AIAA JOURNAL
Vol. 34, No. 4, April 1996

External Flow Computations Using Global Boundary Conditions

S. V. Tsynkov,* E. Turkel,† and S. Abarbanel‡
Tel-Aviv University, Ramat-Aviv, Tel-Aviv 69978, Israel

We numerically integrate the compressible Navier–Stokes equations by means of a finite volume technique on the domain exterior to an airfoil. The curvilinear grid we use for discretization of the Navier–Stokes equations is obviously finite, it covers only a certain bounded region around the airfoil, consequently, we need to set some artificial boundary conditions at the external boundary of this region. The artificial boundary conditions we use here are nonlocal in space. They are constructed specifically for the case of a steady-state solution. In constructing the artificial boundary conditions, we linearize the Navier–Stokes equations around the far-field solution and apply the difference potentials method. The resulting global conditions are implemented together with a pseudotime multigrid iteration procedure for achieving the steady state. The main goal of this paper is to describe the numerical procedure itself, therefore, we primarily emphasize the computation of artificial boundary conditions and the combined usage of these artificial boundary conditions and the original algorithm for integrating the Navier–Stokes equations. The underlying theory that justifies the proposed numerical techniques will accordingly be addressed more briefly. We also present some results of computational experiments that show that for the different flow regimes (subcritical and supercritical, laminar and turbulent), as well as for the different geometries (i.e., different airfoils), the global artificial boundary conditions appear to be essentially more robust, i.e., they may provide far better convergence properties and much weaker dependence of the solution on the size of computational domain than standard external boundary conditions, which are usually based on extrapolation of physical and/or characteristic variables.

I. Introduction

We consider an unbounded uniform at infinity flow of viscous compressible perfect gas past a finite body (airfoil). Such a flow is governed by the system of Navier–Stokes equations, and we are looking for a steady-state solution of this system. The numerical algorithm we develop to obtain the solution involves several steps.

First, we generate a curvilinear C-type grid around the airfoil. This grid forms a finite computational domain, which we designate D_{in} (see Fig. 1); the whole unbounded exterior to D_{in} is designated D_{ex}. The penultimate row of the C-grid nodes constitutes the curve Γ (see Fig. 1); this curve is considered as an artificial boundary, i.e., the external boundary of computational domain D_{in}. Clearly, Γ separates D_{in} and D_{ex}. The outermost row of the C-grid nodes constitutes the curve Γ_1 (see Fig. 1), Γ_1 formally belongs to D_{ex}.

We solve the Navier–Stokes equations (with nonslip boundary conditions at the surface of airfoil) inside D_{in} by means of a finite volume code.[1–3] The code is constructed using a second-order approximation to spatial derivatives on the usual 3×3 stencil. For achieving a steady state, a multigrid pseudotime iteration procedure is implemented in the code, this involves a five-stage Runge–Kutta approximation to time derivatives. Additional convergence acceleration is the use of local time stepping and residual smoothing.[1–3]

While obtaining a steady-state solution, we iteratively solve the system of algebraic equations that is the discrete counterpart of the Navier–Stokes equations on the C-grid. This discrete system is subdefinite unless we specify some artificial boundary conditions (ABCs) at Γ. Indeed, with the structure of stencil 3×3 one cannot discretize the Navier–Stokes equations at any grid node belonging to Γ_1 (since in so doing some part of stencil would simply fall out of the computational domain). Therefore, the number of equations without ABCs would be less than the number of unknowns. Consequently, the conditions at Γ should close the discrete system inside D_{in}. Such a closure must provide some additional (missing) relations between the values of unknowns corresponding to Γ and Γ_1.

The procedure for setting the ABCs incorporated in the original code[1–3] is based on extrapolation of all of the physical variables from Γ to Γ_1 at the outflow part of the external boundary and on the application of usual quasi-one-dimensional characteristic analysis on its inflow part (see Fig. 1). The resulting conditions are local and, therefore, computationally cheap. Moreover, they easily apply to artificial boundaries of any irregular shape (determined by the C-grid). These advantages, which are generally relevant to any local approach (see, e.g., Refs. 4 and 5), had stimulated wide usage of local ABCs for numerous applications. Local conditions, however, usually do not take into account the structure of solution from outside the computational domain, and, generally speaking, one can substantially improve the accuracy of calculated solution, as well as the convergence rate of pseudotime iterations, by implementing global boundary conditions (see some examples in Refs. 4 and 5). The construction of global boundary conditions for the case of external compressible viscous flow is described below.

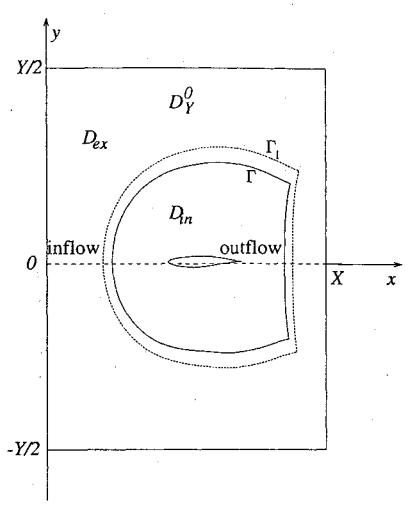

Fig. 1 Configuration of domains.

Received Dec. 1, 1994; presented as Paper 95-0562 at the AIAA 33rd Aerospace Sciences Meeting, Reno, NV, Jan. 9–12, 1995; revision received June 23, 1995; accepted for publication June 23, 1995.

*Currently National Research Council Resident Research Associate, Aerodynamics and Acoustics Methods Branch, Fluid Mechanics and Acoustics Division, NASA Langley Research Center, Mail Stop 128, Hampton, VA 23681-0001. Member AIAA.

†Professor, Department of Applied Mathematics, School of Mathematical Sciences. Member AIAA.

‡Professor, Department of Applied Mathematics, School of Mathematical Sciences.

Our global ABCs preserve the advantages of local approach, namely, they easily apply to the irregular boundary Γ and are not expensive in computational practice. At the same time, they can be constructed arbitrarily close to the exact boundary conditions, which means that one can (uniquely) complement on D_{ex} the solution obtained inside D_{in} so that the original problem for the infinite domain is solved within the prescribed accuracy. The latter problem is defined on $D_{in} \cup D_{ex}$ and has the freestream limit of the solution at infinity.

