JOURNAL OF COMPUTATIONAL PHYSICS 116, 212–225 (1995)

An Application of Nonlocal External Conditions to Viscous Flow Computations

S. V. TSYNKOV

Department of Applied Mathematics, School of Mathematical Sciences, Tel-Aviv University, Ramat-Aviv, Tel-Aviv 69978, Israel

Received January 12, 1994; revised July 12, 1994

We are looking for a steady-state solution of an external flow problem originally formulated on an unbounded domain. Our case is a 2D viscous compressible flow past a finite body (airfoil). We truncate the original domain by introducing a finite grid around the airfoil and integrate the Navier–Stokes equations on this grid with the help of a finite-volume code which involves a multigrid pseudo-time iteration technique for achieving a steady state. To integrate the Navier–Stokes equations on a finite subregion of an original domain only we supplement the numerical algorithm by special nonlocal artificial boundary conditions formulated on an external boundary of the finite computational domain. These artificial boundary conditions are based on the difference potentials method proposed by V. S. Ryaben'kii. We compare the results provided by the nonlocal conditions with those obtained from the standard external conditions which are based on locally one-dimensional characteristic analysis at inflow and extrapolation at outflow. It turns out that the nonlocal artificial boundary conditions accelerate the convergence by about a factor of 3, as well as allow one to shrink substantially the computational domain without loss of accuracy.

1. INTRODUCTION

Many fields of science and engineering currently use numerical algorithms to supplement experimental results or to simulate processes that are hard to investigate experimentally. One field where numerical algorithms have become one of the dominant research tools is fluid dynamics with particular emphasis on aerodynamics.

In computational fluid dynamics (CFD) one is often interested in flows exterior to bodies (e.g., airfoils). Therefore the domain in which we wish to solve the equations extends to infinity. To numerically solve the fluid equations we construct a grid in the domain. However, since the domain is infinite no finite grid can be constructed. The standard procedure is to introduce an artificial outer boundary so that the new domain is now finite. This has to be done in a manner that keeps the error caused by the domain truncation to a minimum. On this artificial outer boundary one needs to specify boundary conditions so that the partial differential equations are well posed and the solution obtained inside the truncated domain is close to the corresponding fragment of the original solution. A new approach to the construction of artificial boundary conditions (ABCs) for flow problems on infinite domains was proposed in [19]. Here we will implement these conditions to some viscous flow computations and discuss the corresponding numerical results.

Computational practice typically demands that any algorithm of ABCs satisfies two groups of restrictions which to a certain extent are contradictory. On one hand, ABCs should be simple in their numerical implementation, cheap from the viewpoint of consuming computer resources and applicable on an artificial boundary of irregular shape. The last property is of special significance. Indeed, the shape of an artificial boundary is actually determined by the grid generated inside the computational domain. Since the grid is typically fitted to the (inner) solid boundary(ies) then its outer boundary may have a rather complicated form which cannot easily be modified for the convenience of the ABCs. All these requirements usually lead to the use of *local boundary conditions*. These can be based on several ideas, e.g., on the analysis of characteristic variables, extrapolation, or some other related ideas (see, e.g., [15, 23] as well as the reviews [12, 13] for more details). On the other hand, *exact boundary conditions*, which give zero truncation, are nonlocal (in space for stationary problems and also in time for time-dependent ones). This can easily be seen from the analysis of model examples (Laplace, Helmholtz, wave equations) [12, 13].

One of the simplest model cases is the following. Consider the 2D Poisson equation

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right)+\frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2}=f(r,\theta) \tag{1.1}$$

and assume that the right-hand side is a continuous function with compact support, $\operatorname{supp} f(r,\theta)\subseteq B, f=0$ outside the finite set B. We want to find a solution to (1.1) which is zero at infinity. Such a solution exists and is unique if we require that $\int_B f\,d\sigma=0$ ($d\sigma$ is an area element). Actually, this solution is represented by an area Newton potential with the density $f(r,\theta)$.

We now introduce a circular artificial boundary $r=R_0$ enclosing B. Then Eq. (1.1) is transformed into the homogeneous Laplace equation on an unbounded exterior to the disk $r\le R_0$.

0021-9991/95 $6.00

We are going to look for the solution to (1.1) only on the finite domain $r \leq R_0$. What kind of ABCs should we impose at $r = R_0$ to ensure that the solution found for $r \leq R_0$ will coincide there with the original solution obtained on R^2 and truncated to this disk?

We Fourier transform (1.1) with respect to θ. Since the equation is homogeneous for $r \geq R_0$, we obtain

$$\frac{1}{r}\frac{d}{dr}\left(r\frac{d\hat{u}_k}{dr}\right) - \frac{k^2}{r^2}\hat{u}_k = 0, \qquad \hat{u}_k = \hat{u}_k(r), \quad k = 0, \pm 1, \pm 2, \ldots, r \geq R_0. \tag{1.2}$$

Since (1.2) is a second-order ODE it has two linearly independent solutions: $\hat{u}_k = r^{-|k|}$ and $\hat{u}_k = r^{|k|}$. The first solution vanishes as $r \to +\infty$, whereas the second one grows without bounds. Since we are looking for a solution vanishing at infinity, we have to prohibit growing modes. Therefore, we use the following conditions:

$$\left.\frac{d\hat{u}_k}{dr} + \frac{|k|}{r}\hat{u}_k\right|_{r=R_0} = 0, \quad k = 0, \pm 1, \pm 2, \ldots, \hat{u}_0(R_0) = 0. \tag{1.3}$$

It is easy to show (see [20, p. 202]) that conditions (1.3) are actually *the exact ABCs*. This means that an original unbounded problem and a new problem on $r \leq R_0$ with conditions (1.3) are equivalent; i.e., their solutions coincide on $r \leq R_0$. An important feature of ABCs (1.3) is the following: an inverse Fourier transform of (1.3) yields a nonlocal pseudodifferential equation for the physical variable $u(R_0, \theta)$ since (1.3) contains $|k|$.

The examples of nonlocal boundary conditions for some flow computations could be found in [8], where the author introduces artificial boundary of an elliptic shape and calculates an inviscid compressible flow past an airfoil, as well as in [10, 9] where analogous boundary conditions (based on the Fourier expansion) are constructed and implemented for computation of the inviscid compressible duct flow. We refer to the reviews [12, 13] for other (not only hydrodynamic) examples and to [5–7] for rigorous analysis of some specific (time-dependent) problems. It is noteworthy that the same situation also occurs for more complicated cases (arising from physical applications). Generally, for both stationary problems as well as for time dependent problems: *exact ABCs are nonlocal.*

In parallel with the better accuracy of approximation, nonlocal ABCs may also provide for stationary problems a much more rapid convergence to steady state when one uses some iteration procedure for computation of the solution (see below). However, nonlocal conditions usually present difficulties in numerical implementation, require considerable computer resources and can be derived easily only for regular boundaries (rectangular, circular, etc.). The last restriction is caused by the techniques most commonly used to derive nonlocal ABCs. These techniques require either implementation of some integral transformation (Fourier, Laplace) or knowledge of an explicit expression for the Green function [11].

