AN APPLICATION OF THE DIFFERENCE POTENTIALS METHOD TO SOLVING EXTERNAL PROBLEMS IN CFD *

Victor S. Ryaben'kii†

Semyon V. Tsynkov‡

†Keldysh Institute for Applied Mathematics, Russian Ac. Sci., Moscow, Russia.
‡NASA Langley Research Center, Hampton, VA, U.S.A.

Abstract

Numerical solution of infinite-domain boundary-value problems requires some special techniques that would make the problem available for treatment on the computer. Indeed, the problem must be discretized in a way that the computer operates with only finite amount of information. Therefore, the original infinite-domain formulation must be altered and/or augmented so that on one hand the solution is not changed (or changed slightly) and on the other hand the finite discrete formulation becomes available.

One widely used approach to constructing such discretizations consists of truncating the unbounded original domain and then setting the artificial boundary conditions (ABC's) at the newly formed external boundary. The role of the ABC's is to close the truncated problem and at the same time to ensure that the solution found inside the finite computational domain would be maximally close to (in the ideal case, exactly the same as) the corresponding fragment of the original infinite-domain solution.

Let us emphasize that the proper treatment of artificial boundaries may have a profound impact on the overall quality and performance of numerical algorithms. The latter statement is corroborated by the numerous computational experiments and especially concerns the area of CFD, in which external problems present a wide class of practically important formulations.

In this paper, we review some work that has been done over the recent years on constructing highly accurate nonlocal ABC's for calculation of compressible external flows. The approach is based on implementation of the generalized potentials and pseudodifferential boundary projection operators analogous to those proposed first by Calderon. The difference potentials method (DPM) by Ryaben'kii is used for the effective computation of the generalized potentials and projections. The resulting ABC's clearly outperform the existing methods from the standpoints of accuracy and robustness, in many cases noticeably speed up the multigrid convergence, and at the same time are quite comparable to other methods from the standpoints of geometric universality and easiness of implementation.

1 Basic Ideas

1.1 Preliminaries

Let us consider two domains: the finite computational domain D_{in} and its infinite complement to the entire plane (entire space) D_{ex}; the domains are separated by the artificial boundary , , which is supposed to be a simple closed curve (surface). We originally formulate our problem on $\bar{D}_{in} \cup D_{ex}$; specifically, we are looking for a vector-function $\mathbf{u} = \mathbf{u}(\mathbf{r}, t)$, $\mathbf{r}$ represents the space coordinates and t is the time, that satisfies some (generally speaking, nonlinear) system of partial differential equations (PDE's)

$$\Phi\left(\mathbf{u}, \frac{\partial \mathbf{u}}{\partial t}, \nabla \mathbf{u}, \ldots\right) = \mathbf{g}(\mathbf{r}, t), \qquad \mathbf{r} \in \bar{D}_{in} \cup D_{ex}, \tag{1a}$$

as well as some boundary conditions at infinity

$$\phi(\mathbf{u}) \longrightarrow \mathbf{0}, \quad \text{as} \quad r \equiv |\mathbf{r}| \longrightarrow \infty. \tag{1b}$$

We will always assume that the problem (1) is uniquely solvable and well-posed. Later on, we will think of $\mathbf{u}$ as of the hydrodynamic variables, e.g., velocity components, density, and pressure. In

*This paper was prepared while the second author held a National Research Council Research Associateship at NASA Langley Research Center, Hampton, VA 23681-2199, U.S.A.

†Miusskaya sq. 4, 125047 Moscow, Russia. Fax: (7-095)972-0737; E-mail: ryab@applmat.msk.su.

‡Correspondence author. Permanent address: Dept. of Appl. Mathematics, School of Mathematical Sciences, Tel Aviv University, Ramat Aviv, Tel Aviv 69978, Israel. Fax: (972-3)640-9357, E-mail: tsynkov@math.tau.ac.il; URL: http://fmad-www.larc.nasa.gov:80/~tsynkov/.

this case, system (1a) is one of the relevant systems of PDE's used in fluid dynamics, e.g., the Navier-Stokes equations, and boundary conditions (1b) may reflect, e.g., the fact that as the coordinate approaches infinity the local flow parameters approach the free-stream ones. We will also assume that in the case of an external flow problem the computational domain D_{in} entirely contains the immersed body(ies), and equations (1a) include the proper boundary conditions at the solid surface(s). In this case, we may think (for simplicity) that $\bar{D}_{in} \cup D_{ex} = R^n$, $n = 2$ or $n = 3$.

The next step is to assume that in the domain D_{ex} system (1a) is either linear and homogeneous from the very beginning or admits replacement by some linear homogeneous system of PDE's under certain conditions. Analogously, we will assume that boundary conditions (1b) can also be thought of as some linear homogeneous relations. Then, instead of (1) we consider

$$\Phi\left(\mathbf{u},\, \frac{\partial \mathbf{u}}{\partial t},\, \nabla \mathbf{u}, \ldots\right) = \mathbf{g}(\mathbf{r},\, t), \quad \mathbf{r} \in D_{in}, \tag{2a}$$

$$\mathbf{L}\tilde{\mathbf{u}} = \mathbf{0}, \quad \mathbf{r} \in \bar{D}_{ex}, \tag{2b}$$

$$\mathbf{l}\tilde{\mathbf{u}} \longrightarrow \mathbf{0}, \quad \text{as} \quad r \longrightarrow \infty. \tag{2c}$$

In (2b) and (2c), $\mathbf{L}$ and $\mathbf{l}$ are some linear operators, and $\tilde{\mathbf{u}} = \tilde{\mathbf{u}}(\mathbf{r},\, t)$ may either coincide with $\mathbf{u}$ or be some function of $\mathbf{u}$. For example, system (2b) can be obtained on the basis of (1a) using linearization in D_{ex}. (Note, linearization of the governing equations in the far field against the constant free-stream background is relevant to many external flows.) Then, $\tilde{\mathbf{u}} = \mathbf{u} - \mathbf{u}_0$, where $\mathbf{u}_0$ represents the corresponding free-stream parameters. Analogously to (1), we will assume that problem (2) is uniquely solvable and well-posed.

Our final goal will be to find the solution $\mathbf{u}$ to (2a) on D_{in}. However, we cannot do it directly since system (2a) is not closed if considered separately. On the other hand, we also cannot directly solve on the computer neither (1) nor (2) because D_{ex} is infinite. Consequently, we will need to supplement system (2a) by the special artificial boundary conditions (ABC's) at , , so that the resulting problem

$$\Phi\left(\mathbf{u},\, \frac{\partial \mathbf{u}}{\partial t},\, \nabla \mathbf{u}, \ldots\right) = \mathbf{g}(\mathbf{r},\, t), \quad \mathbf{r} \in D_{in}, \tag{3a}$$

$$\mathbf{B}_\Gamma\left(\mathbf{u},\, \frac{\partial \mathbf{u}}{\partial t},\, \nabla \mathbf{u}, \ldots\right) = \mathbf{0}, \quad \mathbf{r} \in , , \tag{3b}$$

is closed and its solution on D_{in} is in a certain sense close to the solution of (2). To construct the ABC's (3b), we will use the linearity of (2b)–(2c).

Before proceeding further, let us define here an important concept of exact ABC's, which we will refer to henceforth. Namely, if the ABC's (3b) are constructed so that the solution to (3) and (2) coincide on D_{in}, then we will call such boundary conditions the exact ABC's. Note, we always assume that the solutions of (1) and (2) are close enough and always treat the exactness of ABC's within the accuracy of replacement of (1) by (2) (e.g., within the accuracy of far-field linearization). Clearly, we can reformulate the definition of exact ABC's as follows: one should be able to uniquely complement the solution of (3) from D_{in} to D_{ex} so that the resulting function on $\bar{D}_{in} \cup D_{ex}$ solves (2).

An extensive work has been done by many authors over the recent years towards constructing the exact ABC's for different problems, as well as towards developing the effective approximate boundary conditions. A survey of this work can be found, e.g., in the recent review papers by Tsynkov [1, 2], as well as in the comprehensive reviews by Givoli [3, 4].

1.2 Boundary Equations with Projections

Clearly, the idea of exact ABC's is that the relation (3b) should equivalently replace (2b)–(2c). Since the equations (2b)–(2c) are linear we can use the apparatus of generalized potentials and boundary projections in order to obtain the desirable boundary conditions. In so doing, $\mathbf{B}_\Gamma$ from (3b) appears to be some linear pseudodifferential operator.

The generalized potentials and boundary projection operators that we employ for constructing the ABC's were first proposed by Calderon [5] and then also studied by Seeley [6]. Ryaben'kii [7, 8] (see also the recent paper [9] and the book by Mikhlin, et al. [10]) had extended and modified the original approach, and also proposed an effective numerical technique for calculation of potentials and projections, this technique is called the difference potentials method (DPM). Additionally, Ryaben'kii in [11] had for the first time shown how the generalized potentials and projections could be used for constructing the ABC's for unsteady finite-difference problems.

We, however, will for the moment restrict ourselves by considering the steady-state problems only, which means $\mathbf{u} = \mathbf{u}(\mathbf{r})$ and $\tilde{\mathbf{u}} = \tilde{\mathbf{u}}(\mathbf{r})$. To simplify the notations, we will further omit the tilde that distinguishes between the "linear" and "nonlinear" solutions; it should not cause misunderstandings since hereafter we will mostly concentrate on the "linear part" of (2).

On $\bar{D}_{in} \cup D_{ex}$, we introduce the following auxiliary problem (AP)

$$\mathbf{L}\mathbf{u} = \mathbf{f}, \quad \mathbf{r} \in \bar{D}_{in} \cup D_{ex}, \quad \operatorname{supp} \mathbf{f}(\mathbf{r}) \subset D_{in}, \tag{4a}$$

$$\mathbf{lu} \longrightarrow \mathbf{0}, \quad \text{as} \quad r \longrightarrow \infty, \tag{4b}$$

which is the key element of our further construction. In the theory of generalized potentials, the solution of the AP (4) plays approximately the same role as the convolution with the fundamental solution plays in the classical potential theory. We will assume that problem (4) is uniquely solvable for any compactly supported right-hand side (RHS) $\mathbf{f}(\mathbf{r})$ and well-posed; the corresponding inverse (Green's) operator is denoted $\mathbf{G}$. For any function $\mathbf{u}(\mathbf{r})$, $\mathbf{r} \in \bar{D}_{ex}$, we also introduce its clear trace ξ_Γ [7, 8] on , ,

$$\xi_\Gamma = \mathbf{Tr}_\Gamma \mathbf{u}. \tag{5}$$

For the applications considered below ($\mathbf{L}$ in (2b) and (4a) is a second-order operator), ξ_Γ is chosen as a vector-function with the components $\left(\mathbf{u}, \dfrac{\partial \mathbf{u}}{\partial \zeta}\right)\Big|_\Gamma$, ζ is the normal to , . The corresponding space of vector-functions ξ_Γ is called the space of clear traces. For any element ξ_Γ of this space, we define the generalized potential with the density ξ_Γ:

$$\mathbf{P}\xi_\Gamma = \Big(\mathbf{G}\Big((\mathbf{Lw})\big|_{D_{in}}\Big)\Big)\Big|_{\bar{D}_{ex}}. \tag{6}$$

Here, $\mathbf{w} = \mathbf{w}(\mathbf{r})$ is an arbitrary function with the only requirement of that it has the clear trace ξ_Γ, $\mathbf{Tr}_\Gamma \mathbf{w} = \xi_\Gamma$, and the RHS for the AP, i.e., the function that the Green operator $\mathbf{G}$ operates on in equation (6), is given by

$$(\mathbf{Lw})\big|_{D_{in}} = \begin{cases} \mathbf{L}\,\mathbf{w}, & \text{on } D_{in}, \\ \mathbf{0}, & \text{on } \bar{D}_{ex}. \end{cases} \tag{7}$$

We also define the operator $\mathbf{P}_\Gamma$,

$$\mathbf{P}_\Gamma \xi_\Gamma = \mathbf{Tr}_\Gamma \mathbf{P} \xi_\Gamma, \tag{8}$$

as the trace (5) of the generalized potential (6); this operator obviously maps the space of clear traces ξ_Γ onto itself. It turns out that $\mathbf{P}_\Gamma$ in (8) is a projection, $\mathbf{P}_\Gamma^2 = \mathbf{P}_\Gamma$, it is called the Calderon boundary projection. This operator will play a fundamental role in our further analysis. It is possible to show [7, 8] that those and only those ξ_Γ that belong to the image of the projection, $\xi_\Gamma \in \mathrm{Im}\mathbf{P}_\Gamma$, i.e., satisfy the boundary equation with projection (BEP)

$$\mathbf{P}_\Gamma \xi_\Gamma = \xi_\Gamma, \tag{9}$$

can be complemented from , to D_{ex} so that the complement $\mathbf{u} = \mathbf{u}(\mathbf{r})$, $\mathbf{r} \in \bar{D}_{ex}$ solves (2b)–(2c). In other words, BEP (9) provides for a complete classification of those and only those densities ξ_Γ of generalized potentials that are actually the traces of some solution $\mathbf{u} = \mathbf{u}(\mathbf{r})$ to (2b)–(2c) on $\bar{D}_{ex}$. When the BEP (9) is satisfied, the corresponding complement on $\bar{D}_{ex}$ can be restored as the potential (6). Clearly, since $\mathrm{Im}\mathbf{P}_\Gamma$ contains all those and only those ξ_Γ that can be complemented on D_{ex} so that the complement solves (2b)–(2c), then equation (9) equivalently replaces system (2b)–(2c), which means the BEP (9) serves as a desirable exact ABC of type (3b). We emphasize that these ABC's are usually nonlocal; on the other hand, the algorithm for constructing the ABC's does not impose any essential limitations on the shape of , .

BEP (9) provides for a general recipe to construct the exact ABC's. However, any specific problem requires its own special approaches. First, certain assumptions in regard to the behavior of solution have to be done in order to pass from (1) to (2). For example, to linearize the governing equations one has to assume that the flow perturbations in the far field are small; this assumption can usually be verified a posteriori. Second, the AP (4), which needs to be actually solved for calculating $\mathbf{P}_\Gamma$, is still formulated on the infinite domain. Special techniques must be employed to replace it by some finite-domain AP. Third, in practice the ABC's are to be constructed for the discrete rather than for the continuous formulation of the problem, which means that some additional steps may be required for the actual implementation of BEP (9). Finally, the numerical process of the DPM [7, 8], which is used for the actual calculation of ABC's, may also vary in some details from one problem to another. In Section 2, we will delineate the corresponding algorithm for the Navier-Stokes equations. Here, we will briefly describe a simpler case of the two-dimensional external inviscid flows.

1.3 An Inviscid Example

Tsynkov [12, 13] and Sofronov and Tsynkov [14] consider a finite body (airfoil) immersed in an infinite flow of inviscid compressible fluid. The flow is governed by the Euler equations and is assumed to be subsonic at infinity; for the purpose of numerical solution, the equations are discretized on a finite-difference O-type grid that is generated around the body. The computational domain D_{in} in [12, 13, 14] is formed by the grid; no special assumptions in regard to the shape of its external boundary , are done. Outside the computational domain, i.e., in D_{ex}, the Euler equations are linearized against the free-stream background. Moreover, we assume the existence of the velocity potential in the far field and split the linearized system into elliptic (velocity) and advection (entropy) parts. After the term

associated with the circulation of flow around the airfoil is subtracted, the regular part of the potential of velocity perturbations can be described by the Prandtl-Glauert equation

$$\left(1 - {M_0}^2\right) \frac{\partial^2 \varphi}{\partial x^2} + \frac{\partial^2 \varphi}{\partial y^2} = 0 \tag{10a}$$

with zero boundary condition at infinity:

$$\varphi(x, y) \longrightarrow 0, \quad \text{as} \quad x^2 + y^2 \longrightarrow \infty. \tag{10b}$$

Equation (10a) can be easily reduced to the Laplace equation by means of an affine coordinate transform; in so doing, boundary condition (10b) obviously remains intact. To obtain the ABC's for velocity components, we represent the solution to (10) in the form of a generalized potential and then construct the corresponding BEP. We use the AP formulated on a ring-shaped domain $\{R_1 \le r \le R_0\}$; the external circle $\{r = R_0\}$ encompasses , and the internal circle $\{r = R_1\}$ lies inside D_{in}. At $r = R_1$, we specify homogeneous Dirichlet boundary conditions; at $r = R_0$, we specify boundary conditions

$$\left.\frac{d\hat{\varphi}_k}{dr}\right|_{r=R_0} + \left.\frac{|k|}{r}\hat{\varphi}_k\right|_{r=R_0} = 0, \tag{11}$$
$$k = 0, \pm 1, \pm 2, \ldots,$$

where $\hat{\varphi}_k$, $k = 0, \pm 1, \pm 2, \ldots$, are the Fourier components of the potential (10) after the transform with respect to the polar angle. It is possible to make sure that this AP is uniquely solvable and well-posed for any RHS concentrated inside the ring. Moreover, it is easy to see that once complemented to the exterior of the ring, the solution to this AP vanishes at infinity (compare to (4b)). Numerically, the AP is easy to solve by means of the discrete Fourier transform in polar coordinates.

Analogously to the continuous formulations, the ABC's in the discrete form should simply close the system that is solved inside D_{in}. For a second-order scheme employed inside the computational domain, we can always assume that the velocity components on the penultimate coordinate row , of the O-type grid are known, whereas the corresponding values on the outermost row , $_1$ should be determined using the ABC's. Therefore, we consider the penultimate coordinate line , as the actual artificial boundary. Referring the reader to the original work [12, 13, 14], as well as to Section 2, in which the Navier-Stokes case is analyzed, we skip here the details of the numerical procedure for calculating generalized potentials and projections and setting the inviscid ABC's. We only mention that after the finite-difference BEP is obtained, it is solved so that the velocity on , provided from inside D_{in} coincides with the gradient of the potential (10) in a certain generalized sense. Once the BEP is solved, we can find the discrete potential density for any data (velocity components) specified on , . When this grid density is known, we calculate the generalized potential and find the trace of its gradient on the outermost coordinate line , $_1$ by means of interpolation; this procedure yields the ABC's for velocities. Finally, the ABC's for thermodynamic parameters are obtained using local relations, specifically, the Bernoulli equation and the entropy advection equation.

The technique of [12, 13, 14] for constructing the ABC's was combined with an iterative method [15] by Sofronov for calculating steady solutions to the Euler equations. A few subsonic and transonic airfoil flows have been numerically studied; in the transonic case, local supercritical regions were always kept inside D_{in} so that one could treat the exterior linearized flow in D_{ex} as purely subsonic. Numerical results presented in [14] demonstrate clear superiority of the nonlocal DPM-based ABC's over the standard local techniques based on quasi-one-dimensional characteristic analysis. For a fixed computational domain, nonlocal ABC's [12, 13, 14] provide for better accuracy and faster convergence rate than local techniques; the entire numerical algorithm also appears to be more robust. Additionally, when the artificial boundary approaches the airfoil, the solution obtained with the technique of [12, 13, 14] appears to be essentially less influenced by the decrease of the size of computational domain than the solution obtained on the basis of the local boundary conditions. In other words, ABC's [12, 13, 14] allow one to maintain good accuracy of computations for much smaller computational domains than the standard boundary conditions do. These results, along with the geometric universality of ABC's [12, 13, 14], make this approach useful for calculating external Euler flows.

Analyzing this Euler example, we can see that the principle favorable circumstance that allowed us to relatively easy obtain the ABC's is the availability of a finite-domain AP. This, in turn, is due to the fact that the Poisson equation admits variables separation in polar coordinates. However, one obviously cannot rely on such circumstances in constructing the ABC's for more general cases. For example, the linearized Navier-Stokes equations (see Section 2) and even the linearized Euler equations (without introducing the potential) most likely do not admit the separation of variables in any other coordinate system except in the Cartesian one. Therefore, we

need to develop some technique that would generally allow us to approximate the solutions of the infinite-domain AP of type (4) by the solutions of a new finite-domain AP. In so doing, we should be able to control the corresponding approximation error so that the resulting ABC's be as close to the exact ones as desired. Below, the procedure for constructing appropriate finite-domain approximations to the AP's of type (4) is described for a particular class of systems of PDE's with constant coefficients. For wider classes of systems, the possibility to use the same procedure is corroborated by the numerical experiments.

1.4 Systems of Simple Structure

Hereafter in this subsection, we assume that all functions depend on only two space variables, i.e., $\mathbf{r} = (x,\, y)$, x and y being scalars. This assumption allows us to simplify the presentation and at the same time does not imply any loss of generality since the analysis of the case $\mathbf{r} = (x,\, \mathbf{y})$, $\mathbf{y}$ being a vector, is straightforward.

Let us consider the system of PDE's

$$\frac{\partial \mathbf{u}}{\partial x} = \mathbf{A}\left(\frac{\partial}{\partial y}\right)\mathbf{u} + \mathbf{f}(x,\, y), \qquad (12)$$
$$-\infty < x,\, y < \infty,$$

with respect to a n-component vector-function $\mathbf{u} = \mathbf{u}(x,\, y)$. In (12), $\mathbf{A}\left(\frac{\partial}{\partial y}\right)$ is a $n \times n$ matrix, each entry of this matrix is a symbolic polynomial of $\frac{\partial}{\partial y}$. The RHS $\mathbf{f}(x,\, y)$ is a compactly supported n-component vector-function, $\mathbf{f}(x,\, y) \equiv \mathbf{0}$ for $|x-a| > a$, $|y| > a$, a being some positive constant. System (12) can obviously be thought of as a particular class of systems of type (4a); consideration of only the first-order derivatives with respect to x presents no loss of generality since the higher derivatives can be eliminated by introducing additional variables.

We designate by $\mathbf{A}(i\alpha)$ the matrix that is obtained by substituting $i\alpha$ into $\mathbf{A}\left(\frac{\partial}{\partial y}\right)$ instead of $\frac{\partial}{\partial y}$. The entries of the matrix $\mathbf{A}(i\alpha)$ are, therefore, some polynomials of the real variable α (the coefficients of these polynomials may, generally speaking, be complex).

Definition 1 *System (12) is called the system of simple structure if the roots $\lambda_j(\alpha)$ of the equation $\det\|\mathbf{A}(i\alpha) - \lambda\mathbf{I}\| = 0$ ($\mathbf{I}$ is the $n \times n$ identity matrix) satisfy the inequalities*

$$|\Re\lambda_j(\alpha)| \geq \delta(\alpha) \geq \delta > 0, \qquad (13)$$

where $\delta > 0$ does not depend on α.