To construct the global ABCs, we assume that far enough from the immersed body (airfoil) the flow perturbations are small and, therefore, the governing equations can be considered as linear there. Specifically, we henceforth suppose that one can linearize the flow equations just outside D_{in}, i.e., in D_{ex}. Of course, whether it is possible or not to actually replace the original problem by the problem of small perturbations, depends on the location of artificial boundary Γ, i.e., on the distance between airfoil and artificial boundary. Clearly, for a very large computational domain the assumption of linearity in its exterior is definitely true, and as we approach the airfoil this assumption could always be checked a posteriori.

Once we linearize the Navier–Stokes equations (around the constant freestream background), we get a new coupled problem (nonlinear in D_{in} and linear in D_{ex}) instead of the original one. The idea of setting the global ABCs is the following.[6] We solve the linear subproblem in D_{ex} for any boundary data specified on Γ (the boundary conditions at infinity for the linearized system are zero). In other words, we parameterize the variety of solutions to the linearized Navier–Stokes equations on D_{ex} by the boundary data at Γ. In order to complete the algebraic system inside the computational domain, we need a certain number of missing relations between the variables defined on Γ and Γ_1. Therefore, if we consider the values on Γ as already known, then we need to prescribe the values on Γ_1. This can be done simply by calculating the values of the corresponding solution of linear system on Γ_1. Since we parameterize the whole variety of these solutions, then we obtain the linear operator connecting the values of velocity components u and v, pressure p, and density ρ, on Γ with those on Γ_1 through the solution of linear system. We emphasize that the desirable far-field properties of the solution are already taken into account while formulating and solving the linear subproblem on D_{ex}.

The principal question is how to solve the external linear problem on the infinite domain D_{ex}. We use the apparatus of the difference potentials method (DPM)[7,8] for this purpose. The DPM can be considered as a generalization of the methods based on the classical potential theory. Recall, in the framework of classical potentials one replaces the boundary-value problem formulated on the domain by a certain integral equation formulated on the boundary of this domain. This integral equation (the Fredholm equation of the second kind) is written with respect to the potential density; the integral involved is of the convolution type, with the corresponding fundamental solution as a kernel. The solution to the original boundary-value problem can be found in the form of a potential as soon as the potential density is determined by solving the aforementioned integral equation. In the framework of DPM, we solve a special auxiliary problem (AP) instead of calculating the convolution with the fundamental solution; the AP can be constructed even in the case when the actual fundamental solution is unknown. We also consider the generalized vector densities and generalized potentials instead of the scalar densities and classical potentials, respectively; the vector density can be chosen in a certain sense excessive, which imparts more flexibility and universality to the algorithm. Finally, instead of the Fredholm boundary integral equations with respect to the potential density, we obtain the so-called Calderon boundary pseudodifferential equations. The fundamental difference between the former and the latter is that the Fredholm equation is equivalent to a specific boundary-value problem, e.g., the Fredholm boundary integral equations are different for the Dirichlet and Neumann types of boundary data. Contrary to the equations of classical potential theory, the Calderon pseudodifferential equation formulated on the boundary is equivalent to the original differential equation (formulated on the domain) itself, irrespective to the type of boundary conditions (if any). In other words, the Calderon equation provides an exhaustive classification of those and only those boundary data (in the sense of generalized vector densities) that can be complemented on the domain so that the complement solves the original differential equation. To ensure uniqueness, the Calderon equation may be supplemented by some boundary condition, and then, the resulting system becomes equivalent to a specific boundary-value problem. Therefore, the apparatus of the Calderon equations appears to be very convenient for our purpose: to parameterize the general solution to the linearized exterior problem on D_{ex} by the corresponding boundary data.

The DPM in a narrow sense is a method for discretization and numerical solution of the Calderon boundary pseudodifferential equations. One can say that the DPM transfers the constructions of the generalized potentials and boundary pseudodifferential operators to the case of finite difference equations.

The algorithm for setting the ABCs on the basis of the DPM involves two steps: 1) formulation of the special AP on some new finite domain instead of D_{ex}, the solution of AP will possess (asymptotically) the same properties that we require from the solution of the aforementioned linear problem on D_{ex}; 2) solving the AP several times for special right-hand sides and constructing the ABCs on the basis of the obtained solutions. The numerical procedure is described next.

II. Computation of the ABCs

A. Difference Auxiliary Problem

Consider the rectangular domain $D_Y^0 = (0, X) \times (-Y/2, Y/2)$ entirely containing D_{in} and Γ_1 (see Fig. 1). Introduce in this domain a uniform Cartesian grid

$$\mathcal{N}^0 = \{(x_m, y_j) \equiv (mh_x, jh_y - Y/2) \mid m = 0, \ldots, M,\ j = 0, \ldots, 2J+1\} \tag{1}$$

with the sizes h_x, h_y. The AP is formulated for the linearized Navier–Stokes equations. We refer to Ref. 6 for details of the non-dimensionalization and linearization procedure; the result is the system of four linear partial differential equations with constant coefficients,

$$\begin{aligned}
&\frac{\partial \rho}{\partial x} + \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \\
&\frac{\partial u}{\partial x} + \frac{\partial p}{\partial x} - \frac{1}{Re_0}\left[\frac{4}{3}\frac{\partial^2 u}{\partial x^2} + \frac{1}{3}\frac{\partial^2 v}{\partial x \partial y} + \frac{\partial^2 u}{\partial y^2}\right] = 0 \\
&\frac{\partial v}{\partial x} + \frac{\partial p}{\partial y} - \frac{1}{Re_0}\left[\frac{4}{3}\frac{\partial^2 v}{\partial y^2} + \frac{1}{3}\frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 v}{\partial x^2}\right] = 0 \\
&\frac{\partial p}{\partial x} - \frac{1}{M_0^2}\frac{\partial \rho}{\partial x} - \frac{\gamma}{Re_0\, Pr}\left[\nabla^2 p - \frac{1}{\gamma M_0^2}\nabla^2 \rho\right] = 0
\end{aligned} \tag{2}$$

where M_0 is the Mach number at infinity, Re_0 is the Reynolds number based on the freestream values, Pr is the Prandtl number (constant), and u, v, p, and ρ are the perturbations of velocity components, pressure, and density, respectively.