To avoid the computational problems connected with nonlocal conditions, one commonly uses some local approximations to such ABCs. For example, by developing rational (e.g., Padé) approximations to the symbols of the pseudo-differential operators (ΨDOs) involved (e.g., $|k|$ in (1.3)), see [3, 5–7, 14, 17, 24]. In doing so one usually obtains (instead of a pseudodifferential equation) high order local differential relations at the boundary. Using these as ABCs can cause instabilities and/or ill-posedness of the truncated problem which presents additional difficulties for computations. High order local ABCs can also be derived independently, without referring to the approximation of ΨDO symbols. Namely, an alternative approach may be based on asymptotic expansion of the solution. Truncation of this expansion to a finite series leads to a local nonreflecting boundary condition. This approach was implemented in [1–4]. Note, that such independently obtained local conditions can sometimes be considered as the approximation to a nonlocal condition.

A practical conclusion which can be drawn here is that one has to choose an optimal computational strategy for any specific problem. Generally, the better the accuracy of the approximation to the original solution, and/or the faster the convergence one aims to achieve, the more sophisticated are the ABCs (truly nonlocal or higher-order local) to be developed and therefore the greater are the computational difficulties encountered.

In this paper we aim to avoid the difficulties outlined above and to develop such ABCs which would combine the advantages of local and nonlocal ones. We no longer consider model examples but real CFD problems, namely compressible viscous flows. We construct such nonlocal ABCs that provide a good approximation as well as very fast convergence and, at the same time, are easy to use, do not require large additional resources, and apply to the boundaries of any irregular shape.

The material below is organized as follows. In Section 2 we briefly describe the algorithm for ABCs. Section 3 is devoted to the numerical results. Finally, some conclusions and possible generalizations are summarized in Section 4.

2. DESCRIPTION OF ALGORITHM

Our approach is essentially based on the ideas from [19]. We use the technique called the *difference potentials method* (DPM) [20] to equivalently reformulate the problem from the domain to its boundary. This technique does not require knowledge of either the fundamental solution or Green's function and is applicable to irregular boundaries with equal facility.

We consider the plane flow of a perfect compressible viscous gas past an airfoil. This flow is assumed to be stationary and subsonic at infinity. Such a flow is governed by the full Navier–Stokes equations and we use certain finite-difference technique (see below for details) to integrate them on a C-type curvilinear grid generated around the airfoil. Denote by D_{in} the computa-

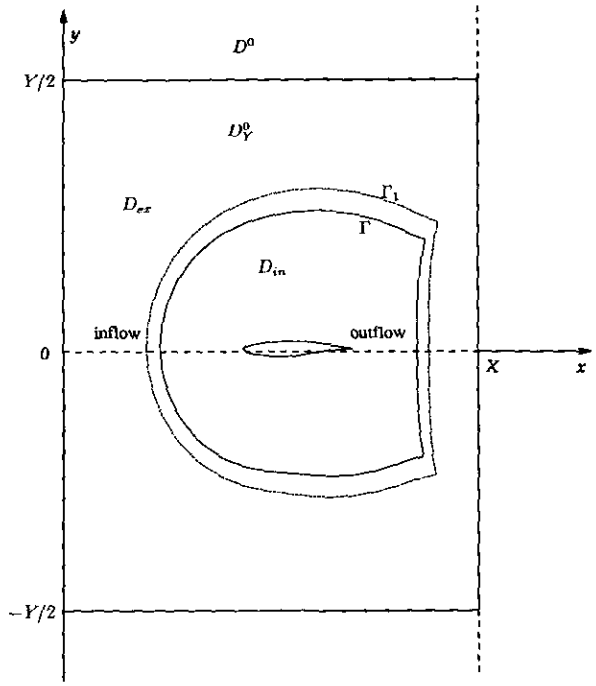

FIG. 1. Configuration of domains.

tional domain covered by this grid, Γ is its external boundary, and D_{ex} is the unbounded exterior to D_{in}. The geometric setup is shown in Fig. 1.

We use the following *fundamental assumption* in our approach: the deviations of local flow parameters from free stream ones are small in the far field, i.e., in the domain D_{ex}. Consequently, the equations governing these perturbations can be considered as linear. The dimensionless and linearized (around the free stream background) Navier–Stokes system takes the form

$$\begin{aligned}
&\frac{\partial \rho}{\partial x}+\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}=0\\
&\frac{\partial u}{\partial x}+\frac{\partial p}{\partial x}-\frac{1}{\mathrm{Re}}\left[\frac{4}{3}\frac{\partial^2 u}{\partial x^2}+\frac{1}{3}\frac{\partial^2 v}{\partial x\partial y}+\frac{\partial^2 u}{\partial y^2}\right]=0\\
&\frac{\partial v}{\partial x}+\frac{\partial p}{\partial y}-\frac{1}{\mathrm{Re}}\left[\frac{4}{3}\frac{\partial^2 v}{\partial y^2}+\frac{1}{3}\frac{\partial^2 u}{\partial x\partial y}+\frac{\partial^2 v}{\partial x^2}\right]=0\\
&\frac{\partial p}{\partial x}-\frac{1}{M_0^2}\frac{\partial \rho}{\partial x}-\frac{\gamma}{\mathrm{RePr}}\left[\Delta p-\frac{1}{\gamma M_0^2}\Delta \rho\right]=0,
\end{aligned} \tag{2.1}$$

where u, v, p, ρ are the deviations of the Cartesian velocity components, pressure and density, respectively, from the corresponding free stream values; $M_0 = u_0(\gamma(p_0/\rho_0)^{-1/2}$, $\mathrm{Re} = \rho_0 u_0 L/\mu_0$, $\mathrm{Pr} = \mu_0 c_p/\kappa_0$ are the Mach, Reynolds, and Prandtl numbers, respectively, at the free stream (parameters with subscript "0"). The following scales were used for nondimensionalization: u_0 for velocity, ρ_0 for density, $\rho_0 u_0^2$ for pressure, u_0^2 for internal energy. To obtain (2.1) we retain only the first-order terms with respect to deviations in the original system, use the equation of state $\varepsilon = (1/(\gamma - 1))p/\rho$ to eliminate the internal energy ε, and assume that the flow direction at infinity coincides with the x-axis. Δ denotes the Laplace operator.

The boundary condition at infinity is the vanishing of all the unknown variables,

$$\rho \to 0,\ u \to 0,\ v \to 0,\ p \to 0 \quad \text{as } x^2 + y^2 \to \infty. \tag{2.2}$$

which simply corresponds to the free stream limit of the solution at infinity.

The possibility of linearization in the far field is a rather natural assumption for external aerodynamic problems. It arises from numerous physical and numerical experiments. The principal question is how close to the body can one locate an artificial boundary Γ and still provide the desired accurate solutions. Computational practice is the main means of answering this question. While analyzing some numerical results we show that for different types of ABCs the admissible distance between the body and the artificial boundary may vary greatly. The nonlocal ABCs based on the linearization and application of the DPM [20] permit placing an artificial boundary closer to the airfoil than extrapolation conditions do (see below).