Let now U be a class of n-component vector-functions, in which we will be looking for the solutions $\mathbf{u}(x,\, y)$ of system (12). The inclusion $\mathbf{u}(x,\, y) \in U$ holds for all those and only those vector-functions that satisfy the following conditions.

- Each component of the vector-function $\mathbf{u}(x,\, y)$ has continuous derivatives of all those orders that are involved in system (12).
- Vector-function $\mathbf{u}(x,\, y)$ can be represented as a Fourier integral:

$$\mathbf{u}(x,\, y) = \frac{1}{\sqrt{2\pi}}\int\limits_{-\infty}^{\infty} \hat{\mathbf{u}}(x,\, \alpha)e^{i\alpha y}d\alpha, \qquad (14a)$$

$$\hat{\mathbf{u}}(x,\, \alpha) = \frac{1}{\sqrt{2\pi}}\int\limits_{-\infty}^{\infty} \mathbf{u}(x,\, y)e^{-i\alpha y}dy. \qquad (14b)$$

- The derivatives involved in (12) can be represented as

$$\frac{\partial \mathbf{u}}{\partial x} = \frac{1}{\sqrt{2\pi}}\int\limits_{-\infty}^{\infty} \left[\frac{d}{dx}\hat{\mathbf{u}}(x,\, \alpha)\right] e^{i\alpha y}d\alpha, \qquad (15a)$$

$$\mathbf{A}\left(\frac{\partial}{\partial y}\right)\mathbf{u} = \frac{1}{\sqrt{2\pi}}\int\limits_{-\infty}^{\infty} \mathbf{A}(i\alpha)\hat{\mathbf{u}}(x,\, \alpha)e^{i\alpha y}d\alpha. \qquad (15b)$$

- Fourier transformation $\hat{\mathbf{u}}(x,\, \alpha)$ of the function $\mathbf{u}(x,\, y)$ is bounded for each α,

$$|\hat{\mathbf{u}}(x,\, \alpha)| < \infty. \qquad (16)$$

Note, $|\hat{\mathbf{u}}(x,\, \alpha)|$ in (16) can be thought of as the sum of the absolute values of components.

Further, for any real $Y > 2a$ we associate with (12) another system of PDE's

$$\frac{\partial \mathbf{u}_Y}{\partial x} = \mathbf{A}\left(\frac{\partial}{\partial y}\right)\mathbf{u}_Y + \mathbf{f}_Y(x,\, y), \qquad (17)$$
$$-\infty < x,\, y < \infty,$$

with the unknowns $\mathbf{u}_Y = \mathbf{u}_Y(x,\, y)$. In (17), $\mathbf{f}_Y(x,\, y)$ is a n-component vector-function that is periodic in the y direction with the period Y and that coincides with the RHS $\mathbf{f}(x,\, y)$ of system (12) for $|y| < Y/2$.

Let U_Y be a class of n-component vector-functions, in which we will be looking for the solutions $\mathbf{u}_Y(x,\, y)$ of system (17). The function $\mathbf{u}_Y(x,\, y)$ belongs to U_Y if and only if it satisfies the following conditions.

- Each component of $\mathbf{u}_Y(x,\,y)$ has continuous derivatives of all those orders that are involved in system (17); moreover, $\mathbf{u}_Y(x,\,y\pm Y) = \mathbf{u}_Y(x,\,y)$.

- Vector-function $\mathbf{u}_Y(x,\,y)$ can be represented as a Fourier series:

$$\mathbf{u}_Y(x,\,y) = \sum_{k=-\infty}^{k=\infty} \hat{\mathbf{u}}_{Y_k}(x) e^{iky\frac{2\pi}{Y}}, \tag{18a}$$

$$\hat{\mathbf{u}}_{Y_k}(x) = \frac{1}{Y}\int_{-Y/2}^{Y/2} \mathbf{u}_Y(x,\,y) e^{-iky\frac{2\pi}{Y}} dy. \tag{18b}$$

- The derivatives involved in (17) can be represented as

$$\frac{\partial \mathbf{u}_Y}{\partial x} = \sum_{k=-\infty}^{k=\infty} \left[\frac{d}{dx}\hat{\mathbf{u}}_{Y_k}(x)\right] e^{iky\frac{2\pi}{Y}}, \tag{19a}$$

$$\mathbf{A}\left(\frac{\partial}{\partial y}\right)\mathbf{u}_Y = \sum_{k=-\infty}^{k=\infty} \mathbf{A}\left(i\frac{2\pi k}{Y}\right)\hat{\mathbf{u}}_{Y_k}(x) e^{iky\frac{2\pi}{Y}}. \tag{19b}$$

- Fourier coefficients $\hat{\mathbf{u}}_{Y_k}(x)$ of the function $\mathbf{u}_Y(x,\,y)$ are bounded for all k,

$$|\hat{\mathbf{u}}_{Y_k}(x)| < \infty. \tag{20}$$

Theorem 1 *The system of simple structure (12) may have at most one solution $\mathbf{u}(x,\,y)\in U$, and system (17) may have at most one solution $\mathbf{u}_Y(x,\,y)\in U_Y$.*

Proof. Clearly, it is sufficient to show that the homogeneous system

$$\frac{\partial \mathbf{u}}{\partial x} = \mathbf{A}\left(\frac{\partial}{\partial y}\right)\mathbf{u}$$

has only trivial solution $\mathbf{u}(x,\,y)\equiv\mathbf{0}$ in the class U. Let $\mathbf{u}(x,\,y)\in U$ be some solution to this homogeneous system. We represent this solution as a Fourier integral in accordance with (14a) and, using (15), obtain

$$\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\left[\frac{d}{dx}\hat{\mathbf{u}}(x,\,\alpha) - \mathbf{A}(i\alpha)\hat{\mathbf{u}}(x,\,\alpha)\right] e^{i\alpha y} d\alpha = \mathbf{0}.$$

Therefore, for any α

$$\frac{d\hat{\mathbf{u}}(x,\,\alpha)}{dx} = \mathbf{A}(i\alpha)\hat{\mathbf{u}}(x,\,\alpha), \tag{21}$$

and $\hat{\mathbf{u}}(x,\,\alpha)$ is bounded. On the other hand, all roots of the characteristic equation of system (21) always have non-zero real parts since (12) is a system of simple structure. It is well-known that in this case system (21) has a unique bounded solution $\hat{\mathbf{u}}(x,\,\alpha)$, which is identically zero. Consequently,

$$\mathbf{u}(x,\,y) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty}\hat{\mathbf{u}}(x,\,\alpha)e^{i\alpha y}d\alpha \equiv \mathbf{0}.$$

The statement about $\mathbf{u}_Y(x,\,y)$ can be proven analogously. □

To formulate the next theorem, we introduce the following notations. Let $\varphi(x,\,y)$ be some n-component vector-function, $\varphi(x,\,y) = (\varphi_1(x,\,y),\,\varphi_2(x,\,y),\,\ldots,\,\varphi_n(x,\,y))^T$. Designate

$$|\varphi(x,\,y)| = \sum_{j=1}^{n}|\varphi_j(x,\,y)|; \tag{22a}$$

$$\left|\frac{\partial^{p+q}\varphi(x,\,y)}{\partial x^p \partial y^q}\right| = \sum_{j=1}^{n}\left|\frac{\partial^{p+q}\varphi_j(x,\,y)}{\partial x^p \partial y^q}\right|; \tag{22b}$$

$$\|\varphi(x,\,y)\|_k = \sum_{p+q=0}^{k}\sup_{x,\,y}\left|\frac{\partial^{p+q}\varphi(x,\,y)}{\partial x^p \partial y^q}\right|. \tag{22c}$$

Theorem 2 *Consider an arbitrary bounded domain $D\subset R^2$. Let z be an arbitrary non-negative integer number. Then, one always can find a sufficiently large number $k = k(z,\,D)$ such that for any compactly supported function $\mathbf{f}(x,\,y)$, $\mathbf{f}(x,\,y)\equiv\mathbf{0}$ for $|x-a|>a$ and $|y|>a$, that satisfies $\|\mathbf{f}\|_k<\infty$ (see (22)) the solutions $\mathbf{u}\in U$ and $\mathbf{u}_Y\in U_Y$ of systems (12) and (17), respectively, exist ($Y>2a$ is arbitrary) and*

$$\|\mathbf{u}-\mathbf{u}_Y\|_{z,\,D} \le \frac{const}{Y^\kappa}, \tag{23}$$

where $\kappa = \kappa(z,\,k,\,D)$, and $\kappa\longrightarrow\infty$ as $k\longrightarrow\infty$.

The importance of Theorem 2 is that it allows us to approximate the solution of the infinite-domain AP by the solutions of another problem, which is periodic and, therefore, finite, in all but one Cartesian directions. Recall, to calculate the ABC's we need to know the projection $\mathbf{P}_\Gamma$ (see (8)), which is the superposition of the trace $\mathbf{Tr}_\Gamma$ (see (6)) and the potential $\mathbf{P}$ (see (5)). We, therefore, conclude that

in order to obtain the ABC's we do not need to know the potential, i.e., the solution of the AP, everywhere on D_{ex}; we need to know it only on some neighborhood of , . Consequently, the approximation of $\mathbf{u}$ by $\mathbf{u}_Y$ on any finite domain D as the period Y increases (see (23)) is sufficient for achieving our goal of constructing the ABC's.

The proof of Theorem 2 is essentially based on the estimates for one-dimensional fundamental solutions obtained by Godunov and Gordienko in [16]. Specifically, let

$$\frac{d\mathbf{v}}{dx} = \mathbf{A}\,\mathbf{v} + \mathbf{f}(x) \tag{24}$$

be an abstract system of n ordinary differential equations (ODE's) with constant coefficients, and $\mathbf{f}(x)$ be a continuous bounded function. Let also the roots λ of the characteristic equation $\det|\mathbf{A} - \lambda\mathbf{I}| = 0$ of system (24) satisfy the inequality

$$|\Re\lambda| \geq \delta > 0. \tag{25}$$

We supplement system (24) by the condition of boundedness of its solution $\mathbf{v}(x) = (v_1(x),\, v_2(x), \ldots, v_n(x))^T$ on the entire line $-\infty < x < \infty$,

$$\|\mathbf{v}(x)\| \leq const. \tag{26}$$

Because of the equivalence of norms on a finite-dimensional space for a fixed n, the choice of the specific norm $\|\mathbf{v}(x)\|$ in (26) is not essential. Hereafter, we will always use the l_1 norm of vectors, i.e., consider as a norm of a vector the sum of absolute values of its components. We now formulate the following result, see [16]. Fundamental solution $\mathbf{G}(x)$ of the corresponding linear differential operator from (24) exists and is unique in the class of functions that meet boundary condition (26). This fundamental solution is actually a $n \times n$ matrix that satisfy the inequality

$$\|\mathbf{G}(x)\| \leq \Omega(n) \left(\frac{\|\mathbf{A}\|}{\delta}\right)^{n-1} e^{-\delta\frac{|x|}{2}}, \tag{27}$$

where $\Omega(n)$ is some number that depends only on n, and the norms of the matrices $\mathbf{G}(x)$ and $\mathbf{A}$ are consistent with the chosen vector norm. Note, according to the definition of the fundamental solution, one can represent the solution to (24), (26) as

$$\mathbf{v}(x) = \int_{-\infty}^{\infty} \mathbf{G}(x-t)\mathbf{f}(t)dt. \tag{28}$$

We will be looking for the solution of system (12) in the form of the Fourier integral (14a), where the Fourier transformation $\hat{\mathbf{u}}(x, \alpha)$ is subject to the determination. Analogously to (14), we represent the RHS $\mathbf{f}(x,\, y)$ of system (12) as

$$\mathbf{f}(x,\, y) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{\mathbf{f}}(x,\, \alpha) e^{i\alpha y} d\alpha, \tag{29a}$$

$$\hat{\mathbf{f}}(x,\, \alpha) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \mathbf{f}(x,\, y) e^{-i\alpha y} dy. \tag{29b}$$

Substituting (14a) and (29a) into (12) and using (15), we formally obtain for $\hat{\mathbf{u}}(x, \alpha)$ the following system of ODE's with constant coefficients

$$\frac{d\hat{\mathbf{u}}(x,\, \alpha)}{dx} = \mathbf{A}(i\alpha)\hat{\mathbf{u}}(x,\, \alpha) + \hat{\mathbf{f}}(x, \alpha). \tag{30}$$

System (30) depends on α as on the parameter. We will supplement this system by boundary condition (16).

To make sure that the function $\mathbf{u}(x,\, y)$ obtained using (14a) (provided that $\hat{\mathbf{u}}(x, \alpha)$ satisfies (30), (16)) does belong to U and does solve (12), we will first have to make certain assumptions in regard to the smoothness of the RHS $\mathbf{f}(x,\, y)$ (recall, this function is always supposed to have compact support). Then, we estimate from above the Fourier transformation $\hat{\mathbf{f}}(x, \alpha)$ (see (29a)) and its derivatives by certain rational functions of α. Using the estimates for $\hat{\mathbf{f}}(x, \alpha)$, inequality (27), and representation (28), we can estimate from above the solution $\hat{\mathbf{u}}(x, \alpha)$ of (30), (16) and its derivatives with respect to x by the (22)-type norm of $\mathbf{f}(x,\, y)$ multiplied by a certain rational function of α and a certain exponential function of x. Further, the differentiability of $\hat{\mathbf{u}}(x, \alpha)$ with respect to α and the estimates for the derivatives (in particular, the rate of decay for big $|\alpha|$) can be obtained by induction with respect to the order of differentiation. Finally, using the aforementioned estimates we make sure that the inverse Fourier transformation (14a) does exist in U and therefore, provides for the solution to (12); using the same estimates, we also make sure that inequality (23) does hold, first for $z = 0$ and then for $z > 0$.

This brief outline will be transformed into the full proof of Theorem 2 in a future paper. Here, we will show how to handle the last remaining infinite direction, x, and to therefore finally replace the infinite-domain AP by a new problem formulated on some bounded domain.

Since we have already replaced the original infinite-domain formulation of the AP by the periodic formulation with respect to y, then, instead of solving systems (30), (16) for each α, $-\infty < \alpha < \infty$, we will need to solve the systems

$$\frac{d}{dx}\hat{\mathbf{u}}_{Y_k}(x) = \mathbf{A}\left(i\frac{2\pi k}{Y}\right)\hat{\mathbf{u}}_{Y_k}(x) + \hat{\mathbf{f}}_{Y_k}(x) \tag{31}$$

with boundary condition (20) for each k, $k = 0, \pm 1, \pm 2, \ldots$ Note, system (31) is obtained on the basis of representation

$$\mathbf{f}_Y(x,\, y) = \sum_{k=-\infty}^{k=\infty} \hat{\mathbf{f}}_{Y_k}(x) e^{iky\frac{2\pi}{Y}}, \tag{32a}$$

$$\hat{\mathbf{f}}_{Y_k}(x) = \frac{1}{Y}\int_{-Y/2}^{Y/2} \mathbf{f}_Y(x,\, y) e^{-iky\frac{2\pi}{Y}} dy \tag{32b}$$

by substituting (18a) and (32a) into (17) and using (19).

For each k, $k = 0, \pm 1, \pm 2, \ldots$, system (31) formally needs to be solved on the entire line $-\infty < x < \infty$. However, since we need to know the solution of the AP only on some finite neighborhood of , , we can always choose a sufficiently large but still finite segment on the x axis, on which it would be sufficient to calculate the solution to (31). Without loss of generality, we may think that it is the segment $(0,\, X)$, $X > 2a$. Since the RHS $\mathbf{f}(x,\, y)$ is compactly supported, system (31) is homogeneous for $x \le 0$ and $x \ge X$. To ensure that boundary condition (20) is met, we have to prohibit for $x \ge X$ all the eigensolutions of the homogeneous counterpart to (31) that increase as $x \longrightarrow +\infty$ and to prohibit for $x \le 0$ all the eigensolutions that increase as $x \longrightarrow -\infty$. This can be done by imposing the boundary conditions

$$\left[\prod_{\Re\lambda_j(k)>0} \left(\mathbf{A}^k - \lambda_j(k)\mathbf{I}\right)\right]\hat{\mathbf{u}}_{Y_k}(0) = \mathbf{0} \tag{33a}$$

and

$$\left[\prod_{\Re\lambda_j(k)<0} \left(\mathbf{A}^k - \lambda_j(k)\mathbf{I}\right)\right]\hat{\mathbf{u}}_{Y_k}(X) = \mathbf{0} \tag{33b}$$

at $x = 0$ and $x = X$, respectively. In (33), $\lambda_j(k)$ are the eigenvalues of $\mathbf{A}^k \equiv \mathbf{A}\left(i\frac{2\pi k}{Y}\right)$ and the matrix products are computed taking into account the multiplicities of these eigenvalues. Clearly, since for the system of simple structure $|\Re\lambda| \ge \delta > 0$, boundary conditions (33) actually imply that the solution $\hat{\mathbf{u}}_{Y_k}(x)$ vanishes at infinity (which is a stronger property than simply boundedness (20)).

Thus, we have shown how to replace the infinite-domain AP for a system of simple structure by the new problem formulated on the rectangle $(0,\, X) \times (-Y/2,\, Y/2)$ for the same RHS so that the solutions of these two problems are arbitrarily close to one another on a fixed finite neighborhood of D_{in}. We only have to mention, that the "asymmetric treatment" of the Cartesian directions x and y is caused mostly by the physical concerns. Specifically, typical external flow formulations (see Section 2) would bear some natural non-isotropy when we treat the Cartesian direction x as a streamwise and the Cartesian direction y as a cross-stream. In so doing, exact analytic treatment (33) of the Cartesian direction x seems to be most relevant for practical computations, since the periodic treatment for x may result in much larger values of period required for achieving the same accuracy than the periodic treatment for y. On the other hand, for the purposes other than the practical computation of projections, we may consider Fourier representation of the solution to the AP in all Cartesian directions. In particular, this has been done by Tsynkov in [17] when proving the solvability of the AP for the linearized thin-layer equations in the sense of tempered distributions.

To conclude the introductory part of the paper, we will also briefly comment on the relation between the systems of simple structure and other equations that can often be encountered in practice.

1.5 Wider Classes of Systems

Many systems of equations that are frequently solved in mathematical physics may not appear to be the systems of simple structure. For example, consider the system

$$\begin{aligned} &\frac{\partial u}{\partial x} = v \\ &\frac{\partial v}{\partial x} = -\frac{\partial^2 u}{\partial y^2} + \mu u + f(x,\, y), \\ &\mu \in R, \quad \mu = \text{const}, \end{aligned} \tag{34}$$

which is obtained when replacing the equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \mu u = f(x,\, y) \tag{35}$$

by the first-order system with respect to x. For (34), the matrix $\mathbf{A}(i\alpha)$ becomes

$$\mathbf{A}(i\alpha) = \begin{bmatrix} 0 & 1 \\ \alpha^2 + \mu & 0 \end{bmatrix},$$

and its eigenvalues are $\lambda(\alpha) = \pm\sqrt{\alpha^2 + \mu}$. When $\mu > 0$ we have $|\Re\lambda(\alpha)| \ge \sqrt{\mu} > 0$, and system

(34) is a system of simple structure. However, when $\mu = 0$ (in this case, equation (35) is the Poisson equation) or when $\mu < 0$ (equation (35) is the Helmholtz equation), the real part of $\lambda(\alpha)$ for some α's is zero, which means that system (34) is not a system of simple structure. Accordingly, the Poisson equation on R^2 not always has a bounded solution. As for the Helmholtz equation, the condition of vanishing of its solution as $r \equiv (x^2 + y^2)^{1/2} \longrightarrow \infty$ does not ensure the uniqueness. To select the unique solution, one has to impose a more fine Sommerfeld radiation condition, which, in particular, also implies vanishing of the solution at infinity.

As shown above, if the linear differential operator $\mathbf{L}$ from (2b) being represented in the form of (12) implies the inequality (13), i.e., the corresponding system of PDE's meets the Definition 1 of the systems of simple structure, and if the boundary conditions at infinity (2c) are actually the conditions of a sufficiently fast decay, then the ABC's at the artificial boundary , $= \partial D_{in}$ can be constructed using the finite-domain AP on the rectangle $(0,\, X) \times (-Y/2,\, Y/2)$. Indeed, under these conditions the infinite-domain AP (4) formulated for the RHS (7) can be replaced by the finite-domain AP using Theorem 2 and boundary conditions (33). If, however, system (2b) represented in the form of (12) is not a system of simple structure, then we can sometimes approximate its solutions by the solutions of a specially chosen system of simple structure. (Note, the representation in the form of (12), if possible at all, may require the solution with respect to $\partial \mathbf{u}/\partial x$ after the Fourier transform; in so doing, the entries of $\mathbf{A}(i\alpha)$ may become rational rather than polynomial functions of α).

Let $\mathbf{B} = \mathbf{B}\left(\dfrac{\partial}{\partial y},\, \varepsilon\right)$ be a $n \times n$ matrix, which approaches the zero matrix as $\varepsilon \longrightarrow +0$, and let the roots $\lambda(\alpha,\, \varepsilon)$ of the equation

$$\det \|\mathbf{A}(i\alpha) + \mathbf{B}(i\alpha,\, \varepsilon) - \lambda(\alpha,\, \varepsilon)\mathbf{I}\| = 0 \tag{36}$$

satisfy for each ε the inequality

$$|\Re\lambda(\alpha,\, \varepsilon)| \geq \delta(\varepsilon) > 0. \tag{37}$$

Let, in addition, the problem

$$\Phi(\mathbf{u},\, \nabla\mathbf{u},\, \ldots, \varepsilon) = \mathbf{g}(x,\, y), \quad (x,\, y) \in D_{in} \tag{38a}$$

$$\frac{\partial \mathbf{u}(x,\, y,\, \varepsilon)}{\partial x} = \left[\mathbf{A}\left(\frac{\partial}{\partial y}\right) + \mathbf{B}\left(\frac{\partial}{\partial y},\, \varepsilon\right)\right]\mathbf{u}, \tag{38b}$$

$$(x,\, y) \in \bar{D}_{ex}$$

$$\mathbf{u}(x,\, y,\, \varepsilon) \longrightarrow \mathbf{0}, \quad \text{as} \quad r \longrightarrow \infty \tag{38c}$$

be uniquely solvable and let its solution $\mathbf{u}(x,\, y,\, \varepsilon)$ approach the solution to the steady-state counterpart of problem (2) as $\varepsilon \longrightarrow +0$ on any finite subdomain of R^2. Then, for constructing the ABC's on , one obviously can use the AP

$$\frac{\partial \mathbf{u}(x,\, y,\, \varepsilon)}{\partial x} = \left[\mathbf{A}\left(\frac{\partial}{\partial y}\right) + \mathbf{B}\left(\frac{\partial}{\partial y},\, \varepsilon\right)\right]\mathbf{u} + \mathbf{f}(x,\, y), \tag{39a}$$

$$(x,\, y) \in R^2, \quad \text{supp}\mathbf{f}(x,\, y) \subset D_{in}$$

$$\mathbf{u}(x,\, y,\, \varepsilon) \longrightarrow \mathbf{0}, \quad \text{as} \quad r \longrightarrow \infty, \tag{39b}$$

where ε is chosen sufficiently small.