We now construct the central difference second order approximation to Eqs. (2) on the grid $\mathcal{N}^0$, Eq. (1). Note that according to the general scheme of the DPM,[7,8] an auxiliary problem is formulated for an inhomogeneous system, and so we introduce some right-hand side (RHS) f^0 whose specific values will be prescribed later on. We symbolically write the resulting finite difference scheme as follows (see Ref. 6 for explicit expressions):

$$L^0 u^0 = f^0 \tag{3}$$

where the vector function u^0 represents the unknowns and the superscript 0 denotes that these variables refer to the AP.

We impose periodic boundary conditions in the y direction, Y being the value of the period, and take the discrete Fourier transform of both sides of Eq. (3). This leads to a certain second-order system of four ordinary difference equations. Then, we introduce

four additional variables, and for each $k, k = -J, \ldots, J$, obtain the following system of eight first-order ordinary difference equations:

$$A_k \hat{v}^0_{m,k} + B_k \hat{v}^0_{m-1,k} = \hat{g}^0_{m,k}, \qquad m = 1, \ldots, M \tag{4}$$

where the 8×8 matrices A_k and B_k and the new vectors of the unknowns $\hat{v}^0_{m,k}$ and of the RHS's $\hat{g}^0_{m,k}$ are explicitly written down in Ref. 6.

We have not yet specified for AP the boundary conditions at $x = 0$ and $x = X$, i.e., at $m = 0$ and $m = M$. We set these conditions separately for each wave number $k, k = -J, \ldots, J$, as follows:

$$\left(\prod_{|\mu_s(k)| > 1} [Q_k - \mu_s(k) I] \right) \hat{v}^0_{0,k} = 0 \tag{5a}$$

$$\left(\prod_{|\mu_s(k)| \le 1} [Q_k - \mu_s(k) I] \right) \hat{v}^0_{M,k} = 0 \tag{5b}$$

Here, $Q_k = A_k^{-1} B_k$, I is the identity 8×8 matrix, and $\mu_s(k)$, $s = 1, \ldots, 8$, are the eigenvalues of Q_k (to be found numerically in practice). Boundary conditions (5) ensure that if we formally consider Eqs. (4) on the infinite grid, $-\infty < m < \infty$, then the solution found for $0 \le m \le M$ will admit a unique complement, which is bounded downstream, i.e., as $m \to +\infty$, [condition (5b)] and vanishes upstream, i.e., as $m \to -\infty$ [condition (5a)].

The formulation of the AP is now complete; we will use the separation of variables (Fourier technique) for its solution. In so doing, we need to solve the system of ordinary difference equations (4) with boundary conditions (5) for each wave number $k, k = -J, \ldots, J$. The numerical algorithm, which is the same for all k, effectively uses the fact that the subspaces selected by boundary conditions (5a) and (5b) are the eigen subspaces of the operator Q_k. This accounts for the cheapness of the corresponding computational procedure. We do not describe the algorithm here since it is delineated in the forthcoming paper.[10]

After solving problem (4), (5) for each $k, k = -J, \ldots, J$, we use the inverse discrete Fourier transform and obtain the solution of the AP. This solution is defined on the grid $\mathcal{N}^0$ [see Eq. (1)], which belongs to domain D^0_Y (see Fig. 1).

We designate the Green (i.e., inverse) operator of the AP by G^0, this operator yields the solution u^0 of the AP in accordance with the prescribed RHS f^0: $G^0 f^0 = u^0$. The operator G^0 is used below for calculating the ABCs; see Sec. II.B.

We also need to introduce the concept of convergence for the solutions of the difference AP. The continuous AP is formulated in Ref. 6. We consider convergence of the difference solution to the continuous one while the grid size vanishes and the period Y synchronously grows, $(h_x, h_y, Y) \to (0, 0, +\infty)$. A detailed discussion on this type of convergence, as well as some estimates connecting h_x, h_y, and Y are contained in Ref. 6.

B. Global Boundary Conditions

Let us introduce the following discrete sets: ν and ν_1 are those nodes of the C-grid that constitute the curves Γ and Γ_1, respectively. Then, we consider the stencil $St_{m,j}$ of scheme (3) with its center in the node (x_m, y_j) of the grid $\mathcal{N}^0$,

$$St_{m,j} = \bigcup_{\substack{m' = m-1, m, m+1 \\ j' = j-1, j, j+1}} (x_{m'}, y_{j'})$$

and define

$$\mathcal{M}^0 = \mathcal{N}^0 \backslash \{(x_m, y_j) \mid m = 0, m = M, j = 0, \ldots, 2J+1\}$$

$$\mathcal{M}_{\text{in}} = \mathcal{M}^0 \cap D_{\text{in}}, \qquad \mathcal{M} = \mathcal{M}^0 \backslash \mathcal{M}_{\text{in}}$$

$$\mathcal{N} = \bigcup_{(x_m, y_j) \in \mathcal{M}} St_{m,j}, \qquad \mathcal{N}_{\text{in}} = \bigcup_{(x_m, y_j) \in \mathcal{M}_{\text{in}}} St_{m,j}, \qquad \gamma = \mathcal{N} \cap \mathcal{N}_{\text{in}}$$

The set γ is called the grid boundary; it consists of those nodes of the grid $\mathcal{N}^0$ that are located in a certain sense not far from Γ. Moreover, we select all nodes $\kappa \subset \mathcal{N}^0$ that are used for interpolation of any function defined on $\mathcal{N}^0$ from this grid to $\nu_1 \subset \Gamma_1$ by quadratic polynomials. Obviously, κ is located near Γ_1. Introduce also the set of collocation points $\omega \subset \Gamma$, $|\omega| = \text{const} \cdot |\gamma|^{1/2}$; the constant is normally about $1.5 \div 2$, the distribution of the collocation points is uniform on the inflow part of Γ (see Fig. 1) and more concentrated in the wake region on the outflow part (see Fig. 1). Define the space of unknowns $\Xi_\omega \ni \xi_\omega$, where ξ_ω are the eight-component vector functions defined on the set ω. We treat ξ_ω hereafter as vectors containing the values of u, v, p, and ρ (perturbations), and $\partial u/\partial n$, $\partial v/\partial n$, $\partial p/\partial n$, $\partial \rho/\partial n$, here n is a normal to Γ.

Choose any basis in Ξ_ω; it evidently contains $8|\omega|$ elements (which is the dimensionality of Ξ_ω). As stated earlier, we will parameterize the variety of solutions to Eqs. (2) in D_{ex} by the boundary data. To do that, let us first enumerate the basis in Ξ_ω; we then get an ordered sequence of $8|\omega|$ elements. For each element (basis vector) we implement the following procedure.