We will develop ABCs directly for the finite-difference case. First, introduce an auxiliary domain $D_Y^0 = (0, X) \times (-Y/2, Y/2)$ of rectangular shape fully containing D_{in} (see Fig. 1). We solve on D_Y^0 the difference *auxiliary problem* (AP) for the inhomogeneous version of (2.1). The right-hand side $f(x, y)$ for this problem will be specified later. It is a function with compact support, $\mathrm{supp}\, f(x, y)$ is concentrated near Γ. Consider a uniform grid in each Cartesian direction inside D_Y^0 (with mesh sizes h_x and h_y, respectively) and construct a second-order finite-difference approximation to (2.1) on this grid using central differences (of corresponding order) for the first- and second-order derivatives.[1] The resulting finite-difference system is written out explicitly in [19]. For reasons stated below (see also [19] for more details) we assume periodic boundary conditions in the y-direction,

$$\begin{aligned}
&\mathbf{u}^0_{m,0} = \mathbf{u}^0_{m,2J+1}, \quad m = 0, \ldots, M,\\
&\mathbf{u}^0_{m,-1} = \mathbf{u}^0_{m,2J}, \quad m = 0, \ldots, M,
\end{aligned} \tag{2.3}$$

where $\mathbf{u}^0_{m,j} = (u_0, v_0, p_0, \rho_0)^T_{m,j}$ denotes the vector of unknowns for the AP (superscript "0" hereafter), $M + 1$ and $2J + 1$ are the number of nodes of the Cartesian grid in the x and y directions, respectively. We then introduce a certain compactly

[1] In principle one may use a higher order finite-difference approximation to (2.1) when the basic algorithm used for integrating the Navier–Stokes equations inside D_{in} is of higher order. In this paper, however, the basic algorithm, see [16, 21, 22], is second-order accurate.

supported right-hand side $\mathbf{f}^0_{m,j} = (f^0_1, f^0_2, f^0_3, f^0_4)^T_{m,j}$ (see below), apply a discrete Fourier transform with respect to y, and for each wavenumber k, $k = -J, ..., J$, get the following second-order system of ordinary difference equations ("^" designates Fourier variables):

$$\frac{\hat\rho^0_{m+1,k} - \hat\rho^0_{m-1,k}}{2h_x} + \frac{\hat u^0_{m+1,k} - \hat u^0_{m-1,k}}{2h_x} + s_k \hat v^0_{m,k} = \hat f^0_{1_{m,k}}$$

$$\frac{\hat u^0_{m+1,k} - \hat u^0_{m-1,k}}{2h_x} + \frac{\hat p^0_{m+1,k} - \hat p^0_{m-1,k}}{2h_x}$$

$$- \frac{1}{\mathrm{Re}}\left[\frac{4}{3}\frac{\hat u^0_{m+1,k} - 2\hat u^0_{m,k} + \hat u^0_{m-1,k}}{h_x^2}\right.$$

$$\left. + \frac{1}{3} s_k \frac{\hat v^0_{m+1,k} - \hat v^0_{m-1,k}}{2h_x} - t_k \hat u^0_{m,k}\right] = \hat f^0_{2_{m,k}}$$

$$\frac{\hat v^0_{m+1,k} - \hat v^0_{m-1,k}}{2h_x} + s_k \hat p^0_{m,k} - \frac{1}{\mathrm{Re}}\left[\frac{\hat v^0_{m+1,k} - 2\hat v^0_{m,k} + \hat v^0_{m-1,k}}{h_x^2}\right.$$

$$\left. + \frac{1}{3} s_k \frac{\hat u^0_{m+1,k} - \hat u^0_{m-1,k}}{2h_x} - \frac{4}{3} t_k \hat v^0_{m,k}\right] = \hat f^0_{3_{m,k}} \qquad (2.4)$$

$$\frac{\hat p^0_{m+1,k} - \hat p^0_{m-1,k}}{2h_x} - \frac{1}{(M_0)^2}\frac{\hat\rho^0_{m+1,k} - \hat\rho^0_{m-1,k}}{2h_x}$$

$$- \frac{\gamma}{\mathrm{RePr}}\left[\frac{\hat p^0_{m+1,k} - 2\hat p^0_{m,k} + \hat p^0_{m-1,k}}{h_x^2} - t_k \hat p^0_{m,k}\right.$$

$$\left. - \frac{1}{\gamma(M_0)^2}\left(\frac{\hat\rho^0_{m+1,k} - 2\hat\rho^0_{m,k} + \hat\rho^0_{m-1,k}}{h_x^2} - t_k \hat\rho^0_{m,k}\right)\right] = \hat f^0_{4_{m,k}}$$

$$m = 1, ..., M-1, \; k = -J, ..., J,$$

$$s_k = \frac{i \sin k\tilde h_y}{h_y}, \; t_k = \frac{4}{h_y^2}\sin^2\frac{k\tilde h_y}{2}, \; \tilde h_y = h_y \frac{2\pi}{Y}.$$

Introducing the additional variables

$$\begin{aligned} &\hat u^0_{m,k} - \hat u^0_{m-1,k} - h_x \chi_{m-1,k} = 0,\\ &\hat v^0_{m,k} - \hat v^0_{m-1,k} - h_x \eta_{m-1,k} = 0,\\ &\hat p^0_{m,k} - \hat p^0_{m-1,k} - h_x \zeta_{m-1,k} = 0,\\ &\hat\rho^0_{m,k} - \hat\rho^0_{m-1,k} - h_x \psi_{m-1,k} = 0,\\ &\qquad m = 1, ..., M, \end{aligned} \qquad (2.5)$$

and designating

$$\hat{\mathbf{v}}^0_{m,k} = (\chi, \eta, \zeta, \psi, \hat u^0, \hat v^0, \hat p^0, \hat\rho^0)^T\Big|_{m,k},$$

$$\hat{\mathbf{g}}^0_{m,k} = (\hat f^0_1, h_x \hat f^0_2, h_x \hat f^0_3, h_x \hat f^0_4, 0, 0, 0, 0)^T\Big|_{m,k}$$

we rewrite (2.4) as the system of eight first-order ordinary difference equations,

$$\begin{aligned} &\mathbf{A}_k \hat{\mathbf{v}}^0_{m,k} + \mathbf{B}_k \hat{\mathbf{v}}^0_{m-1,k} = \hat{\mathbf{g}}^0_{m,k},\\ &m = 1, ..., M, \; k = -J, ..., J, \end{aligned} \qquad (2.6)$$

where the coefficients of 8×8 matrices $\mathbf{A}_k$, $\mathbf{B}_k$ can be easily obtained from (2.4), (2.5). The explicit expressions for $\mathbf{A}_k$ and $\mathbf{B}_k$ are given in [19].

The principal point in the AP formulation consists in special boundary conditions at the lines $x = 0$ and $x = X$ (i.e., $m = 0$ and $m = M$). Namely, we impose these conditions separately for each wavenumber k as

$$\begin{aligned} &\mathbf{S}^-_{h,Y}(k)\hat{\mathbf{v}}^0_{0,k} = \mathbf{0}, \quad k = 0, \pm 1, \pm 2, ..., \pm J,\\ &\mathbf{S}^+_{h,Y}(k)\hat{\mathbf{v}}^0_{M,k} = \mathbf{0}, \quad k = 0, \pm 1, \pm 2, ..., \pm J, \end{aligned} \qquad (2.7)$$

where

$$\begin{aligned} \mathbf{S}^-_{h,Y}(k) &= \prod_{|\mu_s(k)|>1} (\mathbf{Q}_k - \mu_s(k)\mathbf{I}),\\ \mathbf{S}^+_{h,Y}(k) &= \prod_{|\mu_s(k)|\le 1} (\mathbf{Q}_k - \mu_s(k)\mathbf{I}). \end{aligned} \qquad (2.8)$$

$\mathbf{Q}_k = \mathbf{A}_k^{-1}\mathbf{B}_k$ in (2.8), $\mathbf{I}$ denotes the identity matrix, and $\mu_s(k)$, $s = 1, ..., 8$, are the eigenvalues of $\mathbf{Q}_k$ (to be found numerically in practice, e.g., using standard NAG routines).