For example, if $\mu = 0$ in system (34), then the correction

$$\mathbf{B}\left(\frac{\partial}{\partial y},\, \varepsilon\right) = \begin{bmatrix} 0 & 0 \\ \varepsilon & 0 \end{bmatrix}.$$

will transform (34) into a system of simple structure. In the case of the Helmholtz equation ($\mu < 0$ in (34)) supplemented by the Sommerfeld radiation conditions at infinity, we choose

$$\mathbf{B}\left(\frac{\partial}{\partial y},\, \varepsilon\right) = \begin{bmatrix} 0 & 0 \\ i\varepsilon & 0 \end{bmatrix}.$$

This choice corresponds to the well-known principle of limitary absorption, which allows one to approximate both the Helmholtz equation and the Sommerfeld radiation conditions with the increasing accuracy as $\varepsilon \longrightarrow +0$. We should note that another choice of $\mathbf{B}$,

$$\mathbf{B}\left(\frac{\partial}{\partial y},\, \varepsilon\right) = \begin{bmatrix} 0 & 0 \\ -i\varepsilon & 0 \end{bmatrix}.$$

also provides for the system of simple structure (39a). However, in this case we approximate (as $\varepsilon \longrightarrow +0$) the solution composed of incoming waves rather than the one that meets the radiation conditions.

Finally, we mention that for a given system of PDE's it is, generally speaking, not always easy to find a good analogue to the principle of limitary absorption so that the solutions of this system can be approximated by the solutions of a certain system of simple structure. However, the replacement of an infinite-domain AP by the periodic problem may still be possible, the validity of such a replacement can be verified a posteriori by means of the numerical experiments as done, e.g., in our work [18].

2 ABC's for External Viscous Flows

2.1 Governing Equations

We consider a flow of compressible viscous fluid over a finite body or configuration of bodies (e.g., single-element or multi-element airfoil in two space dimensions). The flow is supposed to be uniform and subsonic at infinity. As mentioned above, we intend to actually calculate the flow only on some finite computational domain D_{in} (which contains the immersed body(ies)). The infinite exterior D_{ex} of the domain D_{in} is truncated, and the equations there along with the boundary conditions at infinity are to be replaced by the ABC's at , $= \partial D_{in}$.

To construct the ABC's, we linearize the governing equations in D_{ex} against the constant free-stream background. Generally, to justify the possibility of linearization we have to make sure that first, the flow perturbations in the far field are small, and second, the differential equations that describe these small perturbations are linear. As concerns the first assumption, it does hold for the external viscous flows, at least when the size of D_{in} is much larger than the size of the immersed body. The linearity of the governing equations for small perturbations is easy to establish for the subsonic flows, i.e., when the free-stream Mach number M_0 is not too close to one. For the transonic flows, it basically requires a special study (see our work [19] and also the recent paper [20]). The situation appears different for three space dimensions, when the far field is essentially linear, and for two space dimensions, when the consideration of a simplified potential model may formally require to introduce some nonlinear corrections in the transonic limit, i.e., as $M_0 \longrightarrow 1$. We, however, always consider the full flow system rather than the potential equation, we also never approach the transonic limit too closely. In so doing, we use the far-filed linearization for two space dimensions as well. In practice, the validity of the far-field linearization is always verified by an a posteriori numerical check.

Either one of the two following systems of PDE's

$$\frac{\partial \rho}{\partial x}+\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0 \tag{40}$$

$$\frac{\partial u}{\partial x}+\frac{\partial p}{\partial x}-\frac{1}{Re}\left[\frac{4}{3}\frac{\partial^2 u}{\partial x^2}+\frac{1}{3}\frac{\partial^2 v}{\partial x \partial y}+\frac{\partial^2 u}{\partial y^2}\right]=0$$

$$\frac{\partial v}{\partial x}+\frac{\partial p}{\partial y}-\frac{1}{Re}\left[\frac{4}{3}\frac{\partial^2 v}{\partial y^2}+\frac{1}{3}\frac{\partial^2 u}{\partial x \partial y}+\frac{\partial^2 v}{\partial x^2}\right]=0$$

$$\frac{\partial p}{\partial x}-\frac{1}{M_0^2}\frac{\partial \rho}{\partial x}-\frac{\gamma}{Re\,Pr}\left[\frac{\partial^2 p}{\partial x^2}+\frac{\partial^2 p}{\partial y^2}-\frac{1}{\gamma M_0^2}\left(\frac{\partial^2 \rho}{\partial x^2}+\frac{\partial^2 \rho}{\partial y^2}\right)\right]=0$$

and

$$\frac{\partial \rho}{\partial x}+\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0 \tag{41}$$

$$\frac{\partial u}{\partial x}+\frac{\partial p}{\partial x}-\frac{1}{Re}\frac{\partial^2 u}{\partial y^2}=0$$

$$\frac{\partial v}{\partial x}+\frac{\partial p}{\partial y}-\frac{1}{Re}\frac{4}{3}\frac{\partial^2 v}{\partial y^2}=0$$

$$\frac{\partial p}{\partial x}-\frac{1}{M_0^2}\frac{\partial \rho}{\partial x}-\frac{\gamma}{Re\,Pr}\left[\frac{\partial^2 p}{\partial y^2}-\frac{1}{\gamma M_0^2}\frac{\partial^2 \rho}{\partial y^2}\right]=0$$

can be used (and has actually been used) for constructing the ABC's in two space dimensions for steady-state flows. Hereafter, we will concentrate mostly on this specific case (2D steady-state flows); in the end of the paper, we will also present some recent three-dimensional results (see our work [19, 20], as well as the earlier papers [21, 22]) and comment on the possible approaches to treating the time-dependent problems.

System (40) represents the linearized dimensionless full Navier-Stokes equations, and system (41) represents the linearized dimensionless thin-layer equations. In (40) and (41), u, v, p, and ρ are the perturbations of the Cartesian velocity components, pressure, and density with respect to the corresponding free-stream parameters, M_0 is the free-stream Mach number, Re is the Reynolds number, Pr is the Prandtl number, and γ is the ratio of specific heats. In both cases we assume that the direction of flow at infinity coincides with the positive x direction, and that the gas is perfect. To obtain the dimensionless quantities, we use the following scales: u_0, for velocity components; ρ_0, for density; $\rho_0 {u_0}^2$, for pressure; μ_0, for viscosity; characteristic size L (typically, airfoil chord), for all distances. Here, the subscript "0" denotes the free-stream parameters.

As mentioned above, to construct the ABC's we need to be able to solve the AP for a nonhomogeneous counterpart to either (40) or (41) driven by a certain compactly supported RHS. In our work [17] we have, in particular, shown that when supplemented by a compactly supported RHS, system (41) is always solvable in the sense of tempered distributions (see, e.g., book [23] by Hörmander or [24] by Vladimirov for a detailed description of the concept of distributions and its application to solving the PDE's). We have also shown that if this solution satisfies the boundary condition

$$(u,\, v,\, p,\, \rho) \longrightarrow (0,\, 0,\, 0,\, 0), \tag{42}$$

$$\text{as} \quad r \equiv (x^2+y^2)^{1/2} \longrightarrow +\infty,$$

then it is unique in the class of distributions vanishing at infinity. Note, the solvability in $\mathcal{S}'$ (space of tempered distributions) does not necessarily mean the fulfillment of (42). To make sure condition (42) does hold, we need to do certain assumptions about the RHS $\mathbf{f}(x, y)$. From the very beginning, we always think that $\mathbf{f}$ is compactly supported and that $\mathbf{f} \in L^1(R^2)$, which is no loss of generality. If we additionally assume that $\mathbf{f} \in L^2(R^2)$ (which basically presents no loss of generality as well), then the solution $\mathbf{u} \equiv (u, v, p, \rho)$ to the infinite-domain AP (of type (4)) can be represented as $\mathbf{u} = \mathbf{u}^{(1)} + \mathbf{u}^{(2)}$, where $\mathbf{u}^{(1)}$ satisfies (42) and $\mathbf{u}^{(2)} \in L^2(R^2)$; the latter inclusion can be treated as "a generalized decay at infinity". If, however, we require that $\mathbf{f}(x, y)$ be sufficiently smooth on R^2 so that its Fourier transformation with respect to both x and y belongs to $L^1(R^2)$, then we can show (see [17]) that the solution $\mathbf{u}$ to the AP for system (41) meets boundary condition (42). Similar results for three space dimensions have been obtained in our work [20]. Let us also note that the work [17] is devoted to studying a wider class of formulations than only the steady-state flows, specifically, we study there the flows that oscillate in time. The aforementioned solvability results for the steady-state thin-layer equations can be obtained as one consequence of the considerations of [17]. We will briefly review the general results of [17] in the end of this paper.

2.2 Geometric Setup

Let us now introduce the geometric setup typical for external flow problems in two space dimensions; an example is shown in Figure 1. We are interested in calculating the flow around an airfoil, the single-element configuration presented on the figure does not imply any loss of generality since the treatment of external boundary for the multiple immersed bodies would basically be the same. To calculate the flow, we first generate the grid around the airfoil; for this specific example it will be a C-type curvilinear grid, which actually forms the computational domain D_{in}. The nonlinear flow equations are discretized and solved on this grid inside D_{in}. However, analogously to the continuous case (see Section 1) the discrete system inside D_{in} is subdefinite unless we supplement it by some ABC's. Indeed, the stencil of the finite-difference operator used inside D_{in} cannot, generally speaking, be applied to those nodes of the C-grid that are located near the external boundary, e.g., it cannot be applied to any node of the outermost coordinate row of this grid, since in so doing the part of the stencil may simply "fall out" of the domain. Therefore, the discrete system inside D_{in} without ABC's would merely have less equations than it has unknowns. Consequently, unlike the continuous case, for which to close the system inside D_{in} means to set the ABC's exactly at the continuous external boundary, to close the system inside D_{in} in the discrete framework means to provide for some additional relations between the values of the solution in the nodes located in a certain external part of the grid. For example, if the scheme employed inside D_{in} is written on the 3×3 stencil, which, in particular, corresponds to a widely used second-order central-difference approach, then the ABC's should provide for the missing relations between the values of the solution on the penultimate and outermost coordinate rows of the C-grid. On Figure 1, the penultimate coordinate row is designated , and the outermost row is designated , $_1$.

Henceforth, we will treat the penultimate coordinate row , of the C-grid as a formal continuous artificial boundary. This, in particular, means that the outermost curve , $_1$ belongs already to the area, in which we linearize the governing equations. It also means that if we were looking for the continuous solution on D_{in}, then we would need to construct the ABC's exactly at the penultimate coordinate line , . For the discrete formulation, however, we need to obtain missing relations between the values of the solution on , and , $_1$ (see above). To do that, we will first formulate the ABC's of type (9) exactly at , and then use the generalized potential (6) for complementing the boundary data from , to D_{ex}, the trace of this complement on , $_1$ will provide us with the unknown values of the solution at the outermost coordinate line of the C-grid. In so doing, we not only obtain the desirable missing relations that close the discrete system in D_{in}, but at the same time automatically make sure that these relations are right in the sense that they properly take into account the structure of the solution in the far field; the latter is true because the complement we construct on D_{ex} is obtained on the basis of (9).

2.3 Computation of the Potentials and Projections

In fact, we, of course, cannot calculate directly the continuous generalized potentials (6) and boundary projections (8); instead, we calculate their discrete counterparts called the difference potentials and the difference boundary projections, respectively. The corresponding numerical procedure is based on application of the DPM [7, 8]. The issues of consistency and convergence for the difference potentials and their continuous prototypes have been studied by Ryaben'kii in [8] and Reznik in [25].

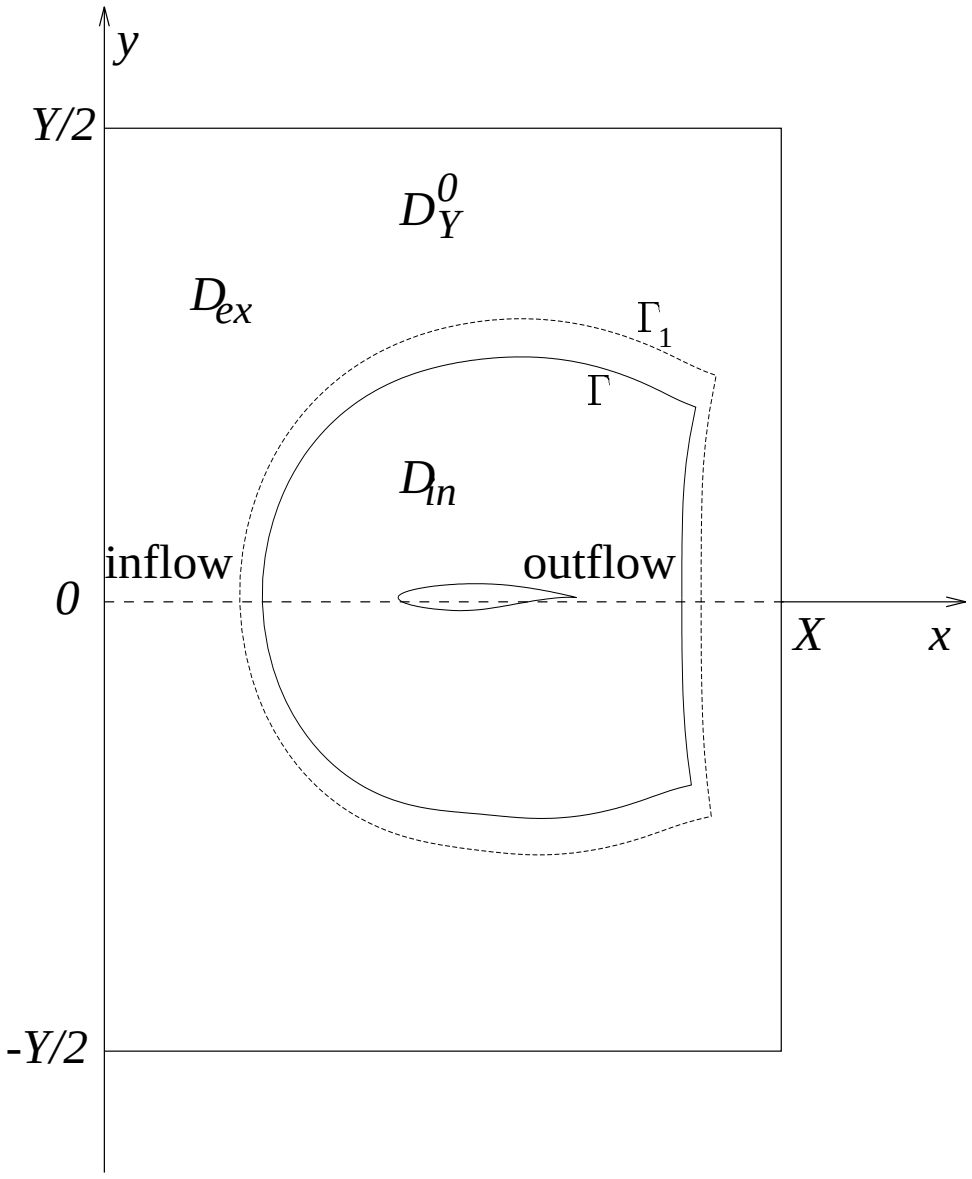

Figure 1. Configuration of domains.

As described in Section 1, the AP for the nonhomogeneous version of either system (40) or (41) is first formulated on the entire plane and then truncated so that one needs to solve it only on the rectangular domain $D_Y^0 = (0,\, X) \times (-Y/2,\, Y/2)$; this auxiliary domain should fully contain both , and , $_1$, see Figure 1. We will now describe the finite-difference formulation of the AP and the DPM-based algorithm for calculating the difference potentials and projections.

Let us introduce in D_Y^0 two Cartesian grids, $\mathcal{M}^0$ and $\mathcal{N}^0$. The grid $\mathcal{M}^0$ will be used for specifying the RHS for the finite-difference AP, the grid $\mathcal{N}^0$ is the one, on which the solution to this AP will be defined. We also introduce the space $F^0 \ni \mathbf{f}^0 \equiv \mathbf{f}_{\mathcal{M}^0}$ of the RHS's for the AP, the space $U^0 \ni \mathbf{u}^0 \equiv \mathbf{u}_{\mathcal{N}^0}$ of its solutions, and the finite-difference operator $\mathbf{L}_h : U^0 \longmapsto F^0$, which can be a discretization of the left-hand side of either (40) or (41). Note, in the work [18, 26, 27] we have used the operator $\mathbf{L}_h$ obtained by the second-order central-difference approximation of (40), in so doing the grids $\mathcal{M}^0$ and $\mathcal{N}^0$ coincide except on the lines $x = 0$ and $x = X$, where the RHS grid $\mathcal{M}^0$ is simply not defined. In a later work [17, 28, 29], we have used the operator $\mathbf{L}_h$ obtained by the second-order approximation of (41)

with the central differences along y and the first-order differences along x. In so doing, the RHS grid $\mathcal{M}^0$ is shifted in the x direction with respect to $\mathcal{N}^0$ by the half of the grid size, and the finite-difference equations of the AP are written as

$$\mathbf{L}_h\mathbf{u}^0 \equiv \mathbf{D}\frac{\mathbf{u}_{m+1,j}-\mathbf{u}_{m,j}}{h_x}+ \tag{43}$$

$$\frac{1}{2}\mathbf{F}\left(\frac{\mathbf{u}_{m,j+1}-\mathbf{u}_{m,j-1}}{2h_y}+\frac{\mathbf{u}_{m+1,j+1}-\mathbf{u}_{m+1,j-1}}{2h_y}\right)+$$

$$\frac{1}{2}\mathbf{H}\left(\frac{\mathbf{u}_{m,j+1}-2\mathbf{u}_{m,j}+\mathbf{u}_{m,j-1}}{h_y^2}+\right.$$

$$\left.\frac{\mathbf{u}_{m+1,j+1}-2\mathbf{u}_{m+1,j}+\mathbf{u}_{m+1,j-1}}{h_y^2}\right)=\mathbf{f}_{m+1/2,j},$$

$$m=0,\dots,M-1,\quad j=0,\dots,2J,$$

where h_x and h_y are the Cartesian grid sizes; subscript m corresponds to the Cartesian direction x, and $M+1$ is the total number of nodes of the grid $\mathcal{N}^0$ in this direction; subscript j corresponds to the Cartesian direction y, and $2J+2$ is the total number of nodes of the grid $\mathcal{N}^0$ (and $\mathcal{M}^0$) in this direction. The 4×4 matrices $\mathbf{D}$, $\mathbf{F}$, and $\mathbf{H}$ are given by

$$\mathbf{D}=\begin{bmatrix}1&0&0&1\\1&0&1&0\\0&1&0&0\\0&0&1&-{M_0}^{-2}\end{bmatrix}, \tag{44}$$

$$\mathbf{F}=\begin{bmatrix}0&1&0&0\\0&0&0&0\\0&0&1&0\\0&0&0&0\end{bmatrix},$$

$$\mathbf{H}=-\frac{1}{Re}\begin{bmatrix}0&0&0&0\\1&0&0&0\\0&4/3&0&0\\0&0&\gamma Pr^{-1}&-Pr^{-1}{M_0}^{-2}\end{bmatrix}.$$

Note, we omit the superscript "0" for the solutions and the RHS's of the AP when referring these functions to specific grid nodes, see (43).

Following the considerations of Section 1, we require that any function $\mathbf{u}^0\in U^0$ be periodic in the y direction with the period Y, i.e.,

$$\mathbf{u}_{m,0}=\mathbf{u}_{m,2J+1},\quad m=0,\dots,M, \tag{45}$$

$$\mathbf{u}_{m,-1}=\mathbf{u}_{m,2J},\quad m=0,\dots,M.$$

As concerns the RHS's $\mathbf{f}^0\in F^0$, they may, generally speaking, differ from zero only for those nodes of the grid $\mathcal{M}^0$ that belong to D_{in}. Instead of (18) and (32), we now use the standard discrete Fourier transform (see, e.g., [17, 18, 26] for details), and obtain the following family of systems of ordinary difference equations

$$\mathbf{A}_k\hat{\mathbf{u}}_{m+1,k}+\mathbf{B}_k\hat{\mathbf{u}}_{m,k}=\hat{\mathbf{f}}_{m+1/2,k}\,, \tag{46}$$

$$m=0,\dots,M-1,\ k=-J,\dots,J,$$

which represent the finite-difference analogue of equations (31). In system (46), subscript k is the dual discrete Fourier variable that corresponds to the Cartesian direction y, $\hat{\mathbf{u}}_{m+1,k}$ and $\hat{\mathbf{f}}_{m+1/2,k}$ are the discrete Fourier transformations of the solution and the RHS, respectively; $\mathbf{A}_k$ and $\mathbf{B}_k$ are the square coefficient matrices, which depend on k but do not depend on m. These matrices are obviously determined by the structure of the finite-difference operator $\mathbf{L}_h$. For system (43), $\mathbf{A}_k$ and $\mathbf{B}_k$ are given by

$$\mathbf{A}_k=\frac{1}{h_x}\mathbf{D}+\frac{r_k}{2}\mathbf{F}+\frac{t_k}{2}\mathbf{H}, \tag{47}$$

$$\mathbf{B}_k=-\frac{1}{h_x}\mathbf{D}+\frac{r_k}{2}\mathbf{F}+\frac{t_k}{2}\mathbf{H},$$

where the matrices $\mathbf{D}$, $\mathbf{F}$, and $\mathbf{H}$ are defined in formula (44), $r_k=i\sin\left(kh_y\frac{2\pi}{Y}\right)/h_y$, and $t_k=-4\sin^2\left(\frac{1}{2}kh_y\frac{2\pi}{Y}\right)/h_y^2$. Specific expressions for $\mathbf{A}_k$ and $\mathbf{B}_k$ that correspond to system (40) can be found in [18] and those that correspond to a more general case of time-periodic flows can be found in [17]. Note, the matrices $\mathbf{A}_k$ and $\mathbf{B}_k$ from (47) have order 4, whereas for system (40) these matrices would have order 8 (see [18]), this is caused by the necessity to introduce additional variables when reducing the order of differencing with respect to x from the second to the first.