1) Using functions and their normal derivatives from ξ_ω, we construct two first terms of the Taylor expansion and obtain u, v, p, and ρ, on γ; let us designate this operation $\pi_{\gamma\omega}$, $v_\gamma = \pi_{\gamma\omega}\xi_\omega$, the operator $\pi_{\gamma\omega}$ obviously includes one-dimensional interpolation along Γ from ω to the bases of normals dropped from γ to Γ.

2) Consider the function

$$v_{\mathcal{N}^0}|_{m,j} = \begin{cases} v_\gamma|_{m,j}, & \text{if } (x_m, y_j) \in \gamma \\ 0, & \text{if } (x_m, y_j) \in \mathcal{N}^0 \backslash \gamma \end{cases}$$

apply the direct operator L^0 [Eq. (3)] to $v_{\mathcal{N}^0}$, and truncate the result only to $\mathcal{M}_{\text{in}}$, i.e.,

$$f_{\mathcal{M}^0}|_{m,j} = \begin{cases} L^0 v_{\mathcal{N}^0}|_{m,j}, & \text{if } (x_m, y_j) \in \mathcal{M}_{\text{in}} \\ 0, & \text{if } (x_m, y_j) \in \mathcal{M} \end{cases}$$

3) Solve the AP for the RHS $f_{\mathcal{M}^0}$, $u_{\mathcal{N}^0} = G^0 f_{\mathcal{M}^0}$, and retain the result only on γ and κ; this yields

$$P_\gamma v_\gamma \overset{\text{def}}{=} G^0 f_{\mathcal{M}^0}\big|_\gamma \qquad \text{and} \qquad P_{\kappa\gamma} v_\gamma \overset{\text{def}}{=} G^0 f_{\mathcal{M}^0}\big|_\kappa$$

4) Interpolate from κ to ν_1.

Combining steps 1–4, we obtain two operators, $P_{\gamma\omega}$ and $P_{\nu_1\omega}$, $P_{\gamma\omega}\xi_\omega = P_\gamma \pi_{\gamma\omega}\xi_\omega = u_\gamma$, $P_{\nu_1\omega}\xi_\omega = \text{interpolation } (P_{\kappa\gamma}\pi_{\gamma\omega}\xi_\omega)|_{\nu_1} = u_{\nu_1}$. Successively computing the functions $P_{\gamma\omega}\xi_\omega$ and $P_{\nu_1\omega}\xi_\omega$ for each basis vector from Ξ_ω, we obtain the matrix representations for the operators. Note, the special type of the RHS $f_{\mathcal{M}^0}$ (see step 2), as well as the fact that we actually need to know the solution of the AP only on $\gamma \cup \kappa$, implies that the numerical procedure for solving the AP will cost a total of $\mathcal{O}(M \cdot J)$ operations for each basis vector in Ξ_ω; see Ref. 6 for details.

Let $\Xi_\omega = \Xi^{(1)}_\omega \oplus \Xi^{(2)}_\omega$, where $\Xi^{(1)}_\omega \ni \xi^{(1)}_\omega$ corresponds to functions u, v, p, and ρ, and $\Xi^{(2)}_\omega \ni \xi^{(2)}_\omega$ corresponds to their normal derivatives. For any $\xi^{(1)}_\omega$, we will choose $\xi^{(2)}_\omega$ in such a way that the corresponding $v_\gamma = \pi_{\gamma\omega}\xi_\omega$ (see step 1) will be the trace of some function $u_{\mathcal{N}}$ satisfying the homogeneous equation $L^0 u_{\mathcal{N}} = 0$ and the boundary conditions of the AP. It is possible to prove[7,8] that those and only those v_γ that coincide on γ with some $u_{\mathcal{N}}$, $L^0 u_{\mathcal{N}} = 0$, satisfy the following equation:

$$P_\gamma v_\gamma = v_\gamma \tag{6}$$

Equation (6) is the main equation for any DPM-based approach.[7,8] It is a difference analog to the Calderon boundary pseudodifferential equation. Using Eq. (6), we construct a continuation of the boundary data ξ_ω that would solve the linearized Navier–Stokes equations in $D^0_Y \cap D_{\text{ex}}$ and satisfy the boundary conditions of the AP. Substituting $v_\gamma = \pi_{\gamma\omega}\xi_\omega$ into Eq. (6), we obtain $P_{\gamma\omega}\xi_\omega - \pi_{\gamma\omega}\xi_\omega = 0$. Then, introducing $Q_{\gamma\omega} = P_{\gamma\omega} - \pi_{\gamma\omega}$ and splitting this operator in accordance with the structure of direct sum $\Xi_\omega = \Xi^{(1)}_\omega \oplus \Xi^{(2)}_\omega$, $Q_{\gamma\omega}\xi_\omega = Q^{(1)}_{\gamma\omega}\xi^{(1)}_\omega + Q^{(2)}_{\gamma\omega}\xi^{(2)}_\omega$, we rewrite Eq. (6) as follows:

$$Q^{(2)}_{\gamma\omega}\xi^{(2)}_\omega = -Q^{(1)}_{\gamma\omega}\xi^{(1)}_\omega \tag{7}$$

Note that since we have already computed the matrices $P_{\gamma\omega}$ and $\pi_{\gamma\omega}$, then $Q^{(1)}_{\gamma\omega}$ and $Q^{(2)}_{\gamma\omega}$ may be obtained simply by taking the

corresponding blocks of the matrix $\boldsymbol{P}_{\gamma\omega} - \pi_{\gamma\omega}$. We may now solve Eq. (7) with respect to $\xi_\omega^{(2)}$ for any given $\xi_\omega^{(1)}$. In so doing, we express the normal derivatives in terms of functions through the solution of the linearized exterior problem.