Now we formally consider (2.6) on an infinite mesh $-\infty < m < \infty$. The system will be homogeneous for $m < 0$ and $m > M$ since $\hat{\mathbf{g}}^0_{m,k}$ has compact support. This homogeneous system has solutions increasing as $m \to +\infty$ as well as solutions increasing as $m \to -\infty$. The former correspond to eigenvalues $|\mu_s(k)| > 1$ and the latter to $|\mu_s(k)| < 1$. It is shown in [19] that *the conditions* (2.7) *prohibit, at* $m = 0$, *all the solutions which do not decrease as* $m \to -\infty$ *and at* $m = M$, *all the solutions which infinitely grow as* $m \to +\infty$. Thus, the solution to (2.6), (2.7), (2.8) coincides on $[0, M]$ with the corresponding fragment of the unique bounded solution to (2.6) if one considers this system for $-\infty < m < +\infty$. Then, implementing an inverse discrete Fourier transform we get a solution to the difference AP. The latter is a truncation to the strip $0 \le x \le X$ of a bounded on R^2 and a periodic in y-direction solution to the inhomogeneous discrete version of (2.1) for the same (see (2.4)) compactly supported right-hand side.

We have to specify what kind of convergence we assume while considering this finite-difference AP. An auxiliary problem can be constructed for the continuous case. We consider the strip $D_0 = \{(x, y) | 0 \le x \le X\}$ (see Fig. 1) as an auxiliary domain, implement the Fourier transform along the y-axis for (2.1), proceed to the first-order system of ODEs, and impose boundary conditions fully analogous to (2.7), (2.8) for each wavenumber $k \in R$. This procedure is described in [19]. The

only appreciable difference between the continuous and discrete cases is that the natural way of reducing (2.1) to the first-order system of ODEs in Fourier space actually yields the system of *seven equations*. In the finite-difference case we obtain the system of *eight equations* which is mainly for reasons of convenience. The difference between these two approaches is discussed in [19]. We require that the solution to the continuous AP be bounded on D_0, and absolutely integrable and representable as a Fourier integral along y for any x which in particular implies vanishing at any line $x = \text{const}$, $\mathbf{u}^0(x, y) \to 0$ as $y \to \pm\infty$. Boundary conditions of type (2.7) guarantee that this solution can be continued from D_0 to R^2 in such a way that it will vanish as $x \to \pm\infty$ uniformly with respect to $y \in (-\infty, +\infty)$ [19]. Hereafter we will use the above properties (which are slightly weaker) instead of (2.2).

We may expect that the solution of the difference AP converges uniformly to the solution of the continuous AP on any fixed finite subset $(0, X) \times (-\check{y}, \check{y}) \subset D_Y^0$ *while the grid size vanishes and the period* Y *grows.* This is a somewhat nonstandard definition of convergence. Namely, we consider the decreasing grid size h and simultaneously increasing period Y. Moreover, we no longer consider convergence on the whole domain D_Y^0 but only on any fixed subset $(0, X) \times (-\check{y}, \check{y})$ containing D_{in}. The reason is that we approximate a continuous solution by the difference one and *at the same time* we approximate a nonperiodic function by the periodic one. The latter is constructed in a special way. Namely, it can be obtained by calculating the Fourier integral which represents the nonperiodic function by the approximate quadrature formula of rectangles [19]. Therefore, one cannot achieve a uniform approximation of an original nonperiodic function by this periodic function *on the whole period* Y but it is possible on any (fixed) smaller interval $(-\check{y}, \check{y})$. To provide better accuracy, Y must increase. On the other hand, it is sufficient for our purposes to consider convergence only on the subdomain $(0, X) \times (-\check{y}, \check{y})$, since the right-hand sides of the AP are concentrated in some neighborhood of Γ and the solution is of interest to us also only "not far" from Γ (see below and [19] for more details). For this type of convergence the residuals should be simultaneously estimated in terms of size h and period Y. Some estimates connecting grid size and period with the desired accuracy are presented in [19]. A detailed discussion on this definition of convergence also appears there.

We will next implement an apparatus of the DPM to obtain the nonlocal ABCs themselves. We formulate here the main ideas, referring to [19, 20] for further details.

The space of the difference AP solutions $U_{h,Y}^0$ is determined on the grid $\mathcal{N}^0 = \{(x_m, y_j) \equiv (mh_x, jh_y - Y/2) | m = 0, \ldots, M, j = 0, \ldots, 2J + 1\}$ and the space of its right-hand sides $F_{h,Y}^0$, on the grid $\mathcal{M}^0 \stackrel{\text{def}}{=} \mathcal{N}^0 \backslash \{(x_m, y_j) | m = 0, M, j = 0, \ldots, 2J + 1\}$. $F_{h,Y}^0$ contains all the grid functions determined on $\mathcal{M}^0$. Designate $D_Y = D_Y^0 \cap D_{\text{ex}}$ and define the grid sets: $\mathcal{M} = \{(x_m, y_j) | (x_m, y_j) \in \mathcal{M}^0 \cap \overline{D}_Y\}$, $\mathcal{M}_{\text{in}} = \{(x_m, y_j) | (x_m, y_j) \in \mathcal{M}^0 \cap D_{\text{in}}\}$. An operator $\mathbf{L}_h^0$ of the difference AP is constructed using a 3×3 stencil, see above (and [19] for details). Denote such a stencil with the center (x_m, y_j) as $St_{m,j}$, then $\mathcal{N} \stackrel{\text{def}}{=} \bigcup_{(x_m, y_j) \in \mathcal{M}} St_{m,j}$; $\mathcal{N}_{\text{in}} \stackrel{\text{def}}{=} \bigcup_{(x_m, y_j) \in \mathcal{M}_{\text{in}}} St_{m,j}$; $\gamma \stackrel{\text{def}}{=} \mathcal{N} \cap \mathcal{N}_{\text{in}}$. We call the set γ the *grid boundary* (by analogy to the continuous boundary Γ). Evidently, γ consists of those nodes of the grid $\mathcal{N}^0$ located "not far from" Γ. Then introduce the operators: $\Theta_{\mathcal{M}}$—restriction of the grid function domain from $\mathcal{M}^0$ to $\mathcal{M}$, $\Theta_{\mathcal{M}}^0$—extension of the grid function domain from $\mathcal{M}$ to $\mathcal{M}^0$ in such a way that the new function would be zero on $\mathcal{M}^0 \backslash \mathcal{M} = \mathcal{M}_{\text{in}}$ and would coincide with the original one on $\mathcal{M}$, as well as the operators $\Theta_{\mathcal{N}}$ and $\Theta_{\mathcal{N}}^0$ acting analogously for the sets $\mathcal{N}$ and $\mathcal{N}^0$, respectively, and define the spaces of grid functions: $U_{h,Y} = \{\Theta_{\mathcal{N}} \mathbf{u}_{h,Y}^0 | \mathbf{u}_{h,Y}^0 \in U_{h,Y}^0\}$, $F_{h,Y} = \{\Theta_{\mathcal{M}} \mathbf{f}_{h,Y}^0 | \mathbf{f}_{h,Y}^0 \in F_{h,Y}^0\}$. The difference operator $\mathbf{L}_h : U_{h,Y} \to F_{h,Y}$ acts according to the same rule as $\mathbf{L}_h^0$. Moreover, introduce the operator $\mathbf{G}_{h,Y} : F_{h,Y} \to U_{h,Y}$ as follows: $\forall \mathbf{f}_{h,Y} \in F_{h,Y}$, $\mathbf{G}_{h,Y} \mathbf{f}_{h,Y} \stackrel{\text{def}}{=} \Theta_{\mathcal{N}} \mathbf{G}_{h,Y}^0 \Theta_{\mathcal{M}}^0 \mathbf{f}_{h,Y}$, where $\mathbf{G}_{h,Y}^0$ is the Green operator of the difference AP. The space of *difference-clear-traces* Ξ_h [20, p. 122] consists of all the four-component vector-functions ξ_h determined at the nodes of grid boundary γ, where the *difference-clear-trace* operator [20, p. 122] $\mathbf{Tr}_h : U_{h,Y} \to \Xi_h$ simply restricts the domain of the corresponding function from $\mathcal{N}$ to γ. The difference potential [20, p. 124] (see also [19]) with density from the space of clear traces Ξ_h is defined by