To complete the formulation of the finite-difference AP on the rectangle D_Y^0, we have to specify the boundary conditions at $x=0$ and $x=X$, i.e., at $m=0$ and $m=M$. Analogously to how it is done in Section 1 for the continuous formulation, we first note that system (46) formally considered on the infinite mesh $-\infty<m<\infty$ is homogeneous for $m\le 0$ and $m\ge M$. Then, to meet boundary condition (42), we have to prohibit all the growing eigensolutions of (46) for $m\longrightarrow-\infty$, as well as for $m\longrightarrow+\infty$. This can be done by imposing for each k, $k=-J,\dots,J$, the boundary conditions

$$\left[\prod_{|\lambda_r(k)|>1}(\mathbf{Q}_k-\lambda_r(k)\mathbf{I})\right]\hat{\mathbf{u}}_{0,k}=\mathbf{0}, \tag{48a}$$

$$\left[\prod_{|\lambda_r(k)|\le 1}(\mathbf{Q}_k-\lambda_r(k)\mathbf{I})\right]\hat{\mathbf{u}}_{M,k}=\mathbf{0}, \tag{48b}$$

where $\mathbf{Q}_k=(\mathbf{A}_k)^{-1}\mathbf{B}_k$, $\lambda_r(k)$ are the eigenvalues of $\mathbf{Q}_k$, $\mathbf{I}$ is the identity matrix, and the matrix products are calculated in accordance with the multiplicities of eigenvalues. Discrete boundary conditions (48) are analogous to the continuous boundary conditions (33). The only difference is that unlike (33a) and (33b), equations (48a) and (48b) are not symmetric. In formula (48b) (downstream boundary condition), we admit the eigenvalues $|\lambda(k)|=1$, which in the continuous case would correspond to $|\Re\lambda(\alpha)|=0$. The reason for this asymmetry is that none of the systems (40) and (41) is actually a system of simple structure. For $\alpha=0$ ($k=0$ in the discrete formulation), we have (multiple) eigenvalue with $|\Re\lambda(\alpha)|=0$ ($|\lambda(k)|=1$) for both (40) and (41). Unfortunately, neither for the linearized Navier-Stokes nor for the linearized thin-layer equations we are unaware of any good analogues of the principle of limitary absorption (see Section 1) that could have helped us to approximate the solutions of these systems by the solutions of systems of simple structure. We have, therefore, to entirely rely on the numerical experiments as the means to justify the possibility to replace the original AP by the periodic one. The computations that do corroborate the possibility to introduce the periodic formulation are reported in [18]. Moreover, it is also possible to make sure that the matrix $\mathbf{Q}_k$ for $|\lambda(k)|=1$ ($k=0$) still has a full system of eigenvectors, which means that the corresponding eigensolutions of the homogeneous one-dimensional system are at most oscillatory, but never increasing. Using this fact, one can show (see [18]) that the resulting solution obtained by means of an inverse Fourier transform still vanishes at infinity, even if for a selected finite number of α's (k's) we require only the boundedness (48b) rather than the true decay of the one-dimensional solution in the Fourier space.

Thus, we have finally completed the formulation of the finite-difference AP, and have, therefore, defined its Green (i.e., inverse) operator $\mathbf{G}_h$, $\mathbf{G}_h: F^0\longmapsto U^0$. It is easy to see that this AP is uniquely solvable for any compactly supported RHS and well-posed, the well-posedness can be established on the basis of the considerations of [30]. An effective numerical algorithm for solving the AP (more precisely, for solving one-dimensional systems (46) with boundary conditions (48)) is described in our work [31]. This algorithm can be referred to as a version of the well-known successive substitution technique, but without its "inverse" or "resolving stage". The particular efficacy of the approach of [31] is based on the fact that boundary conditions (48) are formulated in terms of eigen subspaces of the operator $\mathbf{Q}_k$.

Let us now split the nodes of the grid $\mathcal{M}^0$ into two groups, $\mathcal{M}_{in}=\mathcal{M}^0\bigcap\bar{D}_{in}$ and $\mathcal{M}_{ex}=\mathcal{M}^0\backslash\mathcal{M}_{in}$. Then, we apply the stencil of the finite-difference operator $\mathbf{L}_h$ to each node of the set $\mathcal{M}_{in}$ and call the union of all these stencils $\mathcal{N}_{in}$. Analogously, we apply the stencil of $\mathbf{L}_h$ to each node of $\mathcal{M}_{ex}$ and obtain $\mathcal{N}_{ex}$. The intersection of the two sets $\mathcal{N}_{in}$ and $\mathcal{N}_{ex}$ is called the grid boundary γ, $\gamma=\mathcal{N}_{in}\bigcap\mathcal{N}_{ex}$. The subset $\gamma\subset\mathcal{N}^0$ is actually a multi-layered fringe composed of those nodes of the grid $\mathcal{N}^0$ that are located in a certain sense near the continuous artificial boundary , .

For any function $\mathbf{u}^0\in U^0$ we introduce its difference clear trace ξ_γ on the grid boundary γ as merely a contraction,

$$\xi_\gamma=\mathbf{Tr}_\gamma\mathbf{u}^0\equiv\mathbf{u}_{\mathcal{N}^0}\big|_\gamma. \tag{49}$$

Since γ is a multi-layered set of nodes, ξ_γ of (49) in a certain sense models ξ_Γ of (5). Let us now introduce the space of difference clear traces that would contain all grid vector-functions of dimension 4 defined on γ. For each element ξ_γ of this space, we can construct the generalized difference potential

$$\mathbf{P}_{ex}\xi_\gamma=\Big(\mathbf{G}_h\Big(\big(\mathbf{L}_h\mathbf{w}^0\big)\big|_{\mathcal{M}_{in}}\Big)\Big)\Big|_{\mathcal{N}_{ex}}, \tag{50}$$

where $\mathbf{w}^0\in U^0$ is chosen so that it has the trace ξ_γ, $\mathbf{Tr}_\gamma\mathbf{w}^0=\xi_\gamma$, and is arbitrary in the rest. For example, one always may choose $\mathbf{w}^0=\begin{cases}\xi_\gamma, & \text{on }\gamma\\ \mathbf{0}, & \text{on }\mathcal{N}^0\backslash\gamma\end{cases}$. As concerns the RHS for the difference AP, i.e., the function that the discrete Green operator $\mathbf{G}_h$ operates on in equation (50), it is defined as

$$\big(\mathbf{L}_h\mathbf{w}^0\big)\big|_{\mathcal{M}_{in}}=\begin{cases}\mathbf{L}_h\mathbf{w}^0, & \text{on }\mathcal{M}_{in},\\ \mathbf{0}, & \text{on }\mathcal{M}_{ex}.\end{cases} \tag{51}$$

Clearly, to calculate the difference potential $\mathbf{P}_{ex}\xi_\gamma$ (50), which is analogous to the continuous generalized potential (6), we need to actually solve the difference AP.

Finally, we define the operator $\mathbf{P}_\gamma$,

$$\mathbf{P}_\gamma\xi_\gamma=\mathbf{Tr}_\gamma\mathbf{P}_{ex}\xi_\gamma, \tag{52}$$

as the composition of potential (50) and trace (49). This operator obviously maps the space of difference clear traces ξ_γ onto itself. As in the continuous case (compare (52) with (8)), $\mathbf{P}_\gamma$ appears to be a projection, $\mathbf{P}_\gamma^2=\mathbf{P}_\gamma$, it is called the difference boundary

projection. It is possible to show [7, 8] that those and only those ξ_γ that satisfy the BEP

$$\mathbf{P}_\gamma \xi_\gamma = \xi_\gamma, \tag{53}$$

i.e., belong to the image of the boundary projection $\mathbf{P}_\gamma$, $\xi_\gamma \in \mathrm{Im}\mathbf{P}_\gamma$, can be complemented on $\mathcal{N}_{ex}$ so that the complement $\mathbf{u}_{\mathcal{N}_{ex}}$ satisfies the homogeneous equation $\mathbf{L}_h \mathbf{u}_{\mathcal{N}_{ex}} = \mathbf{0}_{\mathcal{M}_{ex}}$ and boundary conditions (45), (48) of the difference AP. In other words, BEP (53) provides for an exhaustive classification of those and only those ξ_γ's that can be represented as a trace of some solution $\mathbf{u}_{\mathcal{N}_{ex}}$ of the equation $\mathbf{L}_h \mathbf{u}_{\mathcal{N}_{ex}} = \mathbf{0}_{\mathcal{M}_{ex}}$ supplemented by boundary conditions (45), (48). Provided that the BEP (53) is satisfied, the aforementioned complement $\mathbf{u}_{\mathcal{N}_{ex}}$ can be obtained in the form of the generalized potential (50), $\mathbf{u}_{\mathcal{N}_{ex}} = \mathbf{P}_{ex}\xi_\gamma$. Difference boundary projection (52) and difference BEP (53) are analogous to the continuous boundary projection (8) and continuous BEP (9), respectively.

Recall, we have formulated the periodic AP so that its solution approaches the solution of the infinite-domain AP on any finite neighborhood of , as the period Y increases. This gives us grounds (see Section 1) for setting the continuous ABC's in the form of BEP (9), in which the Calderon projection $\mathbf{P}_\Gamma$ is calculated using the new finite-domain rather than the original infinite-domain AP. Here, we, in turn, approximate the solution of the continuous periodic AP by the solution of the finite-difference periodic AP, which presents the next key step of the entire procedure (we mean the next one after introducing the periodic formulation). This two-step scheme leads us to a somewhat non-standard concept of convergence for the solutions of the difference AP. Namely, we will consider convergence of the difference solution to the solution of the infinite-domain continuous AP on some finite fixed domain (e.g., on any rectangle $|x - a| < a$, $|y| < a$, where $a \le X/2$, $a < Y/2$) as $(h_x, h_y, Y) \longrightarrow (0, 0, +\infty)$. We have already discussed the reasons and consequences of considering the convergence on a fixed finite subdomain only. We should also emphasize that the convergence is considered not only as the grid size vanishes but also as the period Y synchronously grows. Note, to achieve some initially prescribed accuracy, one should increase the period and decrease the grid size consistently. Some estimates connecting the grid size, the period, and the desired accuracy can be found in [18].

Following the considerations of Section 1, one can conclude that the convergence of the foregoing type is sufficient for the purpose of constructing the ABC's. We will, therefore, use the difference analogues (50), (52), and (53) to the continues potentials (6), projections (8), and BEP's (9), respectively, to set the ABC's in the discrete framework. Later on, we will comment on how to choose the specific values of the grid size and the period; in practice, this choice is always done on the basis of the numerical experience.

2.4 An Application of the Difference Potentials and BEP's for Setting the ABC's

Let us denote by ν the set of those nodes of the C-grid that actually determine the penultimate coordinate line , (see Figure 1); analogously, nodes ν_1 will correspond to , $_1$. Additionally, we introduce on , the set of collocation points ω, which is also called the collocation grid. This collocation grid will be used for specifying the unknowns; typically, it is coarser than the grid ν. The size of the collocation grid ω is not arbitrary, it is connected to the size of the Cartesian grid $\mathcal{N}^0$, some relevant estimates can be found in [8]. For the practical purposes, we often take the total number $|\omega|$ of nodes of the collocation grid ω proportional to the square root of the total number $|\gamma|$ of nodes that constitute the grid boundary γ. We should also note that the collocation grid is usually not uniform; as a rule, it is more concentrated towards the wake region.

We will approximate the space of clear traces ξ_Γ (see Section 1) by the finite-dimensional space ξ_ω. Specifically, ξ_ω are the eight-component vector-functions defined at the nodes ω, the components of ξ_ω contain the trace of the solution $\mathbf{u}$ and the trace of its normal derivative $\dfrac{\partial \mathbf{u}}{\partial \zeta}$, $\xi_\omega = \left(\mathbf{u}, \dfrac{\partial \mathbf{u}}{\partial \zeta}\right)\Big|_\omega$. We assume that there is an operation $\mathbf{R}_{\Gamma\omega}$ of interpolation along , so that as the collocation grid ω is refined the continuous function $\mathbf{R}_{\Gamma\omega}\xi_\omega$ approaches the corresponding ξ_Γ in the sense of a sufficiently strong norm.

We also introduce the operation of continuation of the boundary data from the continuous artificial boundary , to the grid boundary γ. Assuming that the size of the Cartesian grid $\mathcal{N}^0$ is reasonably small, one can say that all the nodes γ are located in some small neighborhood of , . Therefore, considering ξ_Γ as the given data, we can drop normal from each node γ to , and then use the first two terms of the Taylor expansion to calculate ξ_γ. We will designate this operation of continuation by $\pi_{\gamma\Gamma}$, $\pi_{\gamma\Gamma}\xi_\Gamma = \xi_\gamma$. Combining $\pi_{\gamma\Gamma}$ with the previously introduced interpolation $\mathbf{R}_{\Gamma\omega}$, we obtain the operation of continuation of the discrete data ξ_ω from , to γ, $\pi_{\gamma\omega}\xi_\omega \equiv \pi_{\gamma\Gamma}\mathbf{R}_{\Gamma\omega}\xi_\omega = \xi_\gamma$.

Finally, recall that the actual data on the curve , is calculated numerically at the nodes ν. Therefore, we will further need the operation $\mathbf{R}_{\omega\nu}$ of one-dimensional interpolation along , that for a specified $\xi_\nu = \left(\mathbf{u},\, \dfrac{\partial \mathbf{u}}{\partial \zeta}\right)\Big|_\nu$ would give ξ_ω, $\xi_\omega = \mathbf{R}_{\omega\nu}\xi_\nu$. Since both ξ_ω and ξ_ν are vector-functions of the same dimension, the interpolation $\mathbf{R}_{\omega\nu}$ is implemented componentwise.

The finite-difference boundary projection $\mathbf{P}_\gamma$ of (52) and BEP (53) can be used for setting the ABC's differently. We will begin with the brief description of the algorithm of [18], which chronologically has come first.

Substituting the expression $\xi_\gamma = \pi_{\gamma\omega}\xi_\omega$ into the BEP (53), we obtain the following equation

$$(\mathbf{I}_\gamma - \mathbf{P}_\gamma)\, \pi_{\gamma\omega}\xi_\omega = \mathbf{0}_\omega \tag{54}$$

with respect to ξ_ω. Equation (54) can be treated as a certain implicit relation between the solution $\mathbf{u}$ and its normal derivative $\dfrac{\partial \mathbf{u}}{\partial \zeta}$ at the nodes ω. For the difference boundary projection $\mathbf{P}_\gamma$ obtained on the basis of the central-difference approximation of system (40), we actually solve equation (54) with respect to the normal derivatives $\dfrac{\partial \mathbf{u}}{\partial \zeta}\Big|_\omega$ (see [18]) and therefore express the normal derivatives explicitly in terms of $\mathbf{u}_\omega$. The method we employ for solving equation (54) is based on the application of a certain variational approach, see [18] for more details. Note that in the literature, the operators that express boundary values of normal derivatives of the solution to a PDE or system of PDE's in terms of boundary values of the solution itself are called the Poincaré-Steklov operators [32] or Dirichlet-to-Neumann maps [3, 4].

Having obtained the discrete Dirichlet-to-Neumann map $\mathbf{S}_\omega$ by solving (54), we are then able to calculate ξ_γ for any $\mathbf{u}_\nu$ provided from inside the computational domain D_{in}:

$$\xi_\gamma = \pi_{\gamma\omega}\left(\mathbf{R}_{\omega\nu}\mathbf{u}_\nu,\, \mathbf{S}_\omega\mathbf{R}_{\omega\nu}\mathbf{u}_\nu\right). \tag{55}$$

Finally, we use the function ξ_γ of (55) as the density of the generalized difference potential (50) and interpolate this potential from the grid $\mathcal{N}_{ex}$ to the nodes $\nu_1 \subset$, $_1$ using some local formulas of sufficiently high order [18]. In so doing, we obtain the desirable ABC's in the form

$$\mathbf{u}_{\nu_1} = \mathbf{R}_{\nu_1\mathcal{N}_{ex}}\mathbf{P}_{ex}\pi_{\gamma\omega}\left(\mathbf{R}_{\omega\nu}\mathbf{u}_\nu,\, \mathbf{S}_\omega\mathbf{R}_{\omega\nu}\mathbf{u}_\nu\right) \equiv \tag{56}$$

$$\mathbf{T}\,\mathbf{u}_\nu,$$

where $\mathbf{R}_{\nu_1\mathcal{N}_{ex}}$ is the aforementioned interpolation from $\mathcal{N}_{ex}$ to ν_1. Boundary conditions (56) obviously close the discrete system solved inside D_{in} because they provide for the missing relations between the values of the solution at the penultimate and outermost coordinate rows of the C-grid. Moreover, we are guaranteed that this closure is right from the standpoint of the far-field asymptotic behavior of the solution since the operator $\mathbf{T}$ of (56) is constructed using the resolved form $\mathbf{S}_\omega$ of the boundary projection and the potential (50). We should also note that since all the operators involved in (56) are linear we can actually calculate the matrix of the operator $\mathbf{T}$ in some appropriately chosen basis. This makes the practical implementation of boundary conditions (56) particularly easy even in spite of their nonlocal nature because this implementation is, in fact, reduced to a matrix-vector multiplication. Note, the simplicity of practical implementation of the DPM-based ABC's is not affected by the shape of artificial boundary , ; the matrix form (56) of the ABC's always remains the same although for the different shapes of , and , $_1$ the matrices $\mathbf{T}$ are also different. Moreover, as could be seen from our previous considerations these different matrices $\mathbf{T}$ are themselves calculated by means of one and the same (i.e., geometrically universal) numerical algorithm, which simply uses the shape of the artificial boundary (more precisely, the actual locations of nodes ν and ν_1) as the input data. This algorithm requires one solution of the difference AP per basis vector, as well as some special numerical procedure that includes matrix inversion and multiplication for obtaining $\mathbf{S}_\omega$. Note, the basis, in which we actually calculate the matrices of all operators, is actually chosen in the space of ξ_ω's, and the interpolation $\mathbf{R}_{\omega\nu}$ is applied afterwards. It is also important to emphasize that the RHS's for the difference AP are always concentrated near , (see (51)), and the solution of the AP also needs to be known only near , and , $_1$. Therefore, the solution of the difference AP (direct and inverse Fourier transforms and the solution of systems (46), (48) for all k) requires only $\mathcal{O}(M \cdot J)$ floating-point operations; a detailed justification of this estimate is contained in [18], see also [31].

Further delineation of the foregoing numerical algorithm for calculating the matrix $\mathbf{T}$ from (56) can be found in our work [18]. The results of implementation of boundary conditions (56) for flow computations are reported in [26, 27], some of these results are reproduced and discussed in Section 3 of this paper.

Another approach to setting the DPM-based ABC's is based on the direct implementation of boundary projections, this approach has recently been proposed in [28, 29], see also [17]. The main purpose of introducing the new approach was to reduce

the computational cost of the ABC's. In the new methodology, the difference boundary projection $\mathbf{P}_\gamma$ of (52) is constructed on the basis of the thin-layer system (41) using the operator $\mathbf{L}_h$ defined in (43); the operation $\pi_{\gamma\omega}$ that we use for the continuation of boundary data from , to γ is the same as described above (it is based on the Taylor expansion). However, unlike the previous case [18, 26, 27], in which the boundary conditions are driven only by $\mathbf{u}_\nu$ (see (56)), we now consider both $\mathbf{u}_\nu$ and $\left.\dfrac{\partial \mathbf{u}}{\partial \zeta}\right|_\nu$ as the input data for the ABC's. Assuming that these data are provided from inside the computational domain D_{in}, we then obtain the density of the generalized difference potential by first continuing the boundary data from , to γ and then applying the difference projection $\mathbf{P}_\gamma$,

$$\xi_\gamma = \mathbf{P}_\gamma \pi_{\gamma\omega} \mathbf{R}_{\omega\nu} \left(\mathbf{u}, \frac{\partial \mathbf{u}}{\partial \zeta} \right)\Bigg|_\nu . \tag{57}$$

In other words, we project the arbitrary boundary data provided from inside D_{in} onto the "right manifold" in the sense that ξ_γ of (57) already belongs to the image of the boundary projection $\mathbf{P}_\gamma$, $\xi_\gamma \in \mathrm{Im}\mathbf{P}_\gamma$, and can therefore be represented as a trace of some $\mathbf{u}_{\mathcal{N}_{ex}}$ that solves the equation $\mathbf{L}_h \mathbf{u}_{\mathcal{N}_{ex}} = \mathbf{0}_{\mathcal{M}_{ex}}$ and satisfies boundary conditions (45), (48).