To solve Eq. (7) with respect to $\xi_\omega^{(2)}$, we use a certain variational approach, which is delineated in Ref. 6. Then, we split the operator $\boldsymbol{P}_{\nu_1\omega}$ in accordance with the structure of direct sum $\Xi_\omega = \Xi_\omega^{(1)} \oplus \Xi_\omega^{(2)}$, $\boldsymbol{P}_{\nu_1\omega}\xi_\omega = \boldsymbol{P}_{\nu_1\omega}^{(1)}\xi_\omega^{(1)} + \boldsymbol{P}_{\nu_1\omega}^{(2)}\xi_\omega^{(2)}$, interpolate the data along Γ (from the actual nodes ν of the C-type grid to the collocation points ω), and finally obtain

$$\boldsymbol{u}_{\nu_1} = \boldsymbol{T}\boldsymbol{u}_\nu \tag{8}$$

Equality (8) is the desired global ABC. It connects the values of u, v, p, and ρ (perturbations against the freestream background) on the curves Γ and Γ_1 through the solution of the linearized Navier–Stokes equations with the freestream limit at infinity. Obviously, practical implementation of ABCs (8) is reduced to the matrix-vector multiplication.

C. Treatment of the Far Field for Turbulent Flow

To numerically simulate turbulent flows, we use an algebraic turbulence model (Baldwin–Lomax) incorporated in the code.[1–3] This model is most relevant to describing the flow in the neighborhood of airfoil. In the far field, we will use simpler approach based on the concept of effective turbulent viscosity.[11] The idea is to qualitatively describe turbulent flow (i.e., the process of turbulent mixing) as a laminar flow of model fluid having some new turbulent viscosity.

To obtain the relation between molecular and effective turbulent viscosity, we use the following considerations. First, we refer to the incompressible case and consider laminar flow. Here, we have the following distribution of u velocity (perturbation with respect to the far-field value u_0)[11]:

$$u = \frac{W}{2\sqrt{\pi\rho^2\nu u_0 x}} \exp\left(-\frac{u_0 y^2}{4\nu x}\right) \tag{9}$$

We recall[11,12] that formula (9) is obtained under the natural assumption that the far-field solution actually depends neither on the shape of the immersed body nor on the type of flow in its close vicinity but only on one constant W, which is the total drag. (Note that the dimension of W is that of force per unit length.)

For the turbulent case we assume[11,12] that the mixing length l for the wake flow is proportional to the local width b of the wake, $l/b = \text{const}$. Then, approximately replacing the value of derivative in the expression for turbulent viscosity, $\nu_t = l^2|du/dy|$, by the ratio $u_{\max}/b$, where $u_{\max}$ is the maximal deviation of actual velocity from u_0 (which corresponds to the middle of the wake), we easily obtain $\nu_t = \text{const} \cdot b \cdot u_{\max}$, which, in particular, implies that ν_t does not depend on y. We may also assume[11] that analogously to the laminar case the wake-type solution for the turbulent flow is self-similar. Then, we have $u = u_{\max} \cdot \phi(y/b)$, which immediately yields

$$W = \rho u_0 \int_{-b}^{b} u\,dy = \rho u_0 b u_{\max} \int_{-1}^{1} \phi\left(\frac{y}{b}\right) d\left(\frac{y}{b}\right)$$

and, consequently,

$$\nu_t = kW/\rho u_0 \tag{10}$$

where $k = \text{const}$. One can easily see from formula (10) that $\nu_t = \text{const}$ throughout the whole wake region and, therefore, the structure of turbulent wake behind the body (i.e., profile of mean velocity) appears to be the same as the structure of laminar wake[11,12] since we actually have the same solution as given by formula (9),

$$u = \frac{W}{2\sqrt{\pi\rho^2\nu_t u_0 x}} \exp\left(-\frac{u_0 y^2}{4\nu_t x}\right) = \frac{1}{2\sqrt{\pi}}\sqrt{\frac{W}{k\rho x}} \exp\left(-\frac{\rho u_0^2 y^2}{4kWx}\right) \tag{11}$$

only for some other constant ν_t instead of the true molecular viscosity ν.

Recall that our purpose is to find such ν_t that, being substituted into formula (9), will provide the same wake solution as we would have for the turbulent case. Clearly, ν_t from formula (10) satisfies this requirement [see formula (11)], and so let us now determine the specific value of k. Since the effective viscosity is constant throughout the whole wake region, we assume that in the far field it preserves the same value as it has in the outer part of the boundary layer near the trailing edge of the immersed airfoil. Using the Clauser conjecture[11] to calculate the latter quantity and restricting ourselves (for qualitative consideration) by the case of not high longitudinal pressure gradients, we can obtain

$$\nu_t = k_C W/\rho u_0$$

which means that the value of the unknown constant k in formula (10) may be chosen the same as the value of the Clauser constant, $k_C = 0.0168$.

To independently determine W, we recall that in the case of turbulent flow past a flat plate the drag is given by[11]

$$W = 0.0307 Re^{-\frac{1}{7}} \rho u_0^2 L \tag{12}$$

where Re is the actual Reynolds number based on the molecular viscosity and L is the characteristic length. To take into account compressibility, we multiply expression (12) by $[2/(2 + 0.5(\gamma - 1)M_0^2)]^{5/7}$ in accordance with the Tucker conjecture.[11] Then, we introduce an (empirical) constant Θ, which is the ratio of pressure and viscous contributions to the total drag (obviously, $\Theta = 0$ for the flat plate); for the transonic flow computed below, $\Theta \approx \frac{5}{2}$. Finally, we obtain the following relation:

$$Re_t = \frac{Re_0^{\frac{1}{7}}}{0.0307 k_C(\Theta + 1)} \left(\frac{2}{2 + 0.5(\gamma - 1)M_0^2}\right)^{-\frac{5}{7}}$$

which is used for determining the effective turbulent Reynolds number in all of the turbulent computations presented in the next section. We re-emphasize that the proposed treatment of turbulence in the far field is only qualitative. This approach, however, seems to be justified by the numerical experiments.

III. Numerical Results

We first calculate a transonic steady-state flow past the airfoil RAE2822, with $M_0 = 0.73$, $Re_0 = 6.5 \times 10^6$, $Pr = 0.72$, and angle of attack $\alpha = 2.79$ deg. To calculate such a flow, we have to use the grids that are very fine in the vicinity of the airfoil. The finest grid cells are located just next to the airfoil surface, with normal spacings varying from 10^{-4} to 10^{-5} chords for the family of C-type grids that we generated; in so doing, the "average radii" of computational domains vary from 3 to 30 chords of the airfoil. (Note, we use the airfoil chord as a unit of length hereafter.) All of these grids are stretched, which means that the ratio of minimal/maximal grid sizes is very small; therefore, the application of local time stepping to accelerate the convergence of iteration procedure to the steady state becomes an issue of special importance. The specific iteration procedure we use for calculating transonic flow over RAE2822 involves four levels of multigrid with W cycles, with one iteration done on the finest mesh and two on each of the coarser ones.