$$\mathbf{P}_{\mathcal{N}} \xi_h = \mathbf{u}_{h,Y} - \mathbf{G}_{h,Y} L_h \mathbf{u}_{h,Y}, \quad \mathbf{P}_{\mathcal{N}} : \Xi_h \to U_{h,Y}, \tag{2.9}$$

where $\mathbf{u}_{h,Y} \in U_{h,Y}$ in (2.9) is such that $\mathbf{Tr}_h \mathbf{u}_{h,Y} = \xi_h$ and is arbitrary in the rest. It is shown in [20, p. 125] that the potential $\mathbf{P}_{\mathcal{N}} \xi_h$ depends only on ξ_h and not on the choice of $\mathbf{u}_{h,Y}$ in the formula (2.9). Therefore we can choose $\mathbf{u}_{h,Y} \in U_{h,Y}$, $\mathbf{Tr}_h \mathbf{u}_{h,Y} = \xi_h$, in (2.9) in such a way that it is zero everywhere, except for some neighborhood of the set γ, or even simply everywhere except for γ. Since the operator $\mathbf{L}_h$ acts according to local formulae the function $\mathbf{L}_h \mathbf{u}_{h,Y} \in F_{h,Y}$ (which is the right-hand side for AP) will also differ from zero only in some small neighborhood of γ. Therefore, we can really consider the difference AP only for the compactly supported right-hand sides. *Evidently, the potential* $\mathbf{P}_{\mathcal{N}}$ *satisfies boundary conditions* (2.7) *of the difference AP and it is a solution to the homogeneous equation* $\mathbf{L}_h \mathbf{u}_{h,Y} = \mathbf{0}$.

Further, introduce the difference boundary projection [20, p. 125]: $\mathbf{P}_\gamma : \Xi_h \to \Xi_h$, $\mathbf{P}_\gamma \stackrel{\text{def}}{=} \mathbf{Tr}_h \mathbf{P}_{\mathcal{N}}$. The following *proposition holds for* $\mathbf{P}_\gamma$ (see [20, p. 126]): the equality

$$\xi_h - \mathbf{P}_\gamma \xi_h = \mathbf{0} \tag{2.10}$$

is valid *for those and only those* $\xi_h \in \Xi_h$ which are the trace $\xi_h = \mathbf{Tr}_h \mathbf{u}_{h,Y}$ of some solution $\mathbf{u}_{h,Y} \in U_{h,Y}$ of the homogeneous equation $\mathbf{L}_h \mathbf{u}_{h,Y} = \mathbf{0}$. When (2.10) is valid the solution $\mathbf{u}_{h,Y} \in$

$U_{h,\gamma}$ of the equation $\mathbf{L}_h\mathbf{u}_{h,\gamma} = \mathbf{0}$ with trace ξ_h, $\mathbf{Tr}_h\mathbf{u}_{h,\gamma} = \xi_h$, is unique and can be computed by means of the generalized difference Green formula [20, p. 125]:

$$\mathbf{u}_{h,\gamma} = \mathbf{P}_{\mathcal{N}}\xi_h. \tag{2.11}$$

To construct the ABCs we need to complete the system of difference equations (on a C-grid) in D_{in}. Therefore, we have to relate the values of the solution at the inner (penultimate) and outermost rows of the grid nodes. Assume that the inner row corresponds to the curve Γ and contains the nodes ν, and the outermost row consists of the nodes ν_1 belonging to the curve Γ_1 (see Fig. 1). We know all the flow parameters $\mathbf{u}_\nu$ at ν and we have to prescribe them at ν_1 using the representation of the solution outside Γ in the form of the potential (2.9). Indeed, this potential is a solution of the discrete counterpart to (2.1) (see above) in D_{ex}. Moreover, because of conditions (2.3), (2.7) the far-field behavior of the potential (2.9) tends to the far-field behavior of the solution to the continuous AP (see definition above) while the grid size vanishes and the period simultaneously grows (provided that convergence takes place).

The following procedure is implemented to obtain the nonlocal ABCs relating $\mathbf{u}_\nu$ and $\mathbf{u}_{\nu_1}$. First, introduce the space Ξ of continuous clear traces. This space consists of the eight-component vector functions defined on Γ. The functions $\xi \in \Xi$ contain the unknowns (u, v, p, ρ) and their normal derivatives on the curve Γ. All the constructions of potentials and projections (see above) including the analogue of (2.10) can be developed in the continuous case using the space Ξ and the Green operator of the continuous AP [19]. Now introduce some finite-dimensional approximation to Ξ and designate it: $\Xi_\omega \ni \xi_\omega$. Since ξ are the vector-functions, the finite-dimensional approximation is implemented componentwise. The dimensionality of Ξ_ω is $8|\omega|$, where $|\omega|$ corresponds to each component. In practice ξ_ω are the eight-component vector functions defined on the discrete finite set of points $\omega \subset \Gamma$. We use spline interpolation (local splines [20, p. 303]) $\mathbf{R}: \Xi_\omega \to \Xi$ to get a continuous function $\mathbf{R}\xi_\omega$ approximating ξ and we certainly demand the following property: $\forall\varepsilon > 0\ \exists\omega$ ($|\omega|$ is sufficiently large) such that $\forall\xi \in \Xi\ \exists\xi_\omega \in \Xi_\omega$ for which $\|\xi - \mathbf{R}\xi_\omega\|_\Gamma < \varepsilon$, where $\|\cdot\|_\Gamma$ is the norm chosen in an appropriate way [20, pp. 162–167].

The functions ξ_h contain (u, v, p, ρ) at the nodes γ located near Γ. One can therefore easily compute ξ_h from ξ using Taylor expansion. We designate the composition of spline interpolation $\mathbf{R}$ and the Taylor formula as $\pi: \Xi_\omega \to \Xi$. π is the operator of boundary data continuation from the boundary to the domain (i.e., from ω to γ).