To actually find the aforementioned complement $\mathbf{u}_{\mathcal{N}_{ex}}$ of ξ_γ on $\mathcal{N}_{ex}$, we need to compute the difference potential $\mathbf{P}_{ex}\xi_\gamma$ (see (50)) for the density ξ_γ of (57) (which requires the solution of the difference AP). Then, interpolating this potential from $\mathcal{N}_{ex}$ to $\nu_1 \subset$, $_1$, we obtain $\mathbf{u}_{\nu_1}$. Combining the foregoing steps, we can write the desirable ABC's in the form

$$\mathbf{u}_{\nu_1} = \mathbf{R}_{\nu_1 \mathcal{N}_{ex}} \mathbf{P}_{ex} \mathbf{P}_\gamma \pi_{\gamma\omega} \mathbf{R}_{\omega\nu} \left(\mathbf{u}, \frac{\partial \mathbf{u}}{\partial \zeta} \right)\Bigg|_\nu . \tag{58}$$

Let us now consider an arbitrary function ξ_γ. By definition (see formula (50)), the generalized potential $\mathbf{P}_{ex}\xi_\gamma$ with the density ξ_γ satisfies on $\mathcal{N}_{ex}$ the equation $\mathbf{L}_h \mathbf{P}_{ex}\xi_\gamma = \mathbf{0}_{\mathcal{M}_{ex}}$ and boundary conditions of the AP (45), (48). In turn, one can easily show (see [8]) that any function $\mathbf{u}_{\mathcal{N}_{ex}}$ that solves the equation $\mathbf{L}_h \mathbf{u}_{\mathcal{N}_{ex}} = \mathbf{0}_{\mathcal{M}_{ex}}$ with boundary conditions (45), (48) can be represented as $\mathbf{u}_{\mathcal{N}_{ex}} = \mathbf{P}_{ex}\eta_\gamma$, where $\eta_\gamma = \mathbf{Tr}_\gamma \mathbf{u}_{\mathcal{N}_{ex}}$. In our case $\eta_\gamma = \mathbf{Tr}_\gamma \mathbf{P}_{ex}\xi_\gamma = \mathbf{P}_\gamma \xi_\gamma$ and therefore $\mathbf{P}_{ex}\xi_\gamma = \mathbf{u}_{\mathcal{N}_{ex}} = \mathbf{P}_{ex}\eta_\gamma = \mathbf{P}_{ex}\mathbf{P}_\gamma\xi_\gamma$. In other words, for any ξ_γ the following relation

$$\mathbf{P}_{ex}\xi_\gamma = \mathbf{P}_{ex}\mathbf{P}_\gamma\xi_\gamma$$

is true. Consequently, instead of (58) we can write the ABC's as follows

$$\mathbf{u}_{\nu_1} = \mathbf{R}_{\nu_1 \mathcal{N}_{ex}} \mathbf{P}_{ex} \pi_{\gamma\omega} \mathbf{R}_{\omega\nu} \left(\mathbf{u}, \frac{\partial \mathbf{u}}{\partial \zeta} \right)\Bigg|_\nu \equiv \mathbf{T} \left(\mathbf{u}, \frac{\partial \mathbf{u}}{\partial \zeta} \right)\Bigg|_\nu \tag{59}$$

(the matrices $\mathbf{T}$ in equations (56) and (59) are obviously different). Clearly, boundary conditions (59) provide for the missing relations between the values of the solution on the penultimate and outermost coordinate rows of the C-grid and therefore close the finite-difference system that we solve inside D_{in}. Moreover, this closure is consistent with the desirable far-field behavior of the solution because of the projection $\mathbf{P}_\gamma$ incorporated in (58).

Comparing boundary conditions (56) and (59) we see that they are essentially different. Direct usage of the boundary projection $\mathbf{P}_\gamma$ in (58), (59) allows us to completely avoid the entire resolving stage of the algorithm that is inherent for the approach of the first type summarized in formula (56). Recall, the resolving stage in (56) is associated with the computation of $\mathbf{S}_\omega$ (resolved form of the boundary projection) on the basis of equation (54). Elimination of this stage implies an essential simplification of the algorithm, as well as noticeable economy of computer resources, i.e., the reduction of the computational cost of boundary conditions (59) in comparison with boundary conditions (56).

Of course, another part of this cost reduction, which is even more essential, is accounted for by the reduction of order n of one-dimensional finite-difference system (46) from the eighth to the fourth. Indeed, we recall that boundary conditions (56) were obtained on the basis of the second-order central-difference discretization of system (40) and boundary conditions (59) were obtained on the basis of the discretization (43) of system (41). In so doing, the matrices $\mathbf{A}_k$ and $\mathbf{B}_k$ in system (46) have order $n = 8$ for boundary conditions (56) (see [18]) and order $n = 4$ for boundary conditions (59) (see (44), (47)). According to [31], the solution of one-dimensional problem (46), (48) for each k costs $\mathcal{O}(M \cdot n^2)$ floating-point operations. Therefore, the solution of the entire AP costs $\mathcal{O}(M \cdot J \cdot n^2)$ operations. For both boundary conditions (56) and (59) we have to solve the AP repeatedly, one time per basis vector. Consequently, one can expect that for the same geometry of the discrete sets ν, ν_1, and ω, for the same basis in the space of ξ_ω's, and for the same grid $\mathcal{N}^0$, the computational cost of the matrix $\mathbf{T}$ from (59) will be at least four times less that

the cost of matrix $\mathbf{T}$ from (56). Taking into account the foregoing elimination of $\mathbf{S}_\omega$ from the structure of ABC's, we conclude that the overall improvement of the computational efficacy when going from (56) to (59) will be even more drastic. Our computational experiments do corroborate this theoretical expectations. In fact, we could gain up to a factor of five in the reduction of cost of the ABC's (56) in comparison with ABC's (59). Moreover, our preliminary estimates show that in the case of three space dimensions this gain may increase.

As mentioned above, along with being computationally cheaper than the original technique (56) the new methodology (59) also appears much simpler from the algorithmic standpoint. At the same time, boundary conditions (59) do posses all the aforementioned favorable properties that are relevant to boundary conditions (56). Namely, they are geometrically universal, easy to implement in practice, and of course, they perform as good as (or even better than) (56) from the standpoints of accuracy, overall efficacy, and robustness (see Section 3).

2.5 Implementation of DPM-based ABC's

The ABC's of both types (56) and (59) are designed to close the finite-difference system solved inside D_{in}, i.e., to make sure that the number of equations solved inside D_{in} is equal to the number of unknowns. Both relations (56) and (59) are spatially nonlocal, which in practical terms means that the matrices $\mathbf{T}$ are dense. Although these matrices are, in fact, structural, we have not used this property for practical computations yet. (Qualitative structure of the matrices $\mathbf{T}$ can be understood from physical considerations. If the vectors $\mathbf{u}_\nu$ and $\mathbf{u}_{\nu_1}$ are arranged properly, then we can consider $\mathbf{T}$ as being composed of several blocks. Each block of $\mathbf{T}$ would correspond to one physical variable (u, v, p, or ρ) row-wise, i.e., for all nodes ν, and one physical variable column-wise, i.e., for all nodes ν_1, and would have a kind of "diagonal dominance" in the sense that the entries located near the main diagonal will be greater that those located far away from this diagonal. In physical terms, it merely means that each specific node influences its close neighbors stronger than it influences the nodes located on the other side of the computational domain.)

So far, we have been discussing the ABC's only from the viewpoint of closing the system solved inside D_{in} so that the closure is consistent with the desirable far-field behavior of the solution. In practice, however, the construction of a formal closure of the finite-difference system solved inside D_{in} is not sufficient, we also have to combine this closing procedure with the specific solver. The majority of solvers currently used in CFD for calculating the steady-state viscous flows on the basis of finite-difference discretizations employ various types of pseudo-time iterations. In most cases, the iterations are explicit in time and may be enhanced by different techniques for the purpose of accelerating the convergence, e.g., by multigrid.

We have to emphasize that both boundary conditions (56) and (59) are particularly well fitted for the combined usage with explicit iterative solvers. Indeed, let us assume that on some time level (which, in particular, may be zero) the solution is already known on the entire C-grid. Then, advancing one time step by means of some explicit technique we obviously cannot obtain the next-level solution also on the entire C-grid. The reason for that is exactly the same as why the original steady-state finite-difference system inside D_{in} would be subdefinite without the ABC's — the stencil applied to some external nodes of the C-grid may partially "fall out" of the domain. In other words, when using solely the procedure employed inside D_{in}, we can obtain the solution on the upper time level only at the "internal" nodes of the C-grid. For the case of the stencil 3×3, this "internal" set includes all nodes of the C-grid except for the outermost coordinate row $\nu_1 \subset , _1$. The values of the solution on this outermost coordinate row should therefore be provided by the ABC's so that the solution on the upper time level becomes available everywhere, which makes the next iteration feasible. Looking at boundary conditions (56) and (59) one can see that the desirable complement of the solution on the upper time level can easily be obtained by means of either one of these techniques. When using boundary conditions (56), we simply take $\mathbf{u}_\nu$ that is already computed on the upper time level and, applying the operator $\mathbf{T}$ (matrix-vector multiplication), obtain $\mathbf{u}_{\nu_1}$. This operation is repeated on every iteration; if the relaxation procedure requires evaluation of the residuals more than once per iteration (e.g., multi-stage Runge-Kutta), then boundary conditions (56) are used as many times per iteration as the residuals need to be evaluated.

The implementation of boundary conditions (59) is analogous. After making one iteration of the Navier-Stokes solver inside D_{in} we know the solution on the upper time level everywhere in the interior of the curve , . Therefore, we can take $\mathbf{u}_\nu$ on the upper time level as done above and also can easily calculate $\left.\dfrac{\partial \mathbf{u}}{\partial \zeta}\right|_\nu$ using the available data. Then, applying the matrix $\mathbf{T}$ from (59), we obtain $\mathbf{u}_{\nu_1}$ on

the upper time level and therefore make it possible to advance another time step. One can see that in both cases (56) and (59) practical implementation of the DPM-based ABC's is very easy since it is reduced to a matrix-vector multiplication on each iteration. Moreover, the implementation is in no way affected by the shape of artificial boundary, although the operators $\mathbf{T}$ themselves would, of course, depend on the geometry.

The implementation of the DPM-based boundary conditions would obviously require some changes if the solver employed inside D_{in} uses multigrid for the acceleration of convergence. Namely, for the multigrid iterative solver the values of the solution on the outermost coordinate row of the grid should be provided every time the residuals need to be evaluated on every level of multigrid. Since the operators $\mathbf{T}$ do depend on the geometry, we may formally need to calculate a separate operator for each subsequent grid. It, however, turns out that at least for some multigrid strategies the numerical process appears to be sensitive only to the ABC's specified at the finest level of multigrid, whereas the sensitivity of the numerical process to the boundary conditions on all the coarser levels is negligible.

In all the computations reported in Section 3, we have used the algorithm [33, 34, 35] by Swanson and Turkel for obtaining the steady-state solutions of the Navier-Stokes equations inside D_{in}. This algorithm is based on the second-order central-difference approximation in space and Runge-Kutta relaxation in time. In practice, we have always used five-stage Runge-Kutta time stepping. The code, in which the algorithm of [33, 34, 35] is implemented, allows one to use different multigrid strategies for the acceleration of convergence. Depending on the specific computational variant, we used from three to five levels of multigrid with W-cycles, when one iteration is done on the finest level and two iterations are done on each of the coarser levels of multigrid. As the additional means of convergence acceleration, the algorithm of [33, 34, 35] includes local time stepping and residual smoothing.

The standard treatment of the external boundary in the code [33, 34, 35] is based on the locally-one-dimensional analysis of characteristics for the inflow part of the boundary and the boundary conditions of extrapolation for the outflow part of the boundary. This approach is actually one of the most well-known and widely used in CFD, its different versions have many times been described in the literature, see, e.g., the reviews [1, 2, 3, 4]. The purely local characteristics/extrapolation treatment may or may not be enhanced in the code [33, 34, 35] by the point-vortex correction. This lift-based correction [36] usually improves the results provided by the original local ABC's.

We have experimentally made certain that for the different flow regimes (see Section 3, as well as [26, 27]) computed on the basis of the foregoing multigrid strategy, neither the convergence history nor the calculated solution depend on whether we specify the standard local ABC's (see above) on all levels of multigrid or we specify these boundary conditions on the finest level only and for all the coarser levels simply retain the boundary values provided from the finest level. This gave us reasons to expect that the nonlocal DPM-based ABC's (56) or (59) can also be set on the finest level of multigrid only, whereas the boundary values for all the coarser levels will be provided from the finest one. In all the computations reported below, we used exactly this scheme of implementation of the nonlocal DPM-based ABC's. Numerical results (see Section 3) corroborate the possibility of doing so, at least for those multigrid strategies that we used for our computations.

Finally, we should note that the implementation of the DPM-based ABC's with implicit iterative solvers also seems feasible, at least from the formal standpoint. Although we have never tried to run any numerical experiments, theoretically this implementation would simply mean that the nonlocal relations of type (56) or (59) are incorporated in the system that is solved on the upper time level on each step of the implicit iteration process.

2.6 Miscellaneous Issues Important for Calculation of DPM-based ABC's

First, we comment here on the choice of the discretization parameters for the difference AP. The unknowns for this problem are specified on the Cartesian grid $\mathcal{N}^0$. The cell size of this grid should be chosen so that the distance between the curves , and , $_1$ (see Figure 1) is resolved. In practical computations we usually take the cell size of the grid $\mathcal{N}^0$ to be 3–5 times smaller than the minimal distance between , and , $_1$.

As concerns another important parameter of the difference AP, the period Y (see Figure 1), it should be chosen so that to ensure the sufficient accuracy of computations. This choice, of course, cannot be done without taking into account the previous computational experience. However, we first have to choose the unit for measuring Y. It is reasonable to expect that the average diameter of the computational domain D_{in} would make a better unit for measuring the period Y than the characteristic size

of the immersed body(ies) (e.g., airfoil chord, see Figure 1). Indeed, the final accuracy will depend on the "extent of convergence", i.e., on how close the solution of the periodic AP has approached the solution of the original nonperiodic AP. Since the size of the computational domain D_{in} actually determines the size of that subdomain in D_Y^0, on which we consider the convergence (e.g., the rectangle $|y| < a$, $|x - a| < a$), then exactly the size of D_{in} becomes a natural unit for measuring the period Y as the variable that controls the convergence.

As mentioned above, the value of the period Y is one of the factors that determine the accuracy of computations. The accuracy, of course, also depends on the actual size of D_{in}. Generally, the DPM-based ABC's allow one to use much smaller computational domains than other available methods do. The specific numerical results will be reported in Section 3; here, we provide only for a qualitative picture. In [26, 27], we have conducted a series of computations for the relatively small values of the period Y, about 2–4 diameters of D_{in}. It turns out that for a small computational domain (2–3 chords of the airfoil for transonic flow regions), the solution obtained on the basis of the DPM-based ABC's differs within 2–4% from the asymptotic solution obtained in a very large computational domain (about 30–50 chords depending on the specific variant). (Note, when comparing the two solutions, we actually compare the corresponding force coefficients: lift, drag, skin friction). These results (see [26, 27]) can be satisfactory for some cases; however, when the accuracy needs to be improved one has to choose larger periods Y. For example, to keep the force coefficients within 0.01%–0.1% of the corresponding asymptotic values, one may need to choose the period Y of about 50 diameters of the computational domain or more.

That big periods Y, along with the small cell size of the grid $\mathcal{N}^0$ prescribed by the distance between , and , $_1$ may, generally speaking, result in an expensive numerical procedure for computation of the operators $\mathbf{T}$. To reduce this cost, we introduce nonuniform grids with respect to y. For example, the cell size of such a grid may remain constant (the same as it would be for the uniform grid) in the neighborhood of D_{in} and then enlarge as $|y|$ increases. Clearly, the discrete Fourier transform in the y direction that we have formerly used for the separation of variables does not apply to the stretched grids. Consequently, we will need some other procedure to separate the variables and to therefore reduce the difference AP to a family of one-dimensional systems (46) with boundary conditions (48). The separation of variables for a finite-difference counterpart of system (41) means that the discrete transform should simultaneously diagonalize the difference approximations to both the first and the second derivatives with respect to y. We have not studied thoroughly the question of whether or not one can find a special distribution of nodes in the y direction and some consistent with this distribution discretization so that the diagonalizing transform appears orthogonal. In practice, it turns out that usage of the skew bases solves the problem of simultaneous diagonalization of the first and the second derivatives and at the same time allows one to still maintain high accuracy of the final results.

Following this idea we first introduce some second order discretization of the first derivative with respect to y on the stretched grid, we also take into account the periodicity conditions (45). In the matrix form, this operation can be represented as a certain $(2J + 1) \times (2J + 1)$ matrix $\mathcal{D}$, which has three non-zero diagonals, $j - 1 \le i \le j + 1$, and also nonzero off-diagonal corner entries. Then, the matrix $\mathcal{D}^2$ represents the second difference derivative with respect to y. Both these matrices can obviously be diagonalized by the same transform, which is computed in practice with the help of the standard library eigenvalues/eigenvectors subroutines (IMSL). The rest of the algorithm remains the same as described above. The only difference is that instead of r_k and t_k in (47) one should plug in the eigenvalues of $\mathcal{D}$ and their squares, respectively. Our experiments show that usage of the stretched grids can drastically reduce the computer effort required for calculating the nonlocal DPM-based ABC's. An essential part of the results reported in Section 3 is obtained on the basis of the stretched-grid algorithm.

Another means for reducing the computational cost of the DPM-based ABC's is implementation of the algorithm on parallel platforms. There are, generally speaking, a few ways to parallelize the computation of operators $\mathbf{T}$. Since we anyway calculate the matrix of the linear operator in a certain basis, it is possible to calculate independently and, therefore, simultaneously, the columns of this matrix, where each column corresponds to its own basis vector. We, however, have chosen another way of parallelization, which, in our opinion, requires minimal modifications of the already developed sequential code. Namely, one-dimensional systems (46) with boundary conditions (48) are obviously independent because they are obtained by the separation of variables (k is a parameter). Therefore, these systems can be solved in parallel on the different processors of a multi-processor computer. This approach has been implemented on an eight-processor CRAY Y-

MP. Numerical experiments show the reduction of the "wallclock" time required for calculating the operator $\mathbf{T}$ of (59) by up to a factor of five in comparison with the standard single-processor implementation.

Finally, we should note that the treatment of turbulence in the far field requires special attention. Indeed, the parameter Re in systems (40) and (41) represents the true molecular Reynolds number, therefore these forms of the governing equations apply directly only to the laminar flows. The approximate treatment of turbulence in the far field for the purpose of constructing the ABC's was proposed in the work [27]. Below, we reproduce some results of this work.

To numerically simulate turbulent flows, we use an algebraic turbulence model (Baldwin–Lomax) incorporated in the code [33, 34, 35]. This model is relevant to describing the flow in the vicinity of the immersed body(ies). In the far field, we use simpler approach based on the concept of effective turbulent viscosity [37]. 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 the 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) [37]:

$$u = \frac{W}{2\sqrt{\pi\rho^2\nu u_0 x}}\, exp\left(-\frac{u_0 y^2}{4\nu x}\right). \tag{60}$$

We recall [37, 38] that formula (60) 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 in (60) is that of force per unit length.)

For the turbulent case we assume [37, 38] that the mixing length l for the wake flow is proportional to the local width b of the wake, $l/b = const$. Then, approximately replacing the value of the derivative in the expression for turbulent viscosity, $\nu_t = l^2 \left|\frac{d\,u}{d\,y}\right|$, by the ratio u_{max}/b, where u_{max} is the maximal deviation of the actual velocity from u_0 (which corresponds to the middle of the wake), we easily obtain $\nu_t = const \cdot b \cdot u_{max}$, which, in particular, implies that ν_t does not depend on y. We may also assume [37] 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,

$$\nu_t = \frac{kW}{\rho u_0}, \tag{61}$$

where $k = const$. Once can easily see from (61) that $\nu_t = const$ throughout the whole wake region and therefore, the structure of turbulent wake behind the body (i.e., the profile of mean velocity) appears to be the same as the structure of laminar wake [37, 38] since we actually have the same solution as (60),

$$\begin{aligned} u &= \frac{W}{2\sqrt{\pi\rho^2\nu_t u_0 x}}\, exp\left(-\frac{u_0 y^2}{4\nu_t x}\right) = \\ &\frac{1}{2\sqrt{\pi}}\sqrt{\frac{W}{k\rho x}}\, exp\left(-\frac{\rho {u_0}^2 y^2}{4kW x}\right), \end{aligned} \tag{62}$$

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

Recall, our purpose is to find such ν_t that, being substituted into (60), will provide the same wake solution as we would have for the turbulent case. Clearly, ν_t from (61) satisfies this requirement (see (62)), 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 [37] to calculate the latter quantity, and restricting ourselves (for qualitative consideration) by the case of not high longitudinal pressure gradients, we can obtain,

$$\nu_t = \frac{k_C W}{\rho u_0},$$

which means that the value of the unknown constant k in (61) 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 [37]

$$W = 0.0307 Re^{-1/7} \rho u_0^2 L, \tag{63}$$

where Re is the actual Reynolds number based on the molecular viscosity and L is the characteristic

length. To take into account the compressibility, we multiply equation (63) by $\left(\frac{2}{2+0.5(\gamma-1)M_0^2}\right)^{5/7}$ in accordance with the Tucker conjecture [37]. Then, we introduce an empirical constant Θ, which is the ratio of the pressure and viscous contributions to the total drag (obviously, $\Theta = 0$ for the flat plate); for the transonic flows computed below, $\Theta \approx 5/2$. Finally, we obtain the following relation,

$$Re_t = \frac{u_0 L}{\nu_t} = \frac{Re_0^{1/7}}{0.0307 k_C (\Theta+1)} \left(\frac{2}{2+0.5(\gamma-1)M_0^2}\right)^{-5/7},$$

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

3 Numerical Results

3.1 Acceleration of Convergence

One of the most important aspects of implementation of any ABC's is the influence that the boundary conditions exert on the convergence to steady state. Our numerical experiments for both two [26, 27] and three [19, 20] space dimensions show that the nonlocal DPM-based ABC's can essentially speed up the convergence of the multigrid iterations compared to the standard characteristics-based boundary conditions.

This positive influence on the convergence rate is, however, not a general situation. It appears that the acceleration of convergence typically occurs only when the interior iterative solver involves multigrid. Otherwise, the nonlocal highly accurate ABC's either do not influence the convergence at all or may even slow it down. For example, the observation of the latter kind was done by Ferm in [39] for the nonlocal ABC's [39, 40] (by Gustafsson and Ferm) that are constructed for the Euler flows in ducts using Fourier transform in the cross-stream direction. The same phenomenon also occurs for the external inviscid flows as shown by Ferm in the work [41], in which he studies the convergence of pseudo-time iterations for the Euler equations supplemented by the nonlocal ABC's [42] (boundary conditions [42] are constructed analogously to [39, 40] for elliptic artificial boundaries). To accelerate the convergence of pseudo-time iterations with nonlocal boundary conditions, Ferm in [39, 41] employs the technique of [43] by Engquist and Halpern (or its modification), which allows him to make the convergence at least as fast as it is for the simplest locally-one-dimensional non-reflecting boundary conditions that are based on the analysis of characteristics. On the other hand, when boundary conditions [42] are implemented along with some multigrid Euler solver inside the computational domain they no longer slow down the convergence and, therefore, no longer require special acceleration procedures (like the one from [43]). This has been demonstrated by Ferm in the work [44], in which he shows that in order to reduce the initial error by a prescribed factor one needs roughly the same number of multigrid cycles for both nonlocal ABC's [42] and the characteristic non-reflecting boundary conditions.

As mentioned above, when implemented along with the multigrid algorithm [33, 34, 35] (see Section 2), the DPM-based ABC's are capable of even speeding up the convergence to steady state compared to the standard boundary conditions. Let us reproduce here several graphs from [26] that represent the convergence history for different subsonic and transonic laminar flows around the airfoil NACA0012. In the captions to the figures below, α denotes the angle of attack.