We compare the computational results corresponding to two types of ABCs; standard[1–3] and global (8). Standard conditions are very easily implemented in practice, the missing values at Γ_1 (i.e., ν_1) are simply extrapolated from Γ (i.e., ν) and/or prescribed from the freestream parameters (depending on the outflow/inflow type of the external boundary) at each time level. Once the solution is known on the whole C-grid including Γ_1, it is possible to advance the next time step (iteration). Practical implementation of global ABCs (8) is also rather easy, in spite of the fact that the shape of artificial boundary is irregular. At each time step we subtract the freestream values u_0, v_0 ($v_0 = 0$), p_0, and ρ_0 from actual values u, v, p, and ρ at Γ (i.e., at ν), then form the properly ordered vector of perturbations, apply the matrix $\boldsymbol{T}$, obtain the perturbations at Γ_1 (at ν_1), and add the freestream values to get the actual solution at Γ_1. Then, after recovering the internal energy from the equation of state, we may advance one time step.

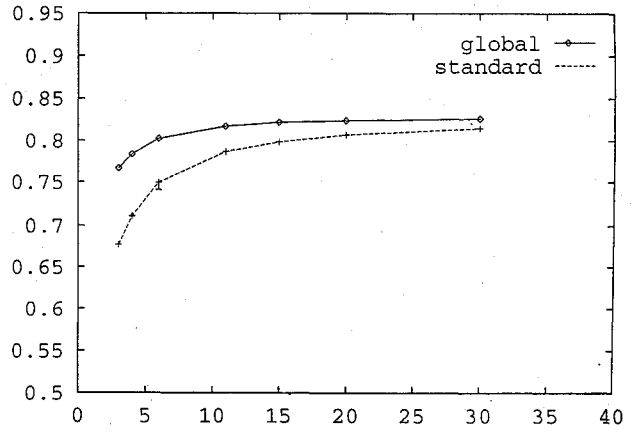

Fig. 2 Lift coefficient C_l vs average radius of computational domain for the supercritical turbulent flow around RAE2822: $M_0 = 0.73$, $Re_0 = 6.5 \times 10^6$, $\alpha = 2.79$ deg.

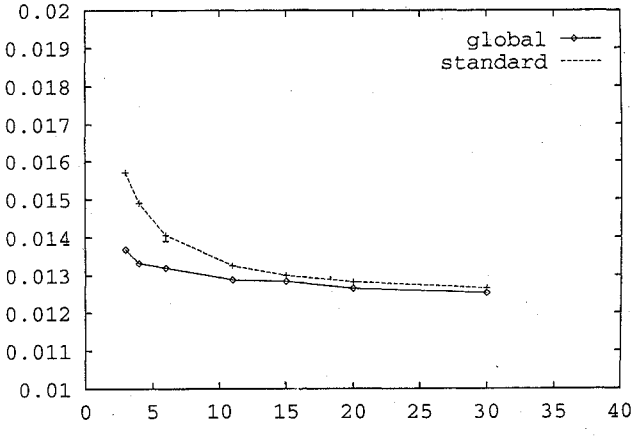

Fig. 3 Drag coefficient C_d vs average radius of computational domain for the supercritical turbulent flow around RAE2822: $M_0 = 0.73$, $Re_0 = 6.5 \times 10^6$, $\alpha = 2.79$ deg.

An important remark concerns the combined usage of global ABCs (8) and multigrid iterations. Indeed, some external boundary conditions should obviously be formulated at all levels of multigrid. As for the standard procedure, it equally applies to any level, but this is not so for global conditions (8) since the latter depend on actual locations of nodes $\nu \subset \Gamma$ and $\nu_1 \subset \Gamma_1$. Therefore, in the meantime we apply global ABCs (8) only at the finest multigrid level and simply retain the corresponding boundary values for all of the coarser levels. The question of implementation of global ABCs at all levels of multigrid (which means different operators $\boldsymbol{T}$) is the subject for future investigation.

Let us now compare the behavior of solutions obtained for either type of ABCs depending on the size (i.e., average radius) of computational domain. Namely, we conduct the series of computations for different domains and compare the values of the lift and drag coefficients C_l and C_d calculated by integrating pressure along the airfoil surface. For this series of computations, we use a sequence of C-type grids with average radii of 3, 4, 6, 11, 15, 20, and 30 chords of RAE2822 airfoil. Each grid is constructed by a hyperbolic generator and has a total of 256×64 nodes; the number of nodes on the airfoil surface is 192, grid size in the normal direction near the surface is 0.33×10^{-4}. All of the operators $\boldsymbol{T}$ that we use hereafter for setting ABCs (8) at Γ are calculated for the value of the period Y equal to four diameters of computational domain.

Figures 2 and 3 represent the dependence of lift and drag coefficients, respectively, on the size of computational domain for both types of ABCs. One can observe from these figures that already for the radius of four chords the coefficient C_l for global ABCs lies within 5% from its asymptotic value and the coefficient C_d within 6%, whereas for standard conditions we have here 12.5 and 18%, respectively.

Table 1 Viscous friction C_f depending on the size of computational domain

Average radius, chords	3	4	6	11	15	20	30
$C_f \cdot 10^3$, standard ABCs	5.418	5.363	5.296	5.218	5.185	5.165	5.137
$C_f \cdot 10^3$, global ABCs	5.270	5.232	5.197	5.158	5.143	5.128	5.112

Table 2 Force coefficients for two different grids

Normal size of grid	10^{-5}		0.33×10^{-4}	
Type of ABCs	Global	Standard	Global	Standard
C_l	0.80898	0.77896	0.81700	0.78658
C_d	0.01263	0.01311	0.01286	0.01326
$C_f \times 10^3$	5.391	5.464	5.158	5.218

One can also say that while moving off the airfoil the coefficient C_l for global ABCs appears to be within 3% range of its asymptotic value starting from the radius about 5–6 chords, and C_d starting from approximately 10 chords, whereas for standard conditions the corresponding figures are 13–15 and 15–17 chords, respectively. We also address the dependence of viscous contribution to total drag C_f ($C_D = C_d + C_f$), on the size of computational domain. The corresponding values are shown in Table 1.