As was already mentioned we consider all the parameters $(u, v, p, \rho)^T|_\nu \equiv \mathbf{u}_\nu \in U_\nu$ at the nodes ν as known values. Now let $\xi_\omega = (\xi_\omega^{(1)}, \xi_\omega^{(2)})^T$, where $\xi_\omega^{(1)} = (u, v, p, \rho)|_\omega$ and $\xi_\omega^{(2)} = (\partial u/\partial n, \partial v/\partial n, \partial p/\partial n, \partial\rho/\partial n)|_\omega$. Introduce one more interpolation operator $\mathbf{R}_\nu$: $U_\nu \to \Xi_\omega$, $\mathbf{R}_\nu\mathbf{u}_\nu = \xi_\omega^{(1)}$. Further, apply the operator π: $\xi_h = \pi\xi_\omega \overset{\text{def}}{=} \pi^{(1)}\xi_\omega^{(1)} + \pi^{(2)}\xi_\omega^{(2)} = \pi^{(1)}\mathbf{R}_\nu\mathbf{u}_\nu + \pi^{(2)}\xi_\omega^{(2)}$ and substitute this expression into (2.10),

$$\mathbf{Q}^{(1)}\mathbf{u}_\nu + \mathbf{Q}^{(2)}\xi_\omega^{(2)} = 0, \tag{2.12}$$

where $\mathbf{Q}_\gamma \overset{\text{def}}{=} \mathbf{I} - \mathbf{P}_\gamma$, $\mathbf{I}$ is the identity operator, $\mathbf{Q}^{(1)} = \mathbf{Q}_\gamma\pi^{(1)}\mathbf{R}_\nu$, $\mathbf{Q}^{(2)} = \mathbf{Q}_\gamma\pi^{(2)}$. Equation (2.12) with respect to $\xi_\omega^{(2)}$ is, generally speaking, overdetermined and has no solution. We define its generalized solution in the sense of the least squares method introducing some Euclidean norm $\|\cdot\|_\gamma$ in Ξ_h for this purpose. This norm is chosen as a discrete analogue to the Sobolev W_2^1 norm (see [19] for the specific formulae) and can be written in the form $\|\xi_h\|_\gamma^2 = (\mathbf{A}_\gamma\xi_h, \xi_h)$, where $\mathbf{A}_\gamma$: $\Xi_h \to \Xi_h$ is some symmetric linear operator and the scalar product in Ξ_h is defined in the usual way, $(a, b) = \sum_{l=1}^{4}\sum_{j\in\gamma} a_j^l b_j^l$, $a, b \in \Xi_h$, where l enumerates components of vector functions a and b.

Define the generalized solution of (2.12) as the solution of the variational problem:

$$\|\mathbf{Q}^{(1)}\mathbf{u}_\nu + \mathbf{Q}^{(2)}\xi_\omega^{(2)}\|_\gamma^2 \to \min. \tag{2.13}$$

A necessary condition for the minimum of (2.13) results in the linear system

$$\mathbf{Q}^{(2)^T}\mathbf{A}_\gamma\mathbf{Q}^{(2)}\xi_\omega^{(2)} = -\mathbf{Q}^{(2)^T}\mathbf{A}_\gamma\mathbf{Q}^{(1)}\mathbf{u}_\nu \tag{2.14}$$

to be solved with respect to $\xi_\omega^{(2)}$, i.e., with respect to the normal derivatives of solution $\mathbf{u}$ at Γ.

One can find the solution of (2.14) directly by means of

$$\xi_\omega^{(2)} = \mathbf{K}_\omega^{-1}\mathbf{K}_\nu\mathbf{u}_\nu, \tag{2.15}$$

$$\mathbf{K}_\omega \overset{\text{def}}{=} \mathbf{Q}^{(2)^T}\mathbf{A}_\gamma\mathbf{Q}^{(2)}, \quad \mathbf{K}_\nu \overset{\text{def}}{=} -\mathbf{Q}^{(2)^T}\mathbf{A}_\gamma\mathbf{Q}^{(1)}$$

and then obtain

$$\xi_h = \pi^{(1)}\mathbf{R}_\nu\mathbf{u}_\nu + \pi^{(2)}\mathbf{K}_\omega^{-1}\mathbf{K}_\nu\mathbf{u}_\nu. \tag{2.16}$$

The operator $\mathbf{K}_\omega^{-1}\mathbf{K}_\nu$ from (2.15) expresses the normal derivatives $\xi_\omega^{(2)}$ in terms of the functions $\mathbf{u}_\nu$. The relation (2.15) is a consequence of (2.10), i.e., of its variational version (2.13). Therefore, ξ_h from (2.16) satisfies (2.10) (in the variational sense (2.13)). This implies that we continue the boundary data $\mathbf{u}_\nu$ to the domain in such a way that this continuation ξ_h is a clear trace of some solution to the difference version of (2.1) (see text above Eq. (2.3)) in D_γ^0. This solution satisfies the boundary conditions (2.3), (2.7) and can be restored by means of (2.11). We use it to find $\mathbf{u}_{\nu_1}$; namely, we find the values $\mathbf{u}_{\nu_1}$ with the help of interpolation from the grid $\mathcal{N}$. Designate as κ that subset of $\mathcal{N}$ nodes where it is necessary to know the solution $\mathbf{u}_{h,\gamma} = \mathbf{P}_{\mathcal{N}}\xi_h$ in order to implement an interpolation procedure of sufficiently high order, e.g., interpolation by quadratic polynomials. Evidently, κ is the grid set located near ν_1. Instead of (2.11), write

$$\mathbf{u}_{\nu_1} = \mathbf{P}_{\nu_1}\xi_h, \tag{2.17}$$

where $\mathbf{P}_{\nu_1} \stackrel{\text{def}}{=} \mathbf{R}_{\nu_1\kappa}\mathbf{P}_{\kappa}$, $\mathbf{R}_{\nu_1\kappa}$ is an interpolation operator from κ to ν_1. Substituting (2.16) into (2.17) we get

$$\mathbf{u}_{\nu_1} = \mathbf{P}_{\nu_1}(\pi^{(1)}\mathbf{R}_{\nu} + \pi^{(2)}\mathbf{K}_{\omega}^{-1}\mathbf{K}_{\nu})\mathbf{u}_{\nu} \equiv \mathbf{T}\mathbf{u}_{\nu}. \tag{2.18}$$

The matrix relation (2.18) is *the desired nonlocal* ABC. Indeed, (2.18) expresses $\mathbf{u}_{\nu_1}$ in terms of $\mathbf{u}_{\nu}$ through the solution of the linearized Navier–Stokes equations outside Γ. The far field behavior of this solution is determined by the boundary conditions of the AP. Therefore, (2.18) completes the system of difference equations in D_{in} in the correct way. While conducting practical computations (see Section 3) we were using an iteration procedure for solving the discretized Navier–Stokes equations inside D_{in}. The ABC (2.18) was implemented at each iteration to complement the values of the solution at the outermost coordinate line. Note that the implementation of the ABC (2.18) requires the explicit computation of matrix $\mathbf{T}$ (as an operator with respect to the specific basis). This necessitates the repeated solution ($8|\omega|$ times) of the difference AP. The computer time expenditure for such a computation is not high; its theoretical estimates are contained in [19] and practical results are described below.

3. NUMERICAL RESULTS

In this section, we describe the results of numerical implementation of the nonlocal ABC (2.18). We compute the steady state of subsonic compressible viscous flow past the airfoil NACA0012. The computations will be carried out for various values of the parameters M_0, Re, and angle of attack α.

First, we introduce a curvilinear boundary-fitted grid of C-type, which determines the shape of the computational domain D_{in}. We use a hyperbolic generator to construct the grid around the airfoil. In all the computations described below the basic grid has 256 nodes "along the airfoil surface" and 64 nodes in the direction "normal to the airfoil surface." To carry out the computations for domains D_{in} of different sizes (see below), we vary the "average radius of the computational domain." It is performed either by varying the "normal" stretching of the grid or by means of truncating some external part of it.