From Figures 2, 3, 4, and 5, one can easily see that usage of the DPM-based ABC's can increase the convergence rate of the multigrid iterations by up to a factor of three depending on the specific variant of computations. Note, the subcritical (i.e., fully subsonic) laminar cases that correspond to Figures 2, 3, 4, and 5 have been computed on the grids with small stretching ratios because near the airfoil surface those grids could be chosen relatively coarse. As a result, we have used global rather than local Courant step for iterations in time. In this respect, one can say that Figures 2, 3, 4, and 5 demonstrate the influence exerted by the DPM-based ABC's on a "pure" multigrid (augmented only by residual smoothing).

For the case of two-dimensional turbulent flows that are computed on the grids with much higher stretching ratio and with the local (i.e., chosen cell by cell) Courant step in time, we have not been able to obtain as drastic convergence speedup as for the foregoing laminar cases. The history of convergence for two different two-dimensional transonic turbulent cases is presented in Figures 6 and 7. We however, mention, that for many three-dimensional transonic turbulent cases, the DPM-based ABC's have been able to produce the increase of the convergence rate about as big as shown above for the two-dimensional laminar flows. The corresponding results are reported in our work [19, 20] and will also

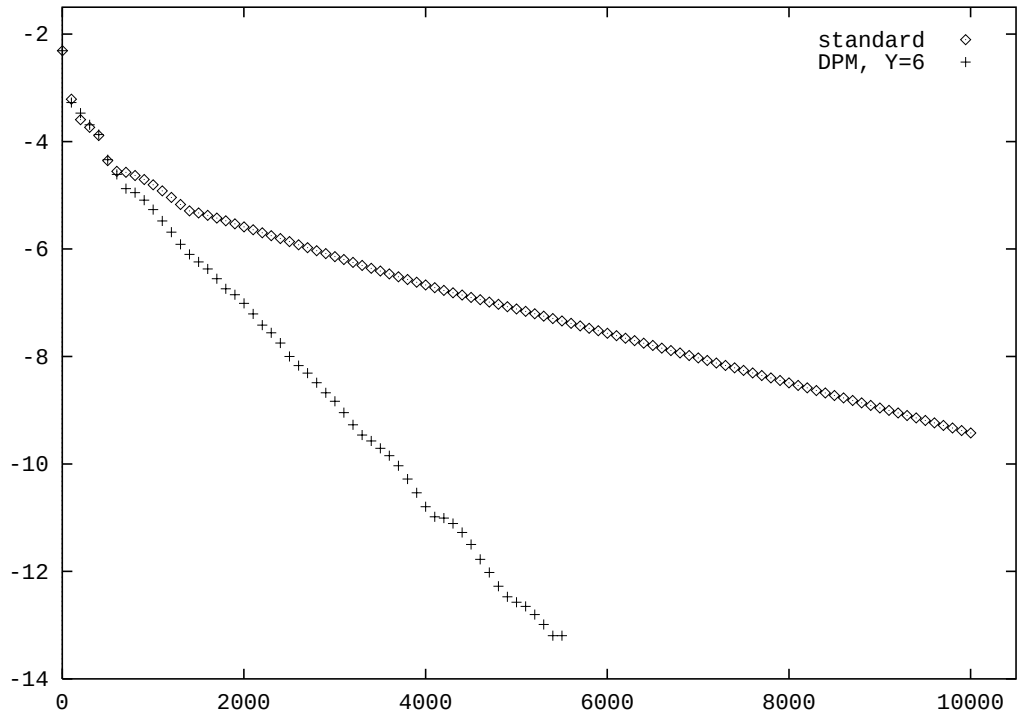

Figure 2. Convergence history: $\log\|\rho_{residual}\|_\infty$ versus number of cycles; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, $Re = 400$. Grid 256×64, average radius of D_{in} about 15 chords.

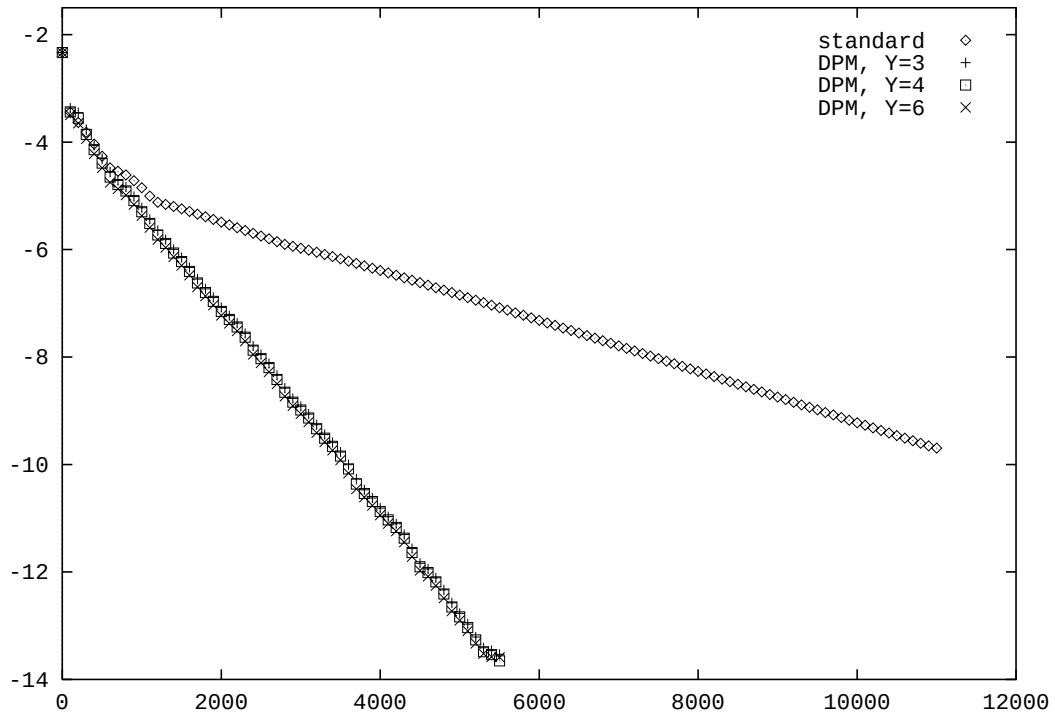

Figure 3. Convergence history: $\log\|\rho_{residual}\|_\infty$ versus number of cycles; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, $Re = 400$. Grid 256×64, average radius of D_{in} about 5.5 chords.

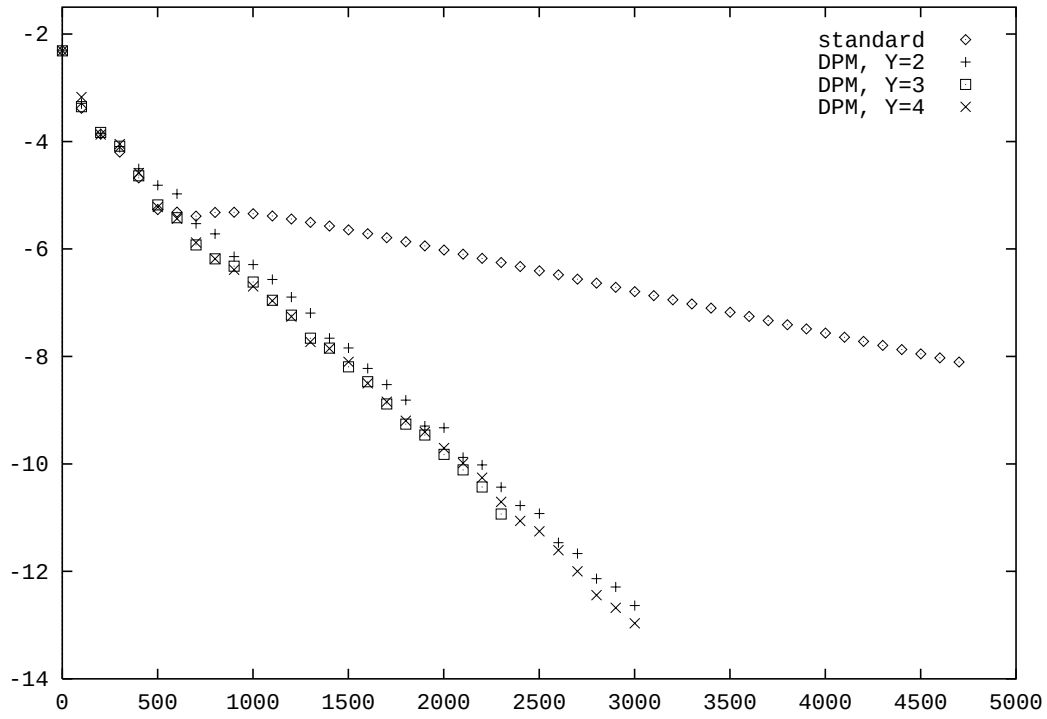

Figure 4. Convergence history: $\log\|\rho_{residual}\|_\infty$ versus number of cycles; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, $Re = 4000$. Grid 256×64, average radius of D_{in} about 5.5 chords.

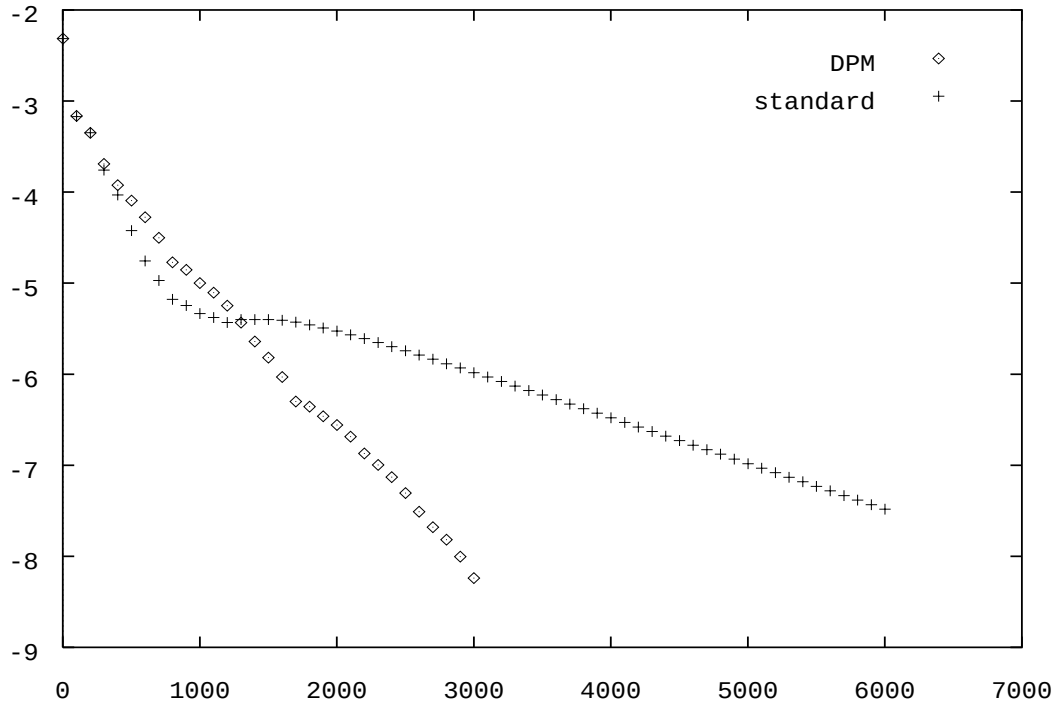

Figure 5. Convergence history: $\log\|\rho_{residual}\|_\infty$ versus number of cycles; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, $Re = 5000$. Grid 256×64, average radius of D_{in} about 10 chords.

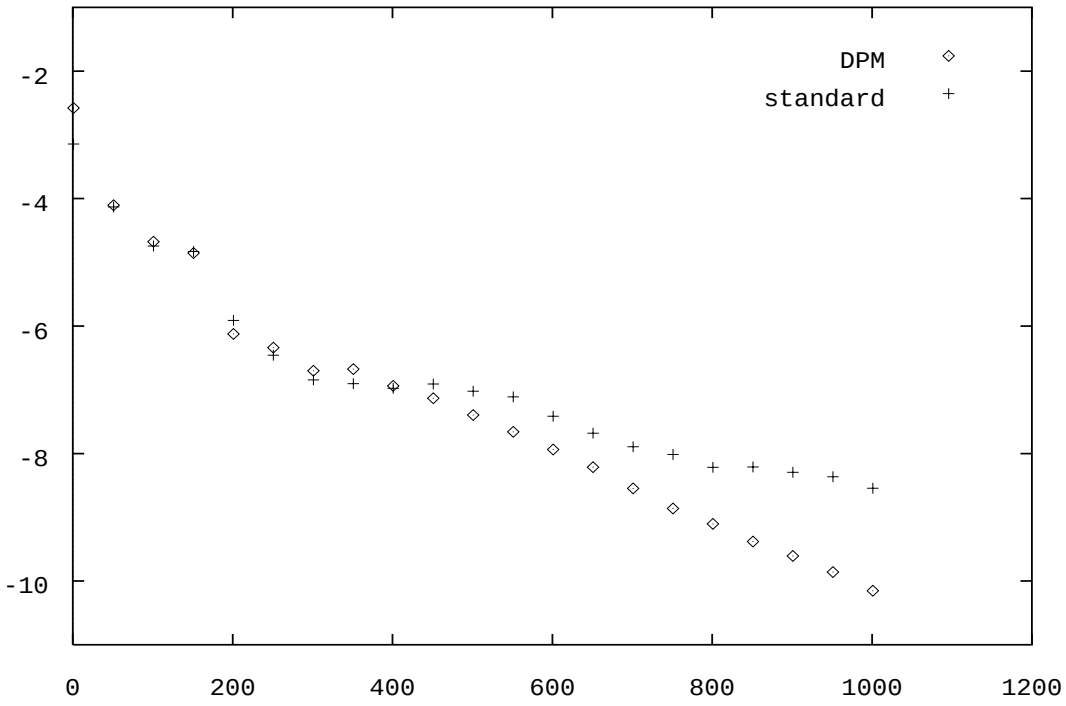

Figure 6. Convergence history: $\log\|\rho_{residual}\|_\infty$ versus number of cycles; RAE2822, $M_0 = 0.73$, $\alpha = 2.79°$, $Re = 6.5 \cdot 10^6$. Average radius of D_{in} about 11 chords, normal spacing near the airfoil 10^{-4}.

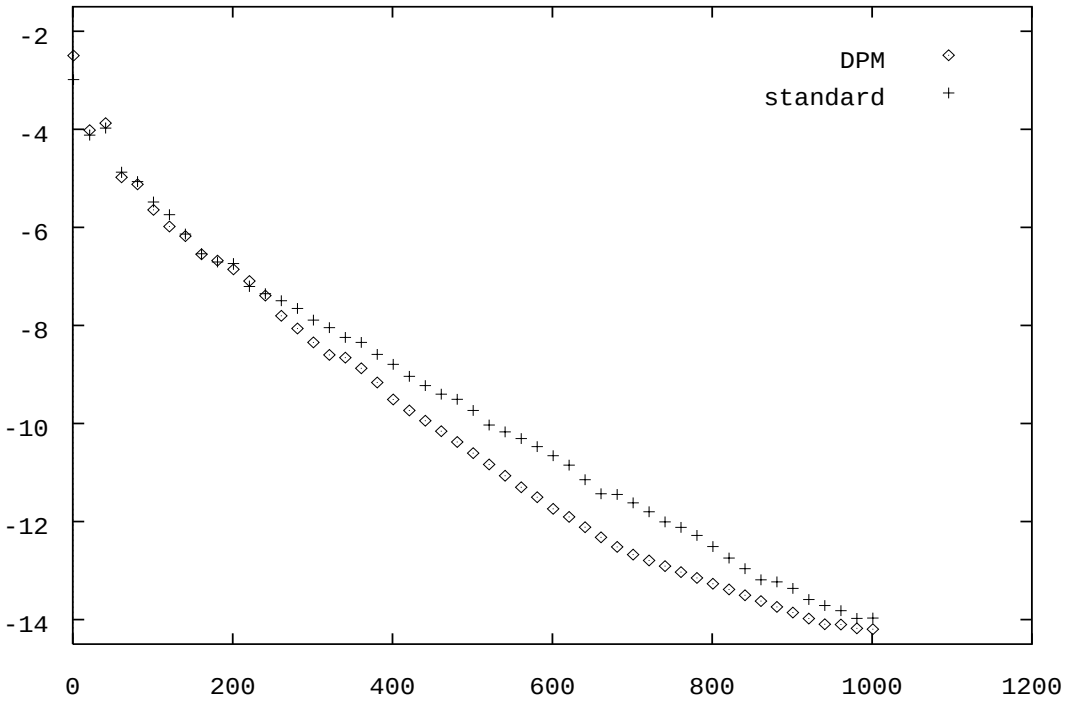

Figure 7. Convergence history: $\log\|\rho_{residual}\|_\infty$ versus number of cycles; RAE2822, $M_0 = 0.73$, $\alpha = 2.79°$, $Re = 6.5 \cdot 10^6$. Average radius of D_{in} about 11 chords, normal spacing near the airfoil 10^{-5}.

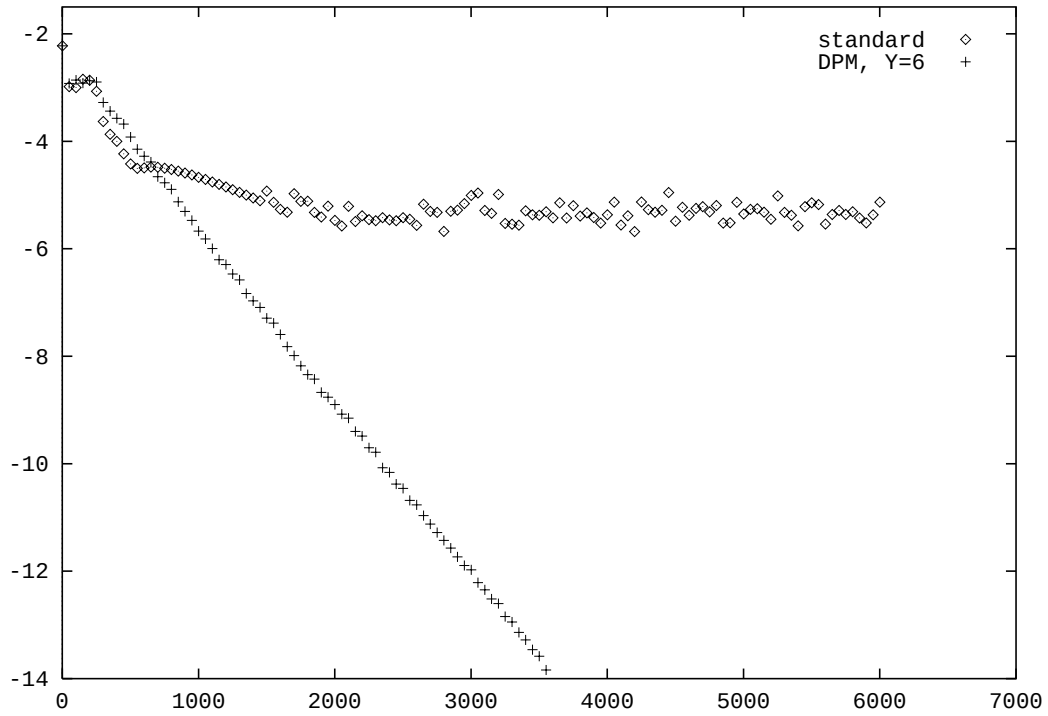

Figure 8. Convergence history: $\log\|\rho_{residual}\|_\infty$ versus number of cycles; NACA0012, $M_0 = 0.85$, $\alpha = 1°$, $Re = 4000$. Grid 256×64, average radius of D_{in} about 5.5 chords.

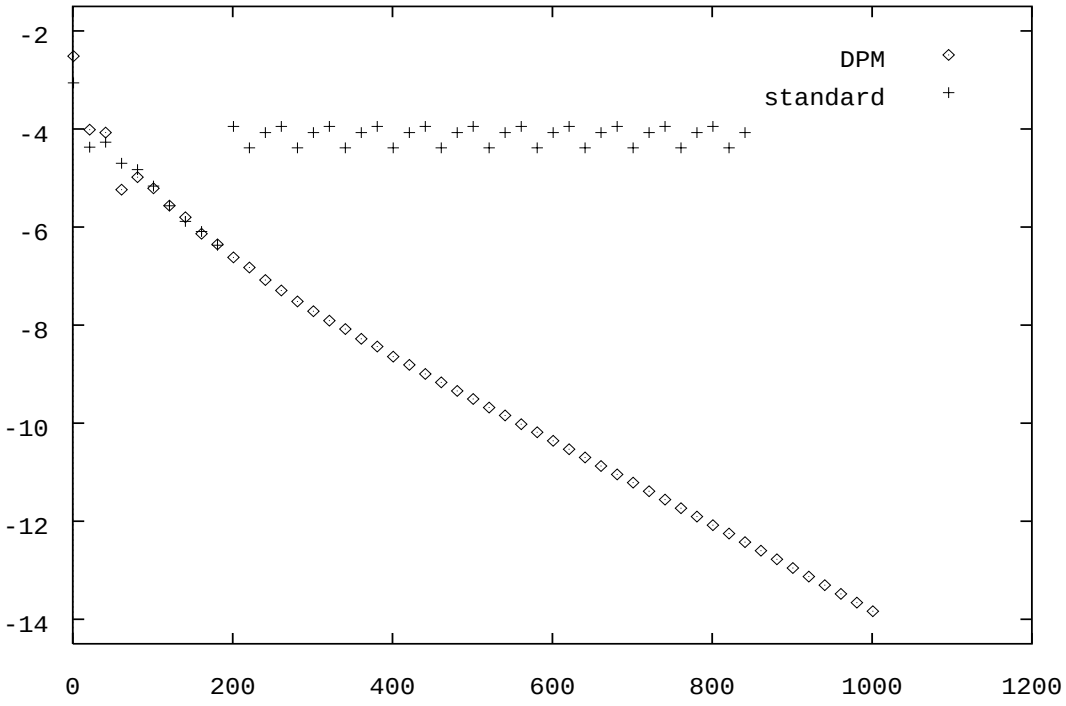

Figure 9. Convergence history: $\log\|\rho_{residual}\|_\infty$ versus number of cycles; RAE2822, $M_0 = 0.73$, $\alpha = 2.79°$, $Re = 6.5 \cdot 10^6$. Average radius of D_{in} about 6 chords, normal spacing near the airfoil $.33 \cdot 10^{-4}$.

be briefly discussed later in this paper.