From Table 1 we observe that when average radius varies from 3 to 30 chords, the coefficient C_f changes within a range of 5.4% for the standard conditions, and only within 3.0% for the global ABCs. Based on the data represented in Figs. 2 and 3 and in Table 1, we conclude that global ABCs (8) enable usage of much smaller computational domains than standard conditions do without loss of accuracy. This, in particular, confirms that taking into account the structure of solution from outside the computational domain is really a very important issue for constructing proper ABCs.

Let us additionally note the following. The entire numerical algorithm for solving external viscous flow problems appears to be very sensitive to the choice of the C-grid. The grid may influence the solution itself, as well as the characteristics of iteration procedure (see discussion later on). Therefore, we make sure that the grids used are sufficiently fine for getting reliable results. Namely, we compare the values of C_l, C_d, and C_f calculated on two different grids for one of the domains already described, with average radius of 11 chords. One grid is the same as before, with normal size near the surface 0.33×10^{-4}, the second one is three times finer, with normal size near the surface (i.e., cell size in the direction normal to the surface) 10^{-5}. The numerical results are shown in Table 2.

We observe that the corresponding values vary within $1\% \div 1.8\%$ for the dynamic coefficients and within $4.5\% \div 4.7\%$ for the viscous friction. Note, the difference for C_f is larger than for dynamic coefficients. This is reasonable since C_f is calculated by integration of viscous stress that is the derivative of solution near the airfoil surface. For the total drag C_D, $C_D = C_d + C_f$, the corresponding differences are about $2.4\% \div 2.5\%$. The data from Table 2 presumably confirms that there is no need in further refinement of the C-grid.

We now address the influence exerted by the far-field boundary conditions on the rate of convergence to steady state. We first note that there is one exceptional point in the series of computations described. Namely, we could not get good convergence in the case of standard boundary conditions for the average radius of 6 chords. This is shown by a special marker in Figs. 2 and 3. Convergence dynamics for this grid is shown in Fig. 4 both for standard and global ABCs. We have no explanation, as of yet, for this phenomenon.

One can easily see that beginning from a certain point the convergence of standard procedure fails, whereas the algorithm with global conditions still converges rather rapidly. It is interesting to notice that the convergence failure for standard procedure is not caused by the small size of computational domain (since for the average radii of 3 and 4 chords this algorithm converges) but apparently by some geometric characteristics of the grid. For the same grid, the algorithm with global ABCs does converge, which allows us to conclude that application of global ABCs improves the robustness of the whole numerical procedure.

The latter conclusion may be additionally justified by comparing the convergence rates of standard and nonlocal procedures for some

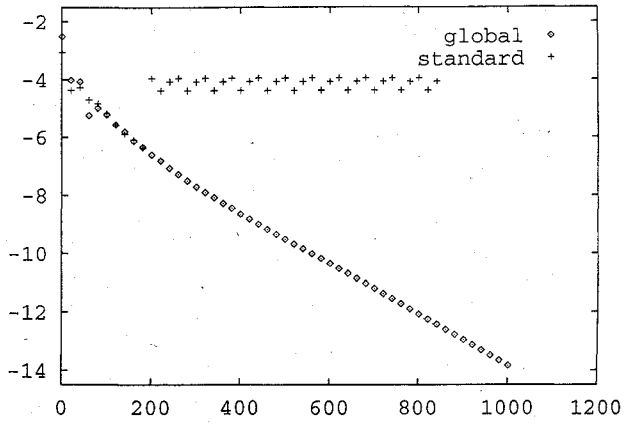

Fig. 4 Logarithm of ρ residual in L_∞ norm vs number of multigrid cycles for the supercritical turbulent flow around RAE2822: $M_0 = 0.73$, $Re_0 = 6.5 \times 10^6$, $\alpha = 2.79$ deg; average radius of computational domain is 6 chords, normal size of C-grid near the airfoil surface is 0.33×10^{-4}.

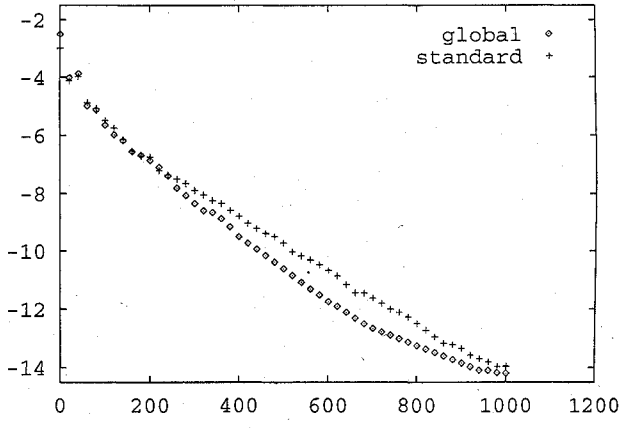

Fig. 5 Logarithm of ρ residual in L_∞ norm vs number of multigrid cycles for the supercritical turbulent flow around RAE2822: $M_0 = 0.73$, $Re_0 = 6.5 \times 10^6$, $\alpha = 2.79$ deg; average radius of computational domain is 11 chords, normal size of C-grid near the airfoil surface is 10^{-5}.

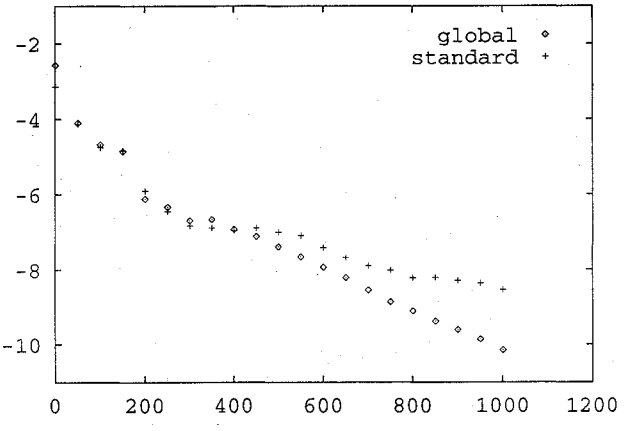

Fig. 6 Logarithm of ρ residual in L_∞ norm vs number of multigrid cycles for the supercritical turbulent flow around RAE2822: $M_0 = 0.73$, $Re_0 = 6.5 \times 10^6$, $\alpha = 2.79$ deg; average radius of computational domain is 11 chords, normal size of C-grid near the airfoil surface is 10^{-4}.

other variants of computation. Figures 5 and 6 represent such a comparison for two different cases.

