computing process.

The data transfer is asynchronous. After processing each block new data are transferred to the neighbor blocks. If blocks are in the same computing process, there is simply a data copying from the sending array of one block to the receiving array of another one. If, on the contrary, the blocks are in different computing processes, we use shmem_put() procedure to asynchronously copy data from the sending array of one block to the receiving array of another one. Addresses of the sending and receiving arrays in each block are adjusted just after the application launch and reading the configuration file. The exchange between computing processes is executed under the control of the CPU-core. In the case of calculation on GPU it is necessary at first to copy data to corresponding CPU memory, then exchange data through Shmem (MPI) and finally copy data to GPU, if needed.

Blocks in computing process are sorted by diminishing the number of cells. For example, the big blocks are processed at first. In combination with asynchronous data transmission it is possible to assume that the main volume of data is already transferred, when processing the last (the smallest) block. The exchange procedure is finished by barrier synchronization. Actually, data exchange consists of two stages with synchronization after each of them. At the first step we exchange information from faces. After obtaining the updated information from faces, at the second step, data from edges are formed and transferred.

In [1][11], efficiency experiments were conducted with program complex "Express-3D". A set of test problems was solved based on QGD equations system on rectangular grids on a large number of graphic processors. Transition to curvilinear structured grids in addition to complication of computing algorithms, demands storage of large volumes of additional information and ensuring access to it.

Explicit finite-volume schemes are quite suitable for realization on CUDA architecture, and we will not dwell on this separately. We will note only some aspects connected with transition to non orthogonal grids.

Graphic processors of CUDA architecture with compute capability 2.0 (such as Tesla C20xx) have small number of registers on thread. As a result, a significant increase in the number of data access operations leads to a loss of efficiency. Acceleration of calculations on such processors is only 5-7 times in comparison with modern CPU cores. However, upon transition to more modern graphic processors with architecture of Kepler and compute capability 3.5 it is possible to achieve the acceptable acceleration without any changes in program code.

The results of the comparison of productivity of various devices are presented in the Table 1. The calculation times were measured for a fixed number of time steps.

TABLE I. DIFFERENT DEVICES PERFORMANCES

Computing Device	Intel Xeon E5-2670 1xCore	Nvidia Tesla C2050	Nvidia Kepler K20	Nvidia Kepler K40	Nvidia Kepler K80
Time (s)	592	175	49,70	39,85	26,81
Speedup	1	3,38	11,91	14,86	22,08

Unfortunately, there was only a small cluster containing only 4 K80 processors at the author’s disposal. That’s why we had no opportunity to make full research of scalability. However the efficiency received on 4 accelerators (more than 95%) and our previous experience in parallel computing allow to suggest that transition to a large number of GPU will be also effective in a case of non orthogonal grids as well.

V. CONCLUSION

The efficiency of using modern multicore systems including those based on NVidia GPUs largely depends on the properties of computational algorithms. On one hand, these algorithms must be logically simple; on the other hand, they must be efficient. These stringent requirements are satisfied by the algorithms based on the use of the hyperbolic variant of the quasi gas dynamic equations system.

Use of multi-block non orthogonal index hexahedral grids allows simulating gas flows in the regions with very complicated geometry as well as multiscale problems extremely close to their real behavior.

Numerical simulation of a set of test problems by use of QGD based algorithm demonstrated its good efficiency. This fact opens wide perspectives for modeling real scientific and engineering problems on modern high performance hybrid computer systems by use of explicit schemes, which are very convenient for parallel implementation.

 ISBN: 978-1-61208-506-7

ACKNOWLEDGMENT

This work was partially supported by Russian Foundation for Basic Research (grants No 15-01-03654-a, 15-01-03445-a and 16-07-00206-a).

REFERENCES

[1] B. N. Chetverushkin, E. V. Shilnikov, and A. A. Davydov, "Numerical Simulation of Continuous Media Problems on Hybrid Computer Systems," Advances in Engineering Software, vol. 60-61, pp. 42–47, 2013, http://dx.doi.org/10.1016/j.advengsoft.2013.02.003.

[2] B. N. Chetverushkin, Kinetic Schemes and Quasi-Gasdynamic System of Equations. Barcelona: CIMNE, 2008.

[3] T. G. Elizarova, Quasi-Gas Dynamic Equations. Berlin Heidelberg New York: Springer-Verlag, 2009.

[4] E. Oñate and M. Manzam, "Stabilization techniques for finite element analysis for convective-diffusion problem, " Barcelona: Publication CIMNE 183, (2000).

[5] S. Succi, The lattice Boltzmann equations for fluid dynamics and beyond, Oxford: Clarendon, 2001.

[6] A. A. Davydov, B. N. Chetverushkin, and E. V. Shilnikov, "Simulating Flows of Incompressible and Weakly Compressible Fluids on Multicore Hybrid Computer Systems," Computational Mathematics and Mathematical Physics, vol. 50, No 12, pp. 2157–2165, 2010.

[7] S. I. Repin and B. N. Chetverushkin, "Estimates of the Difference between Approximate Solutions of the Cauchy Problems for the Parabolic Diffusion Equation and a Hyperbolic Equation with a Small Parameter," Doklady Mathematics, vol. 88, No 1, pp. 417–421, 2013.

[8] A. A. Davydov and E. V. Shilnikov, "Program complex for fluid dynamic problems simulation on GPU-based computer systems," Proc. ICNAAM-2014, AIP Conference Proceedings, vol. 1648, 850071, 2015, AIP Publishing LLC, http://dx.doi.org/10.1063/1.4913126.

[9] J. G. Marvin, J. L. Brown, and P. A. Gnoffo, "Experimental Database with Baseline CFD Solutions: 2-D and Axisymmetric Hypersonic Shock-Wave/Turbulent-Boundary-Layer Interactions," NASA/TM–2013–216604, November, 2013.

[10] N. Wilt, CUDA Handbook: A Comprehensive Guide to GPU Programming. Reading MA: Addison-Wesley Professional, 2013. Available from: http://www.cudahandbook.com/.

[11] E. V. Shilnikov and A. A. Davydov. "Numerical Simulation of the Low Compressible Viscous Gas Flows on GPU-based Hybrid Supercomputers," In: Computing: Accelerating Computational Science and Engineering (CSE). Advances in Parallel Computing, M. Bader, A. Bode, H.-J. Bungartz, M. Gerndt, G.R. Joubert, F. Peters eds. IOS Press, vol. 25, pp. 315-323, 2014.

 ISBN: 978-1-61208-506-7