Returning to Figures 2, 3, 4, and 5, we see that the convergence rates for two different types of ABC's are the same on the initial stage of the iteration process; then, for the DPM-based ABC's the convergence rate remains the same all the time and for the standard boundary conditions it drastically decreases. Therefore, it would be reasonable to assume that the ABC's start to actually influence the convergence only after the numerical perturbations caused by the immersed body reach the external boundary. In other words, the DPM-based ABC's become most effective from the standpoint of convergence acceleration on the so-called asymptotic stage of the multigrid. The similar type of behavior can be observed for the three-dimensional computations as well (see below).

We, however, have to say that although the acceleration of multigrid convergence provided by the DPM-based ABC's is extremely important for applications, the mechanism of interaction of the nonlocal DPM-based ABC's with multigrid may require an additional study. In fact, neither rigorous mathematical explanation of the convergence speedup nor a definite experimental conclusion of why and when it happens is available as of yet. For our two- and three-dimensional computations, we have used different multigrid strategies (W and V cycles, respectively), also all the three-dimensional cases have been computed with the local time step, and the results are also different. Whereas for the two-dimensional transonic turbulent flows we did not see much of an increase in the convergence rate, in three dimensions the strongest speedup occurs right for the transonic turbulent cases (see the work [19, 20] and also below). At the same time, the convergence rates for the subsonic turbulent flows in three space dimensions are the same for the ABC's of different types [19, 20]. As for the laminar flows, the experiments have been conducted in two space dimensions only.

Analyzing the influence that nonlocal boundary conditions may exert on the convergence of multigrid iterations, we should also note that many modern multigrid solvers are not optimal themselves. A massive effort is currently underway towards constructing the new finite-difference schemes, for which the convergence characteristics of multigrid methods would essentially improve. For example, the work in this direction has been done by Ta'asan [45] and Sidilkover [46]. In [45], Ta'asan devised an essentially optimal multigrid solver for the Euler equations in subsonic regime. Due to the separate treatment of the elliptic and advection parts of the system, this approach allows one to achieve in subsonic regime the convergence rates similar to those that can be obtained when solving the full potential equation. Sidilkover in [46] proposed the so-called genuinely multidimensional high-resolution scheme. This scheme has stability properties much superior to those that are relevant to the standard methods and, therefore, facilitates the construction of a very simple and efficient multigrid algorithm (using Gauss-Seidel relaxation as a smoother) that would apply to the entire range of Mach number. Particular efficacy demonstrated by the DPM-based ABC's when implemented in combination with multigrid gives us reasons to hope that these boundary conditions may essentially contribute in developing the new generation of effective multigrid-based algorithms.

Finally, we should mention that the DPM-based ABC's generally improve the robustness of the entire numerical procedure. In conducting our computational experiments [26, 27], we have noticed that sometimes the multigrid iterations [33, 34, 35] supplemented by the standard characteristic/extrapolation boundary conditions simply fail to converge, which never happens if these standard ABC's are replaced by the nonlocal DPM-based boundary conditions. In Figures 8 and 9, we show the history of convergence for the corresponding computations. The similar phenomenon has been observed in three dimensions as well. Namely, for a transonic turbulent flow with separation (see [20]), the multigrid iteration procedure with standard boundary conditions failed to converge, whereas the DPM-based ABC's have still been able to ensure a fast convergence to steady state.

3.2 Accuracy

Another most important outcome of usage of the DPM-based ABC's is the essential increase of accuracy that these boundary conditions provide for in computing the external viscous flows. Below, we compare some numerical results obtained on the basis of boundary conditions (59) for a certain transonic turbulent flow around the airfoil RAE2822 with the results obtained for the same flow regime on the basis of the standard local ABC's (characteristics/extrapolation) enhanced by the point-vortex correction [36]. In Table 1, we present the results for three different grids, 640×128, 608×112, and 600×104 nodes, that correspond to the computational domains of the average radii of 50, 8, and 2.5 chords of the airfoil, respectively. It is important that each subsequent (smaller) grid is obtained here by cutting off several external coordinate lines

Table 1. Comparison with the point-vortex (p.-v.) model for RAE2822 airfoil; $M_0 = 0.73$; $Re_0 = 6.5 \cdot 10^6$; $\alpha = 2.79°$; normal grid spacing near the airfoil $0.5 \cdot 10^{-5}$.

Domain "radius"	3 chords		8 chords		50 chords	
Grid	600×104		608×112		640×128	
Type of ABC's	p.-v.	(59)	p.- v.	(59)	p.-v.	(59)
C_l	0.8653	0.8591	0.8624	0.8589	0.8603	0.8593
relative error	0.58%	0.02%	0.24%	0.04%	0%	0%
$C_d \times 10$	0.1203	0.1263	0.1209	0.1261	0.1255	0.1260
relative error	4.14%	0.24%	3.67%	0.08%	0%	0%
$C_D \times 10$	0.1755	0.1816	0.1762	0.1815	0.1810	0.1815
relative error	3.04%	0.05%	2.65%	0%	0%	0%

Table 2. Comparison with the point-vortex (p.-v.) model for RAE2822 airfoil; $M_0 = 0.73$; $Re_0 = 6.5 \cdot 10^6$; $\alpha = 2.79°$; normal grid spacing near the airfoil $0.5 \cdot 10^{-5}$.

Domain "radius"	2.5 chords		50 chords			
Grid	320×64		320×64		640×128	
Type of ABC's	p.-v.	(59)	p.-v.	(59)	p.-v.	(59)
C_l	0.8688	0.8560	0.8504	0.8492	0.8603	0.8593
relative error	2.15%	0.38%	1.15%	1.17%	0%	0%
$C_d \times 10$	0.1123	0.1259	0.1260	0.1265	0.1255	0.1260
relative error	10.5%	0.07%	0.40%	0.39%	0%	0%
$C_f \times 100$	0.5469	0.5492	0.5478	0.5480	0.5543	0.5544
relative error	1.34%	0.94%	1.17%	1.15%	0%	0%
$C_D \times 10$	0.1670	0.1808	0.1808	0.1814	0.1810	0.1815
relative error	7.73%	0.39%	0.11%	0.05%	0%	0%

of the preceding (bigger) grid. This is done in order to completely avoid any possible influence that the change of the grid near the airfoil surface may exert on the solution.

From Table 1 one can see that the corresponding asymptotic values of the force coefficients (lift C_l, wave drag C_d, total drag C_D), i.e., the values obtained for the large (50 chords) computational domain, are very close to one another for the different types of ABC's. However, as the artificial boundary approaches the airfoil the discrepancy between the corresponding values increases, and the force coefficients obtained on the basis of boundary conditions (59) deviate from their asymptotic values much less than the coefficients obtained using local ABC's. In other words, the nonlocal DPM-based ABC's allow one to use much smaller computational domains than the standard boundary conditions do and to still maintain high accuracy of computations. Moreover, from Table 1 one can see that unlike the ABC's (59), which perform well for all coefficients, the point-vortex boundary conditions perform much better for the lift coefficient C_l than they do for the drag coefficients C_d and C_D. This behavior seems reasonable since the point-vortex model is a purely lift-based treatment and does not take into account drag at all.

In Table 2 we also compare the results obtained using the two aforementioned types of ABC's; however, the computations presented in this table were conducted on the different grids. One can see that boundary conditions (59) outperform the point-vortex ABC's in these cases as well (C_f in Table 2 is the skin friction).

We should also emphasize that the benefit of using smaller computational domains and, as a consequence, smaller grids, i.e., the grids with lesser number of nodes, is not only the direct reduction of the computational work because of the grid shrinkage but also the improvement of convergence because the grids may be chosen less stretched.

3.3 Entry-Wise Interpolation

The computational overhead associated with the usage of nonlocal ABC's (56) or (59) consists of two parts. The first part is accounted for by the matrix-vector multiplications that we do every time we need to evaluate the residuals (see Section 2). These operations add about 1–2% to the total cost of computations (more precisely, to the cost of the same number of multigrid cycles but without the nonlocal ABC's). Generally, we estimate this additional expense as low (taking into account the benefits provided by the DPM-based ABC's), and we do not think that any special effort towards reducing this cost is currently required. Nonetheless, we note that there may still be some room for reducing this cost by using the multiresolution-based techniques (see our work [19, 22]). Moreover, we should mention that the operation of matrix-vector multiplication is, as a rule, fully vectorizeable, which provides for an additional advantage of using the DPM-based ABC's on the CRAY machines. Furthermore, we expect that the new discretizations that are currently under development (see [45, 46]) will allow one to use relaxation procedures other than the methods of Runge-Kutta type that are most widely used now. As a consequence, the overhead associated with the matrix-vector multiplications (56) or (59) may be reduced since, for example, the methods of Gauss-Seidel type would require only one such operation per iteration, whereas the multi-stage Runge-Kutta methods require as many multiplications (56) or (59) as the number of stages is. Finally, we mention that the DPM-based ABC's can be implemented directly, i.e., the operation $\mathbf{T}$ can be computed explicitly (without first calculating the matrix in a basis) every time $\mathbf{u}\big|_{\nu_1}$ needs to be updated. This strategy has, in fact, been chosen for all our three-dimensional computations [19, 20, 21, 22]; the corresponding results will be briefly commented on later in this paper.

If the matrix-based strategy is used, then the second part of the total overhead due to the ABC's is accounted for by the computational cost of the operators $\mathbf{T}$ themselves (see (56) and (59)). In the case of two space dimensions, we could always keep the corresponding cost at a level of about 10% of the total work required to calculate the steady-state solution using the algorithm [33, 34, 35] with the specific multigrid strategy discussed in Section 2. Basically, this additional expense can be regarded low as well.

As mentioned above, in three space dimensions we have so far been using another strategy that did not require the calculation of matrices $\mathbf{T}$ at all. However, in the future we may need it, especially in the view of possible multiple runs. Our preliminary estimates show that in the case of three space dimensions the cost of the operator $\mathbf{T}$ may appear higher (in relative terms) than it is for two space dimensions. Therefore, we propose a special approach to decreasing the computational cost of the nonlocal DPM-based ABC's in the framework of massive computations.

Very often in CFD, one needs to calculate different flows around the same configuration of bodies; in so doing, the same grid is likely to be used. In particular, this may be the case in three space dimensions, especially as the grid generation for this case typically requires a much more substantial effort than for two space dimensions. As can be seen from our previous considerations, the operators $\mathbf{T}$ depend on the geometry, which does not change as long as the grid remains the same. These operators also depend on the coefficients of system (40) or (41), for example, on the Mach number M_0, as well as on the angle of attack α. Note, the angle of attack α formally appears neither in (40) nor in (41) since we always assume that the free-stream velocity is aligned with the positive x direction. We, however, take into account the angle of attack α by rotating the entire computational domain, see Figure 1.

Suppose now that all the computational experiments that we are going to carry out for some chosen geometry belong to a certain range of the parameters involved, say $\alpha_{min} \le \alpha \le \alpha_{max}$, $M_{0_{min}} \le M_0 \le M_{0_{max}}$. Then, we can pick up several points $\alpha^{(p)}$ within the initially prescribed range for the angle of attack and several points $M_0^{(q)}$ within the initially prescribed range for the Mach number and calculate the operator $\mathbf{T}$ for each resulting pair $\left(\alpha^{(p)}, M_0^{(q)}\right)$. Note, the same is possible for the triplets (α, M_0, Re); however, the far-field effective Reynolds number has been noticed to influence the results at a much less extent than the other two parameters. Finally, to obtain the matrix $\mathbf{T}$ for any specific pair of values (α, M_0) (within the corresponding range), we simply interpolate each entry of $\mathbf{T}$ independently with respect to the angle of attack and the Mach number between the known values for $\left(\alpha^{(p)}, M_0^{(q)}\right)$.

We have implemented this approach numerically for the same transonic turbulent flow around RAE2822 as studied above. To simplify our task on the preliminary stage, we used one-dimensional interpolation separately for α and M_0 instead of using the two-dimensional interpolation on the mesh $\left(\alpha^{(p)}, M_0^{(q)}\right)$. The results shown in Table 3 corroborate usefulness of the approach based on the interpolation of coefficients of the operators $\mathbf{T}$. From

Table 3. Interpolation of coefficients of the operator $\mathbf{T}$ for RAE2822 airfoil; $M_0 = 0.73$; $Re_0 = 6.5 \cdot 10^6$; $\alpha = 2.79°$; normal grid spacing near the airfoil $0.5 \cdot 10^{-5}$.

Domain "radius"	3 chords				50 chords	
Grid	600×104				640×128	
Type of ABC's	p.-v.	(59)	Int. M_0	Int. α	p.-v.	(59)
C_l	0.8653	0.8591	0.8593	0.8587	0.8603	0.8593
relative error	0.58%	0.02%	0.0%	0.07%	0%	0%
$C_d \times 10$	0.1203	0.1263	0.1257	0.1252	0.1255	0.1260
relative error	4.14%	0.24%	0.24%	0.6%	0%	0%
$C_D \times 10$	0.1755	0.1816	0.1811	0.1805	0.1810	0.1815
relative error	3.04%	0.05%	0.22%	0.55%	0%	0%

this table, we see that the accuracy of the solutions obtained on the basis of the interpolated matrices is almost not worse than the accuracy that we get using the genuine operator $\mathbf{T}$.

Of course, the results of Table 3 are only preliminary, and the issue of the entry-wise interpolation of the matrices $\mathbf{T}$ with respect to α and M_0 requires a thorough further study. In particular, the approach based on interpolation may have certain limitations on the size of the computational domain, as well as on the actual admissible ranges of the parameters involved. However, the initial results are encouraging. For the repeated computations, this approach can drastically decrease the overall cost of the DPM-based ABC's since it requires one substantial initial effort for calculating the matrices $\mathbf{T}$ on the mesh $\left(\alpha^{(p)}, M_0^{(q)}\right)$, and then the boundary conditions for each subsequent variant of computations (within the prescribed range) will come for almost no extra computational cost because the cost of interpolation itself is virtually negligible.

3.4 Low Mach Number Flows

We finally address another interesting aspect of implementation of the DPM-based ABC's. It is well-known that many standard explicit solvers for compressible flows encounter difficulties when directly applied to calculating the flows with low Mach numbers. The difficulties are caused by the "different scales" of eigenvalues u and $u \pm c$ (u is the flow velocity and c is the speed of sound, $|u| \ll c$), and result in the severe Courant-type limitations on the time step. One possible cure for this problem is based on the so-called local preconditioning techniques. The idea of these techniques is to change the time-evolving system (multiplying it by some non-singular matrix-preconditioner) so that the gap between the eigenvalues is narrowed but at the same time the steady state remains unchanged. An approach of this type has been recently proposed by Turkel, Fiterman, van Leer, and Vatsa, see [47, 48, 49], and has already been implemented in practice on the basis of the code [33, 34, 35].

It, however, turns out that the standard ABC's incorporated in the code [33, 34, 35] (characteristics/extrapolation) perform poorly for the case of low Mach number flows. On the other hand, boundary conditions (59) in this case demonstrate the same good performance as they show in the case of transonic flows. In Table 4, we compare numerical results obtained using two different types of ABC's for a low Mach number turbulent flow around the airfoil RAE2822. One can see that as in the previous cases, the DPM-based ABC's allow us to maintain high accuracy of computations for small computational domains.

According to the authors of [47, 48, 49], their preconditioning technique actually performs better if the governing equations are written with respect to some other equivalent set of unknowns rather than (u, v, p, ρ). As concerns the DPM-based ABC's, we do not rewrite equations (40) or (41). For those cases presented in Table 4, we have been able to obtain accurate results without any changes (except in the input data) in the boundary conditions algorithm. We, however, note that in so doing the system matrices (44) become strongly non-symmetric. The accuracy of the results from Table 4 in this case is probably due to the fact that we solve the difference AP by a direct method. However, for the small free-stream Mach numbers the use of the symmetrizers for the system matrices (for example, those presented in work [50]) may still be recommended. Moreover, in our work [20] we have constructed the three-dimensional DPM-based ABC's for the true incompressible case and then implemented these boundary

Table 4. Low Mach number turbulent flow around RAE2822 airfoil; $M_0 = 0.01$; $Re_0 = 6.5 \cdot 10^6$; $\alpha = 2.79°$; normal grid spacing near the airfoil $0.6 \cdot 10^{-5}$.

Domain "radius"	2.5 chords		20 chords	
Grid	320×64		320×64	
Type of ABC's	p.-v.	(59)	p.-v.	(59)
C_l	0.5708	0.5387	0.5419	0.5390
relative error	5.3%	0.05%	0%	0%
C_d	-0.0005	0.0016	0.0014	0.0016
relative error	???	0%	0%	0%
$C_D \times 100$	0.5676	0.7761	0.7551	0.7764
relative error	24.8%	0.03%	0%	0%

conditions along with the compressible solver for a very low Mach number ($M_0 = 0.01$). In this case, the DPM-based boundary conditions performed as well as they do for the higher subsonic and transonic Mach numbers (see [20]).

3.5 Three-Dimensional Flows

So far, we have been mostly describing the two-dimensional algorithms and the two-dimensional results. In our work [19, 20] (see also [21, 22]), we have constructed and implemented the nonlocal DPM-based ABC's for three-dimensional steady-state viscous flows (perhaps, the most important case from the standpoint of current computational practice).

As can be seen from Section 2, the ABC's algorithm basically consists of two parts. The first part may be called geometrical, it comprises the construction of the grid sets $\mathcal{M}_{in}$, $\mathcal{M}_{ex}$, $\mathcal{N}_{in}$, $\mathcal{N}_{ex}$, γ, collocation grid ω, all the necessary interpolation operations ($\mathbf{R}_{\Gamma\omega}$, $\mathbf{R}_{\omega\nu}$, $\mathbf{R}_{\nu_1\mathcal{N}_{ex}}$), and continuation $\pi_{\gamma\omega}$. The second part is computational, it consists of the cross-stream transforms (e.g., discrete Fourier) and the solution of systems (46), (48).

The principle differences between the two-dimensional case and the three-dimensional case are mostly concentrated in the first (geometrical) part, whereas the second (numerical) part changes only quantitatively. The computational geometry is obviously more cumbersome for the case of three space dimensions than it is for the case of two space dimensions. Moreover, there is a qualitative difference between interpolation along the one-dimensional curve, which constitutes the artificial boundary in two dimensions and which does not differ from the straight line when it comes to the issue of internal geometry, and interpolation along the two-dimensional curvilinear surface, which constitutes the artificial boundary in three dimensions and which has a non-trivial internal geometry. In practice, our computations [19, 20] show that although the geometrical part of the algorithm for three space dimensions is more complicated, it is still universal in the sense that the same procedure serves a variety of artificial boundaries with different shapes; moreover, this geometrical part also turns out numerically cheap.

The major difference between the two- and three-dimensional cases for the computational part of the algorithm is that we basically add another cross-stream direction (in the case of a three-dimensional wing, for example, it may be a span-wise direction). Then, we separate the variables in the AP by transforming in both cross-stream and span-wise directions and obtain a family of one-dimensional systems of type (46) with boundary conditions (48) that formally looks exactly the same although the matrices are of order five and depend on a pair of parameters (wavenumbers) rather than on only one parameter k. The numerical part of the ABC's algorithm in three space dimensions also appears universal in the sense that it does not depend on the shape of the specific computational domain. The details of the three-dimensional DPM-based algorithm, as well as various computational results, can be found in our recent paper [20]. Here, we reproduce only one numerical example.

We consider a steady-state flow of viscous compressible gas around the ONERA M6 wing. We use the NASA-developed code by Vatsa, et al. [51] to integrate the thin-layer equations on a one-block curvilinear C-O type grid generated around the wing (the geometric setup for three space dimension is delineated in our work [19, 20]). The code [51] is similar to the one [33, 34, 35] that we used for two-dimensional computations, it is based on the central-difference finite-volume discretization in space with the first- and third-order artificial dissipation. Pseudo-time

Table 5. Comparison with the standard characteristics-base boundary conditions for a turbulent flow around the ONERA M6 wing: $M_0 = 0.84$; $Re_0 = 11.7 \cdot 10^6$; $\alpha = 3.06°$.

Domain "radius"	3 root chords		10 root chords	
Grid	$197 \times 49 \times 33$		$209 \times 57 \times 33$	
Type of ABC's	standard	DPM	standard	DPM
Full lift, C_L	0.298±0.004	0.2798	0.2805	0.2786
Relative error	6.24%±1.43%	0.43%	0%	0%
Full drag, $C_D \times 10$	0.168±0.008	0.1537	0.1542	0.1531
Relative error	8.95%±5.19%	0.39%	0%	0%

iterations are used for obtaining the steady-state solution; the integration in time is done by the five-stage Runge-Kutta algorithm (with the Courant number calculated locally) supplemented by the residual smoothing. For the purpose of accelerating the convergence, the multigrid methodology is implemented; in our computations we used three subsequent grid levels with V cycles; the full multigrid methodology (FMG) could be employed as well. In addition, we use the preconditioning technique [52] to improve the convergence to steady state. We implement the DPM-based ABC's on the finest level of multigrid on the final FMG stage; the boundary data for coarser levels are provided by the coarsening procedure. As mentioned above, the implementation of the ABC's was direct, i.e., it did not require the calculation of the matrices $\mathbf{T}$. Moreover, even on the finest level we implement the DPM-based ABC's only on the first and the last Runge-Kutta stages, which seems to make very little difference compared to the implementation on all five stages; the boundary data for the three intermediate stages are provided from the DPM-based ABC's on the first stage. Unlike the two-dimensional case, the standard treatment of the external boundary in three dimensions is based on merely the locally one-dimensional characteristics analysis and extrapolation (the point-vortex model is not applicable).

We have conducted the computations for a standard three-dimensional transonic test case for the ONERA M6 wing: $M_0 = 0.84$; $Re_0 = 11.7 \cdot 10^6$; $\alpha = 3.06°$. The solution was calculated for two different computational domains of the average radii of approximately 10 and 3 root chords of the wing, respectively. The results summarized in Table 5 clearly demonstrate that for the small computational domains the DPM-based ABC's generate much more accurate solutions that the standard (characteristics-based) boundary conditions do. Note, the grids for the different domains have different dimensions, and the smaller $197 \times 49 \times 33$ grid (3 root chords) is now an exact subset of the bigger $203 \times 57 \times 33$ grid (10 root chords). This is done in order to eliminate any influence that the change of the grid in the near field could possibly exert on the solution.

Besides the improvement of accuracy, the application of the DPM-based ABC's to transonic flow computations on the small (3 root chords) computational domain yielded much higher convergence rate of the residual (continuity equation), as well as much faster convergence of other quantities, including those deemed as sensitive, e.g., the number of supersonic points in the domain. In Figures 10a and 10b, we show the convergence history for this supercritical flow variant. One can see that the convergence for the standard boundary conditions is poor; therefore the corresponding force coefficients in Table 5 are given with the error bands indicated.