One can observe that the convergence rate for nonlocal ABCs is always somewhat higher than for standard conditions. The difference in convergence rates for two types of boundary conditions may depend on the parameters of specific flow regime as well as on the characteristics of numerical algorithm, e.g., geometry of the grid and type of iteration procedure. One can see that for the grid that is coarser near the airfoil (and consequently less stretched as a whole), Fig. 6, relative convergence acceleration provided by the global ABCs is more significant than for the finer grid, Fig. 5.

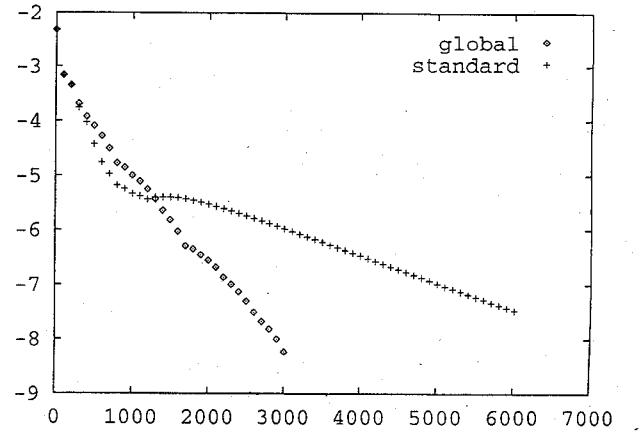

Fig. 7 Logarithm of ρ residual in L_∞ norm vs number of multigrid cycles for the subcritical laminar flow around NACA 0012: $M_0 = 0.63$, $Re_0 = 5000$, $\alpha = 2$ deg; average radius of computational domain is 10 chords, normal size of C-grid near the airfoil surface is 0.005.

The most significant convergence acceleration, however, can be obtained, thus far, for iteration procedures where we use global rather than local time stepping. We show this by calculations of laminar flows over the symmetric airfoil NACA 0012. For such a class of problems we use less stretched grids, and in so doing it appears that the multigrid iteration procedure (the same as already described, only with global time stepping) with ABCs (8) converges 2 to 3 times faster than the one with standard external conditions. Such an acceleration takes place both for subcritical and supercritical regimes of flow around NACA 0012. Details of the results of these computations are reported in Ref. 9. Here we give one example of comparing the convergence rates relevant to two types of ABCs for the laminar flow around NACA 0012; see Fig. 7.

We finally address the computational cost of global ABCs (8). Of course, this cost depends on the specific variant of computation. Namely, for the strongly stretched C-grids used for calculating turbulent flows the distance between Γ and Γ_1 is relatively large, therefore, we can use a rather coarse grid $\mathcal{N}^0$, and the additional expenditure of processor time appears to be about 5–10% of total time required for achieving reasonable accuracy, say, 10^{-8}–10^{-9}. This additional expenditure may sometimes be compensated by convergence acceleration, but even in the case when the convergence rates for global and standard ABCs are almost the same we still have much better solutions for smaller domains if we implement conditions (8), see Figs. 2 and 3. For our laminar computations (see Ref. 9), the integral cost of global ABCs (8), i.e., processor time required for calculating the matrix $\boldsymbol{T}$ as well as for multiplying a vector by this matrix at each time level, may achieve 15–20% of already reduced (by convergence acceleration) total time, which means that we still gain more than a factor of two in comparison with standard procedure.

To conclude the article we note that recently we have conducted some new numerical experiments to compare the results provided by nonlocal ABCs (8) with the ones obtained on the basis of the point-vortex model. The corresponding computations will be reported in detail in a future paper. In the meantime, we can only point out the general conclusion. Namely, our computations show that the point-vortex correction usually improves the results provided by the standard local ABCs. At the same time, nonlocal ABCs (8) still demonstrate their clear superiority over the point-vortex approach as well.

We also note that among a few possible generalizations of the described technique, the one most demanded by the computational practice is obviously the extension of the approach to three-dimensional flows. Though we did not yet try to conduct any experiments, such an extension seems to be quite feasible. The general framework of the algorithm will remain the same as described. As concerns the technical part, it will certainly be more cumbersome, in particular, because of the necessity to consider the two-dimensional

surfaces rather than the one-dimensional curves as the artificial boundaries. The highly accurate and robust ABC's in three dimensions, however, may become even more crucial for computations than for the two-dimensional case since in three dimensions there are, generally speaking, no simple models, such as point-vortex, that may improve the results provided by the standard local techniques.

References

[1]Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge–Kutta Time-Stepping Schemes," AIAA Paper 81-1259, June 1981.

[2]Swanson, R. C., and Turkel, E., "A Multistage Time-Stepping Scheme for the Navier–Stokes Equations," AIAA Paper 85-0035, Jan. 1985.

[3]Swanson, R. C., and Turkel, E., "Artificial Dissipation and Central Difference Schemes for the Euler and Navier–Stokes Equations," AIAA Paper 87-1107, June 1987.

[4]Givoli, D., "Non-reflecting Boundary Conditions," *Journal of Computational Physics*, Vol. 94, 1991, pp. 1–29.

[5]Givoli, D., *Numerical Methods for Problems in Infinite Domains*, Elsevier, Amsterdam, 1992.

[6]Ryaben'kii, V. S., and Tsynkov, S. V., "Artificial Boundary Conditions for the Numerical Solution of External Viscous Flow Problems," *SIAM Journal on Numerical Analysis*, Vol. 32, No. 5, 1995.

[7]Ryaben'kii, V. S., *Difference Potentials Method for Some Problems of Continuous Media Mechanics*, Nauka, Moscow, 1987 (in Russian).

[8]Ryaben'kii, V. S., "Boundary Equations with Projections," *Russian Mathematical Surveys*, Vol. 40, No. 2, 1985, pp. 147–183.

[9]Tsynkov, S. V., "An Application of Nonlocal External Conditions to Viscous Flow Computations," *Journal of Computational Physics*, Vol. 116, 1995, pp. 212–225.

[10]Ryaben'kii, V. S., and Tsynkov, S. V., "An Effective Numerical Technique for Solving a Special Class of Ordinary Difference Equations," *Applied Numerical Mathematics* (to be published).

[11]Loytsyansky, L. G., *Mechanics of Fluid and Gas*, Nauka, Moscow, 1987 (in Russian).

[12]Schlichting, H., *Boundary Layer Theory*, McGraw–Hill, New York, 1968.