We use a finite-volume code [16, 21, 22] to integrate the Navier–Stokes equations on a C-type grid inside D_{in}. This code is based on the second-order approximation to the spatial operator and a five-stage Runge–Kutta integration in time. The spatial approximation is constructed using a 3×3 stencil. The code [16, 21, 22] involves a multigrid pseudo-time iteration procedure for achieving a steady state. While computing we use four levels of multigrid with regular time steps and W-cycles.

The following is an original treatment of the external boundary in the code [16, 21, 22]. Boundary conditions at inflow (see Fig. 1) are based on the usual locally one-dimensional characteristic analysis, and boundary conditions at outflow (see Fig. 1) are simply the extrapolation of all the variables u, v, p, ρ. The advantages of such an approach are its algorithmic simplicity, very low CPU time expenditure, and applicability to domains of irregular shape, in particular, to any specific domain D_{in} determined by the C-grid. It turns out, however, that the accuracy and convergence rate provided by these local conditions may be improved substantially, as will be seen from further consideration.

To implement the ABC (2.18) we have to define the sets ν and ν_1. It turns out that according to the structure of the scheme stencil (3×3) [16, 21, 22] one may consider the outermost row of nodes of the C-type grid as ν_1 and the penultimate row of nodes as ν. Boundary condition (2.18) is applied only on the finest level of multigrid (to complete the system of difference equations inside D_{in} on the finest grid); on the coarser levels we simply retain the boundary values provided from the finest one.

To compute the operator $\mathbf{T}$ from (2.18) we have to specify the period Y (see Eq. (2.3) and the text above it). The units for measuring the value of Y will be "diameters" of the computational domain rather than the airfoil chords. It turns out that this way of measuring Y is more convenient from a practical viewpoint. For each specific flow regime as well as for each specific grid (C-type) we will try to use a few different operators $\mathbf{T}$ corresponding to different values of Y. We will investigate what influence the change of Y exerts on a final solution and on the properties of numerical algorithm.

We consider two well-known flow regimes past a NACA0012 airfoil: $M_0 = 0.63$, $\alpha = 2°$ which is a purely subsonic (subcritical) flow, and $M_0 = 0.85$, $\alpha = 1°$ which is a transonic (supercritical) flow with finite supersonic regions located near the airfoil surface. In the inviscid case both these regimes were studied numerically by many authors. We will investigate the viscous case for several low values of Reynolds number.

3.1. *Subcritical Regime*

Our first computational example is the subcritical flow $M_0 = 0.63$, $\alpha = 2°$ past a NACA0012 airfoil for low Reynolds number $\text{Re} = 400$. We use a basic C-type grid of 256×64 nodes with an "average radius of computational domain" of about 15 chords of airfoil. Figure 2 represents convergence dynamics (i.e., dependence of the ρ-residual in the L_∞-norm on the number of iterations) for this computation.

We compare the convergence rate of the multigrid iteration procedure for standard (extrapolation) conditions and for the nonlocal ABC (2.18) when the period Y is equal to six "diameters." One can easily observe that the number of iterations required to reduce an initial residual by a prescribed factor is more than 3 *times* smaller for the nonlocal ABCs than for extrapolation. This means that the "theoretical convergence rate" is actually more than 3 times faster for nonlocal conditions

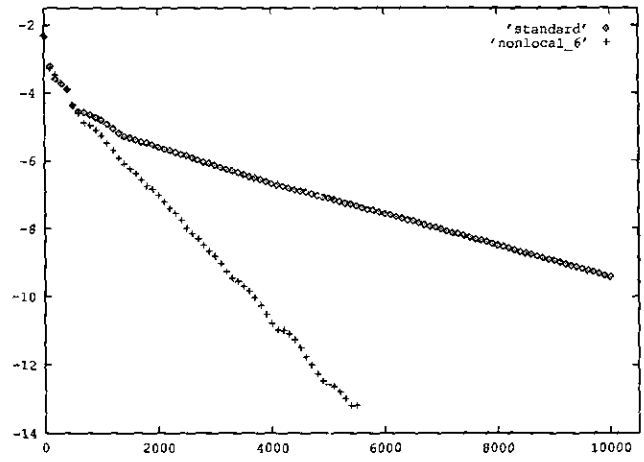

FIG. 2. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 400, grid 256 × 64, "average radius" ≈15 chords.

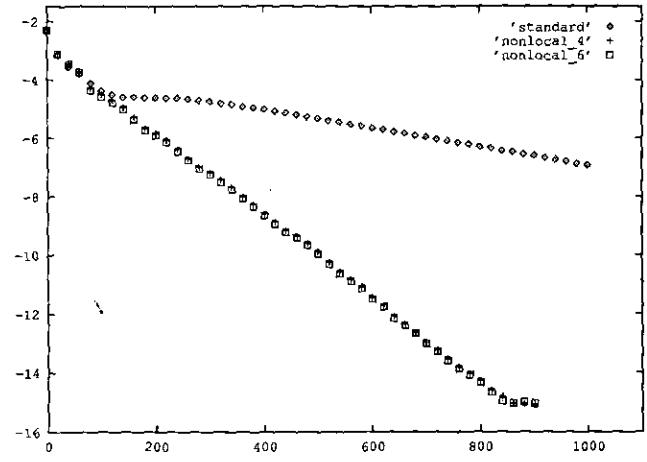

FIG. 4. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 400, grid 192 × 32, "average radius" ≈2 chords.

than for standard ones. Of course, in order to understand what the actual integral gain of CPU time is we have to take into account the additional expenditure due to nonlocal ABCs. This additional expenditure has two components: the cost of the boundary operator **T** and the application itself of the nonlocal condition (2.18) (i.e., matrix–vector multiplication on each iteration). We shall later evaluate precisely both these components, but first let us consider the results of the same flow computations obtained while using smaller grids and, consequently, smaller computational domains. Namely, we consider three C-type (sub)grids of 224 × 48, 192 × 32, and 176 × 24 nodes, obtained by truncating certain external parts of the basic 256 × 64 grid which was used for a computation corresponding to Fig. 2. The "average radii of computational domains" for these grids are approximately 5.5, 2, and 1.2 chords of airfoil, respectively. Figures 3, 4, and 5 represent convergence dynamics for these computations. Again we compare nonlocal and standard (extrapolation) external boundary conditions.

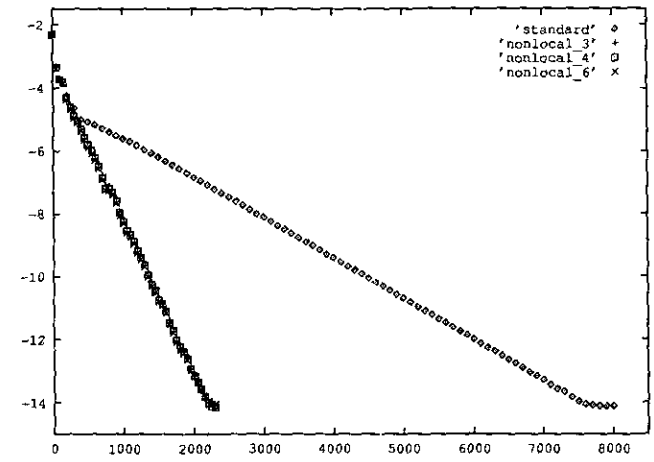

FIG. 3. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 400, grid 224 × 48, "average radius" ≈5.5 chords.