For the 10 root chords domain, the DPM-based ABC's also provide for some convergence speedup, although the difference between the two ABC's techniques is less dramatic here. This is reasonable because one could generally expect that the bigger the computational domain, the smaller is the influence that the external boundary conditions exert on the numerical procedure. The convergence history for the 10 root chords computations is shown in Figures 11a and 11b.

Note, from Figure 10b one can conclude that on the small domain the two algorithms converge to quite different solutions, whereas Figure 11b allows one to assume that on the big domain the final solutions are close to one another. The data from Table 5 corroborate these conclusions. This behavior again fits into the aforementioned concept that the impact of the ABC's decreases as the domain size increases.

As concerns the computational cost of the three-dimensional DPM-based ABC's, we note that by ap-

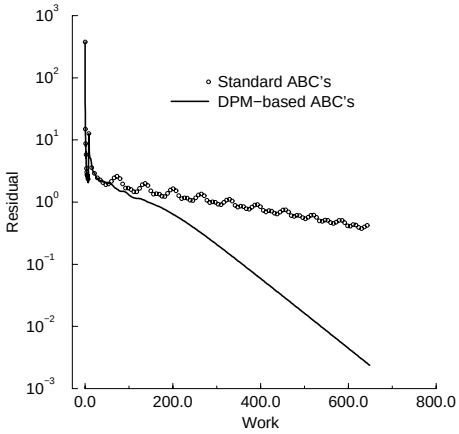

Figure 10a. ONERA M6: $M_0 = 0.84$, $Re_0 = 11.7 \cdot 10^6$, $\alpha = 3.06°$. Convergence history for the residual of the continuity equation. Average domain "radius" is 3 root chords of the wing; grid $197 \times 49 \times 33$.

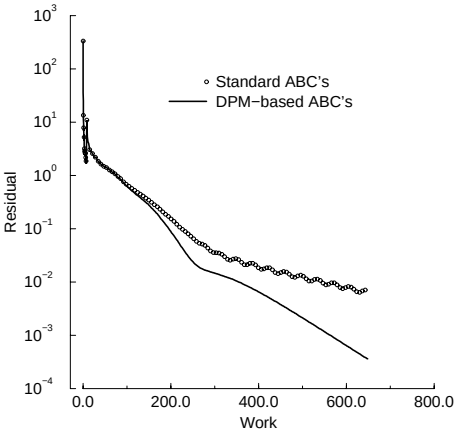

Figure 11a. ONERA M6: $M_0 = 0.84$, $Re_0 = 11.7 \cdot 10^6$, $\alpha = 3.06°$. Convergence history for the residual of the continuity equation. Average domain "radius" is 10 root chords of the wing; grid $209 \times 57 \times 33$.

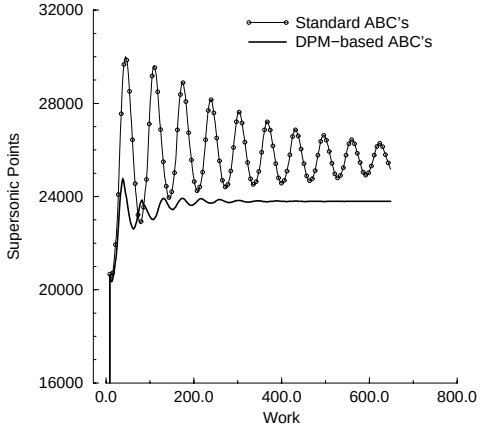

Figure 10b. ONERA M6: $M_0 = 0.84$, $Re_0 = 11.7 \cdot 10^6$, $\alpha = 3.06°$. Convergence history for the number of supersonic nodes in the domain. Average domain "radius" is 3 root chords of the wing; grid $197 \times 49 \times 33$.

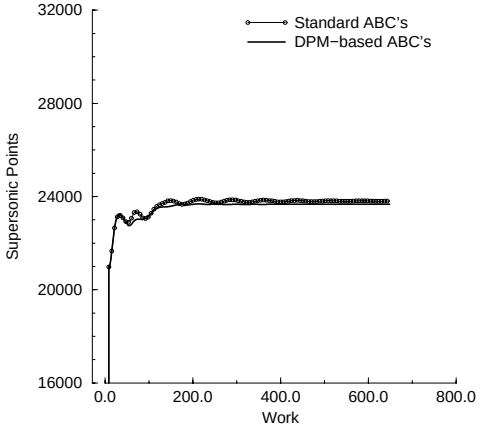

Figure 11b. ONERA M6: $M_0 = 0.84$, $Re_0 = 11.7 \cdot 10^6$, $\alpha = 3.06°$. Convergence history for the number of supersonic nodes in the domain. Average domain "radius" is 10 root chords of the wing; grid $209 \times 57 \times 33$.

plying this procedure only on the first and the last Runge-Kutta stages and only on the finest multigrid level, the total number of the required calculations of generalized potential has been brought to a minimum. In so doing, the average cost of application of the DPM-based ABC's adds about 20–25% of the CPU time to the cost of the same procedure with the standard (characteristics-based) boundary conditions. This extra expense is not high (taking into account the improvement of accuracy); moreover, it can often be compensated for and even noticeably prevailed over by the convergence acceleration and the reduction of the domain size. Besides, to explicitly decrease the computational cost associated with the DPM-based ABC's the entry-wise interpolation of boundary operators (see above) and/or the multiresolution-based methodologies (see [19, 22]) can be used. We expect that the latter can also be employed when implementing the DPM-based ABC's for multi-block grids.

4 Concluding Remarks

The DPM-based approach provides for a geometrically universal and robust means to set the ABC's for steady-state external flow computations. These ABC's enable one to essentially decrease the size of

the computational domain (for the case of steady-state viscous flows) in comparison with the domains that can be used with standard boundary conditions. This implies the possibility to improve the accuracy of computations without increasing their cost or to reduce the total cost of computations without decreasing the accuracy. Moreover, the total computational cost may also be essentially reduced because of the convergence speedup that is accounted for by usage of the DPM-based ABC's. We also emphasize that the DPM-based ABC's apply to the computational domains of irregular shape with equal ease and that the practical implementation of these boundary conditions is easy from the algorithmic standpoint. In particular, it takes a relatively little effort to supplement some already existing code by the new DPM-based ABC's. Altogether, these properties make the DPM-based ABC's a very attractive tool for external flow computations.

The next big challenge would, of course, be extending the DPM-based approach to the case of time-dependent problems. As mentioned above, a particular class of such problems, namely, the fluid flows oscillating in time, has been studied in [17]. In practice, this formulation originates, e.g., from the well-known problem of pitching airfoil. Analogously to the steady-state case, the ABC's of [17] also connect the values of the solution at the penultimate and outermost rows of the grid (e.g., C-grid); however, it is done for the entire time interval T equal to one period. To obtain the ABC's in [17], we first approximate the data on , $\times [t_0, t_0 + T]$ (t_0 is arbitrary) by some periodic in time vector-function in the sense of least squares. The approximant obviously appears to be the Fourier series of the actual data on , $\times [t_0, t_0 + T]$. Assuming that the period T is known in advance, and that there are no other essential time-dependent effects in the model, we implement the Fourier transform in time and end up with the family of "steady-state" systems. Each member of this family has, generally speaking, complex coefficients but can nevertheless be treated analogously to how it is done above, so that the resulting boundary conditions in the frequency domain are obtained in the form (59). Then, implementing the inverse Fourier transform we obtain the ABC's in time domain. When the problem is discretized in time, both direct and inverse transforms are discrete as well, and the aforementioned family of "steady-state" systems is finite, which gives us a way to practically calculate the ABC's. Clearly, the DPM-based ABC's of [17] appear to be nonlocal in both space and time. However, the nonlocality in time is limited by the interval T equal to one period.

As concerns the case of general time-dependent flows, for which we do not do any initial assumptions (like periodicity) about the behavior of the solution, it is, of course, more complicated and more difficult to handle. The main obstacle here is the nonlocality of the ABC's in time. Unlike the periodic case, the exact ABC's for the general time-dependent problem would formally require storing all the preceding information on the artificial boundary from $t = 0$ to $t = t_{final}$. As the solution develops in time, such boundary conditions would become more and more expensive from the standpoints of both memory and computer time, which is required for processing the constantly growing amount of boundary data. The estimated high computational cost severely limits the possibilities of developing the exact ABC's for time-dependent problems and makes most of the available constructions of such boundary conditions practically infeasible for any long-term runs.

Clearly, the crucial issue for any time-dependent ABC's algorithm is how to effectively restrict the nonlocality of the boundary conditions in time. (In time-periodic formulation of [17], this restriction was incorporated in the formulation of the problem from the very beginning.) A promising approach based on implementation of the Laplace transform in time and then on usage of special recursion relations for calculating the convolutions with the kernels of nonlocal operators has recently been proposed by Sofronov in [53]. The limitation of this technique is the requirement that the boundary should be of some regular, e.g., linear, shape. There are several other approaches to this problem, none of which has been studied thoroughly yet, although each one may in principle appear useful. One approach is based on the idea of an artificial periodic formulation in time. It is analogous to what we do for the cross-stream space coordinates in the steady-state problem. Basically, we introduce some error, which is controlled by the value of the period, and in so doing obtain finite formulation of the problem which is available for the numerical treatment (see Section 2). Note, the idea of an artificial periodic formulation in time was earlier proposed by Lax in [54]. Other possible approaches could be based on some properties of solutions relevant to particular classes of equations (systems). For example, one can make use of lacunas that exist in the solutions of hyperbolic equations, or exploit the exponential decay of coefficients of the Green operator as the time interval increases, which is relevant to parabolic equations. More details on these and analogous approaches can be found in the work [11], in which they have been first proposed, as well as in the reviews [1, 2].

We emphasize that the issue of ABC's for time-dependent problems is important not only in CFD, but in many other areas of scientific computing. For example, the problems of computational acoustics and aeroacoustics (Helmholtz, wave, or linearized Euler equations), as well as the problems of electromagnetic waves propagation (Maxwell equations), present significant mathematical interest and also attract more and more attention of the practitioners. Effective algorithms for handling unbounded domains are required for such problems in both single-mode (Helmholtz-type) and wide-band (as a rule, time domain) formulations. For the time domain algorithms, the restriction of nonlocality of the ABC's in time obviously acquires particular importance and presents a major challenge for the future research. Among many interesting links that connect this problem to other areas of computational mathematics we would like to point out one. Namely, one of the principle elements of the construction of highly accurate and numerically efficient DPM-based ABC's for time-dependent problems would be to calculate the Calderon projections for the schemes that are able of clear capturing the lacunas relevant to the solutions of hyperbolic equations (systems). In turn, the ideas for constructing such schemes seem to be closely related to another group of ideas and techniques associated with the genuinely multidimensional methods. The latter are intensively studied now, especially in CFD, where these methods present a major hope for the drastic increase of efficiency of the multigrid solvers (see [46]).

Acknowledgments

The authors are very thankful to Saul Abarbanel of Tel-Aviv University, Alvin Bayliss of Northwestern University, David Gottlieb of Brown University, Bertil Gustafsson of Uppsala University, Dmitrii Kamenetskii of Keldysh Institute for Applied Mathematics, Russian Academy of Sciences, David Sidilkover of ICASE, NASA Langley Research Center, Ivan Sofronov of Keldysh Institute for Applied Mathematics, Russian Academy of Sciences, R. Charles Swanson of NASA Langley Research Center, James L. Thomas of NASA Langley Research Center, Eli Turkel of Tel-Aviv University, and Veer Vatsa of NASA Langley Research Center for the numerous fruitful discussions on the subject of this paper and for their most helpful suggestions and comments. Our special thanks to Eli Turkel, R. Charles Swanson, and Veer Vatsa for giving us an opportunity to use their multigrid Navier-Stokes codes and to Edward Parlette of Vigyan, Inc. for his most valuable help in generating the three-dimensional grids.

We are most grateful to the Editor of the volume Mohamed Hafez of UC Davis for inviting and encouraging us to write this review.

References

1. Tsynkov, S. V., "Artificial Boundary Conditions Based on the Difference Potentials Method," NASA Technical Memorandum No. 110265, Langley Research Center, July 1996.
2. Tsynkov, S. V., "Numerical Solution of Problems on Unbounded Domains and Application of the Difference Potentials Method," submitted to *Applied Numerical Mathematics*.
3. Givoli, D., "Non-reflecting Boundary Conditions," *Journal of Computational Physics*, 94 (1991) pp. 1–29.
4. Givoli, D., *Numerical Methods for Problems in Infinite Domains*, Elsevier, Amsterdam, 1992.
5. Calderon, A. P., "Boundary-Value Problems for Elliptic Equations," in *Proceedings of the Soviet-American Conference on Partial Differential Equations at Novosibirsk*, Fizmatgiz, Moscow, 1963, pp. 303–304.
6. Seeley, R. T., "Singular Integrals and Boundary Value Problems," *American Journal of Mathematics*, 88 (1966) pp. 781–809.
7. Ryaben'kii, V. S., "Boundary Equations with Projections," *Russian Mathematical Surveys*, 40 (1985) pp. 147–183.
8. Ryaben'kii, V. S., *Difference Potentials Method for Some Problems of Continuous Media Mechanics*, Nauka, Moscow, 1987 (in Russian).
9. Ryaben'kii, V. S., "Difference Potentials Method and its Applications," *Math. Nachr.*, 177 (1996) pp. 251–264.
10. Mikhlin, S. G., Morozov, N. F., and Paukshto, M. V., "The Integral Equations of the Theory of Elasticity," B. G. Teubner Verlagsgesellschaft, Stuttgart, 1995.
11. Ryaben'kii, V. S., "Exact Transfer of Boundary Conditions," *Computational Mechanics of Solids,* Issue 1 (1990) pp. 129–145 (in Russian).
12. Tsynkov, S. V., "Boundary Conditions at the External Boundary of the Computational Domain for Subsonic Problems in Computational Fluid Dynamics," Keldysh Institute of Applied Mathematics, U.S.S.R. Academy of Sciences, Preprint 108, Moscow, 1990 (in Russian).

13. Tsynkov, S. V., "An Implementation of the Potential Flow Model in Setting the External Boundary Conditions for the Euler Equations. Part I," Keldysh Institute of Applied Mathematics, U.S.S.R. Academy of Sciences, Preprint 40, Moscow, 1991 (in Russian).

14. Sofronov, I. L., and Tsynkov, S. V., "An Implementation of the Potential Flow Model in Setting the External Boundary Conditions for the Euler Equations. Part II," Keldysh Institute of Applied Mathematics, U.S.S.R. Academy of Sciences, Preprint 41, Moscow, 1991 (in Russian).

15. Sofronov, I. L., "A Rapidly Converging Method for Solving the Euler Equation," *Computational Mathematics and Mathematical Physics*, 31 (1991) pp. 66–78.

16. Godunov, S. K., and Gordienko, V. M., "The Green Matrix of a Boundary-Value Problem for Ordinary Differential Equations," *Russian Mathematical Surveys*, 39 (1984) pp. 45–85.

17. Tsynkov, S. V., "Artificial Boundary Conditions for Computation of Oscillating External Flows," to appear in *SIAM Journal on Scientific Computing* (1997). Also see: NASA Technical Memorandum No. 4714, Langley Research Center, August 1996.

18. 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*, 32 (1995) pp. 1355–1389.

19. Vatsa, V. N., and Tsynkov, S. V., "An Improved Treatment of External Boundary for Three-Dimensional Flow Computations," AIAA Paper No. 97-2074, June 1997.

20. Tsynkov, S. V., "External Boundary Conditions for Three-Dimensional Problems of Computational Aerodynamics," NASA Technical Memorandum No. 110337, Langley Research Center, March 1997; also submitted to *SIAM Journal on Scientific Computing*.

21. Tsynkov, S. V., "Nonlocal Artificial Boundary Conditions for Computation of External Viscous Flows," in: Computational Fluid Dynamics'96, Proceedings of the Third ECCOMAS CFD Conference, September 9–13, 1996, Paris, France, J.-A. Desideri, C. Hirsch, P. Le Tallec, M. Pandolfi, and J. Périaux, eds., John Wiley & Sons, 1996, pp. 512–518.

22. Tsynkov, S. V., "Artificial Boundary Conditions for Infinite-Domain Problems," to appear in Proceedings of the ICASE/LaRC Workshop on Barriers and Challenges in Computational Fluid Dynamics, Hampton, August 5–7, 1996, M. D. Salas et al., eds., Kluwer Academic Publishers, 1997.

23. Hörmander, L., *Linear Partial Differential Operators*, Springer-Verlag, Berlin, 1963.

24. Vladimirov, V. S., *Equations of Mathematical Physics*, Dekker, New York, 1971.

25. Reznik, A. A., "Approximation of Surface Potentials of Elliptic Operators by Difference Potentials," *Soviet Mathematics Doklady*, 25 (1982) pp. 543–545.

26. Tsynkov, S. V., "An Application of Nonlocal External Conditions to Viscous Flow Computations," *Journal of Computational Physics*, 116 (1995) pp. 212–225.

27. Tsynkov, S. V., Turkel, E., and Abarbanel, S., "External Flow Computations Using Global Boundary Conditions," *AIAA Journal*, 34 (1996) pp. 700-706. Also see: AIAA Paper No. 95-0562, January 1995.

28. Tsynkov, S. V., "Construction of Artificial Boundary Conditions Using Difference Potentials Method," to appear in *Proceedings of the International Conference on Optimization of Finite-Element Approximations,* June 25–29, 1995, St.-Petersburg, Russia.

29. Tsynkov, S. V., "Nonlocal Artificial Boundary Conditions Based on the Difference Potentials Method," in *Proceedings of the Sixth International Symposium on Computational Fluid Dynamics,* IV, pp. 114–119, September 4–8, 1995, Lake Tahoe, Nevada.

30. Ryaben'kii, V. S., "Necessary and Sufficient Conditions for Good Definition of Boundary Value Problems for Systems of Ordinary Difference Equations," *U.S.S.R. Computational Mathematics and Mathematical Physics*, 4 (1964) pp. 43–61.

31. Ryaben'kii, V. S., and Tsynkov, S. V., "An Effective Numerical Technique for Solving a Special Class of Ordinary Difference Equations," *Applied Numerical Mathematics*, 18 (1995) pp. 489–501.

32. Agoshkov, V. I., "Domain Decomposition Techniques for Problems in Mathematical Physics,"

in *Computational Processes and Systems,* issue 8, G. I. Marchuk, ed., Nauka, Moscow, 1990, pp. 3–51 (in Russian).

33. Swanson, R. C., and Turkel, E., "A Multistage Time-Stepping Scheme for the Navier-Stokes Equations," AIAA Paper 85-0035, January 1985.

34. 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.

35. Swanson, R. C., and Turkel, E., "Multistage Schemes with Multigrid for the Euler and Navier-Stokes Equations. Volume I: Components and Analysis," NASA Technical Paper No. 3631, Langley Research Center, 1997.

36. Thomas, J. L., and Salas, M. D., "Far-Field Boundary Conditions for Transonic Lifting Solutions to the Euler Equations," AIAA Paper 85-0020, January 1985.

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

38. Schlichting, H., *Boundary Layer Theory*, McGraw-Hill, New York, 1968.

39. Ferm, L., "Non-Reflecting Accurate Open Boundary Conditions for the Steady Euler Equations," Technical Report 143, Department of Scientific Computing, Uppsala University, Uppsala, Sweden, September 14, 1992.

40. Ferm, L., and Gustafsson, B., "A Downstream Boundary Procedure for the Euler Equations", *Computers and Fluids*, 10 (1982) pp. 261–276.

41. Ferm, L., "Modified External Boundary Conditions for the Steady Euler Equations," Technical Report 153, Department of Scientific Computing, Uppsala University, Uppsala, Sweden, August 25, 1993.

42. Ferm, L., "Open Boundary Conditions for External Flow Problems," *Journal of Computational Physics*, 91 (1990) pp. 55–70.

43. Engquist, B., and Halpern, L., "Far Field Boundary Conditions for Computation over Long Time," *Applied Numerical Mathematics*, 4 (1988) pp. 21–45.

44. Ferm, L., "Multigrid for External Flow Problems," Technical Report, Department of Scientific Computing, Uppsala University, Uppsala, Sweden, September 9, 1993.

45. Ta'asan, S., "Canonical-Variables Multigrid Method for Steady-State Euler Equations," ICASE Report 94-14, NASA Langley Research Center, Hampton, VA, U.S.A., 1994.

46. Sidilkover, D., "A Genuinely Multidimensional Upwind Scheme and Efficient Multigrid Solver for the Compressible Euler Equations," ICASE Report No. 94-84, NASA Langley Research Center, Hampton, VA, U.S.A., 1994.

47. Turkel, E., "Review of Preconditioning Methods for Fluid Dynamics," *Applied Numerical Mathematics*, 12 (1993) pp. 257–284.

48. Turkel, E., Fiterman, A., and van Leer, B., "Preconditioning and the Limit of the Compressible to the Incompressible Flow Equations for Finite Difference Schemes," in *Frontiers of Computational Fluid Dynamics 1994*, Caughey, D. A., and Hafez, M. M., eds., John Wiley and Sons, 1994, pp. 215–234.

49. Fiterman, A., Turkel, E., and Vatsa, V., "Pressure Updating Methods for Steady-State Fluid Equations," AIAA paper 95-1652, June 1995.

50. Abarbanel, S. and Gottlieb, D., "Optimal Time Splitting for Two- and Three-Dimensional Navier-Stokes Equations with Mixed Derivatives," *J. Comput. Phys.,* 41 (1981) pp. 1–33.

51. Vatsa, V. N., Sanetrik, M. D., and Parlette, E. B., "Development of a Flexible and Efficient Multigrid-Based Multiblock Flow Solver," AIAA Paper 93–0677, January 1993.

52. Turkel, E., Vatsa, V. N., and Radespiel, R., "Preconditioning Methods for Low-Speed Flows," AIAA Paper 96–2460–CP, June 1996.

53. Sofronov, I. L., "Transparent Boundary Conditions for Unsteady Transonic Flow Problems in Wind Tunnel," Universität Stuttgart, Mathematisches Institut A, Preprint 95-21, Stuttgart, Germany, 1995.

54. Lax, P. D., "On Cauchy's Problem for Hyperbolic Equations and the Differentiability of Solutions of Elliptic Equations," *Communications of Pure and Applied Mathematics*, 8 (1955) pp. 615–633.