In the case of the 224 × 48 grid (Fig. 3) we use three different operators **T** for $Y = 3$, 4, and 6 "diameters"; the corresponding curves are designated *nonlocal_3*, *nonlocal_4*, and *nonlocal_6* in Fig. 3. For the 192 × 32 (Fig. 4) grid we use two different operators **T** corresponding to $Y = 4$ and $Y = 6$. One can easily see that in all the cases the convergence rate provided by nonlocal ABCs is more than 3 times faster than for the standard (extrapolation) conditions. Note, that in so doing both procedures (multigrid with standard and multigrid with nonlocal ABCs) are found to converge faster for smaller domains (see Figs. 2, 3, 4, 5) but the relative difference in convergence rates of these procedures for each specific variant remains approximately the same: about 3 or slightly more. We also note that at least for this specific regime the value of Y has no substantial influence on the convergence rate. Of course, one cannot choose values of Y too small. For example, we have verified that if $Y = 1$ (i.e., one "diameter") then convergence fails; but starting from the values $Y \approx 2$–3 (see Fig. 3 and also below) the convergence rate remains the same for all larger values of Y. This circumstance is very useful from a practical viewpoint since the smaller Y is, the lower the cost of the corresponding operator **T** [19].

Next we investigate the accuracy of these computations, i.e., how the resulting solution depends on the type of external boundary condition and on the size of computational domain. To do this we will analyze the behavior of dynamic force coefficients, namely, lift coefficient C_l and drag coefficient C_d. These coefficients are of particular interest for applications, e.g., for aircraft design. C_l and C_d are calculated in the code [16, 21, 22] by integrating the dynamic part of the momentum flux along the airfoil surface using the trapezoid rule. Since the integration contour is located far from the external boundary, the calculated values of C_l and C_d are not affected explicitly

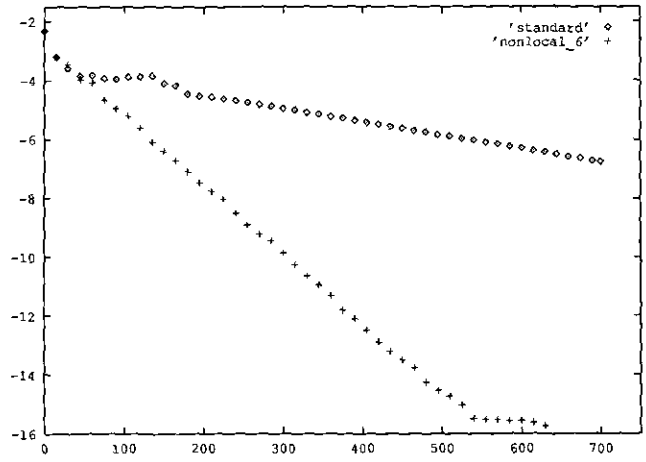

FIG. 5. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 400, grid 176 × 24, "average radius" ≈1.2 chords.

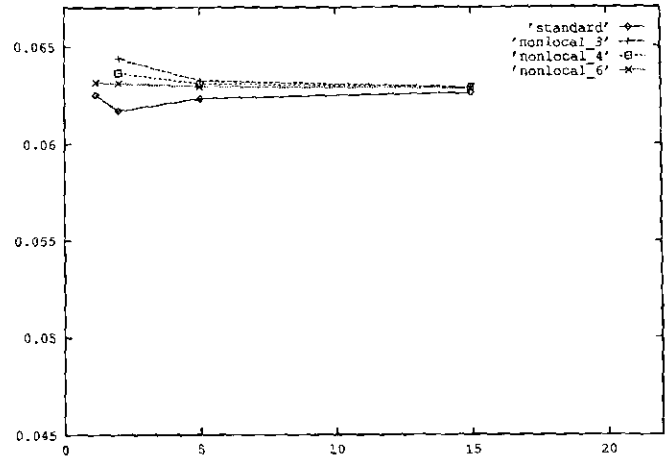

FIG. 7. Dynamic drag coefficient versus size of computational domain; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 400; basic grid, 256 × 64 with "average radius" 15 chords, is the largest domain.

by the type of ABC, but only through the change of solution in the whole computational domain caused by the change of external boundary condition. Figures 6 and 7 represent the behavior of lift and drag coefficients, respectively, for different types of artificial boundary conditions and for different sizes of computational domain.

The data shown in Figs. 6 and 7 correspond to the computations ($M_0 = 0.63$, Re = 400, $\alpha = 2°$) described above (see Figs. 2, 3, 4, 5). We want to emphasize here two facts. The first one is very natural: the larger the computational domain, the less the influence exerted by external conditions on a solution. Indeed, one can see from Figs. 6 and 7 that for an "average radius" of 15 chords the discrepancy between the force coefficients corresponding to the different types of ABCs is small (about 1%). On the other hand, for smaller computational domains this discrepancy increases. It is easy to observe (especially in Fig. 6) that the force coefficients corresponding to nonlocal ABCs vary rather slightly when the computational domain is shrunk, whereas the coefficients provided by standard conditions vary more noticeably.

This is in fact the second important conclusion: nonlocal ABCs produce a solution near the airfoil that is more stable with respect to changes in the computational domain size. This circumstance enables us to use smaller computational domains while using nonlocal ABCs which saves computer resources without loss of accuracy, or on the other hand, to improve accuracy by means of grid refinement keeping the CPU time expenditure at the same level.

Let us now consider the results of the same flow ($M_0 = 0.63$,

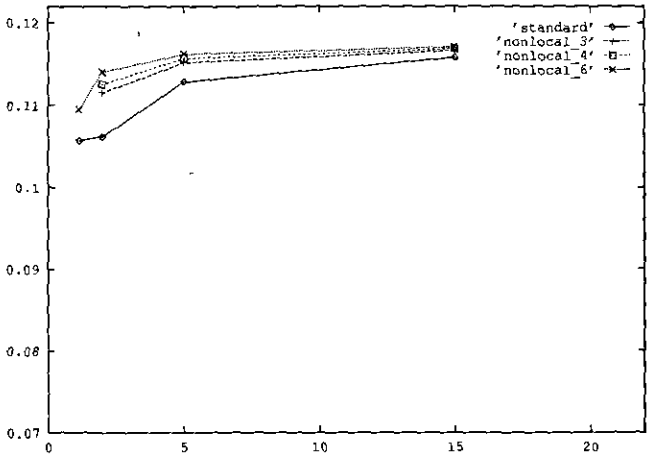

FIG. 6. Dynamic lift coefficient versus size of computational domain; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 400; basic grid, 256 × 64 with "average radius" 15 chords, is the largest domain.

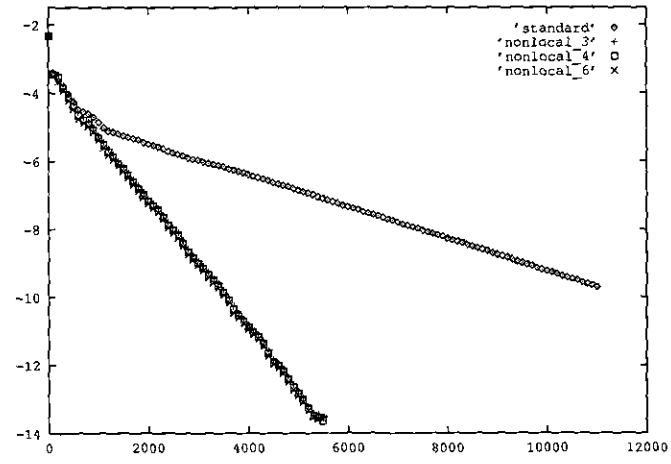

FIG. 8. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 400, grid 256 × 64, "average radius" ≈5.5 chords.

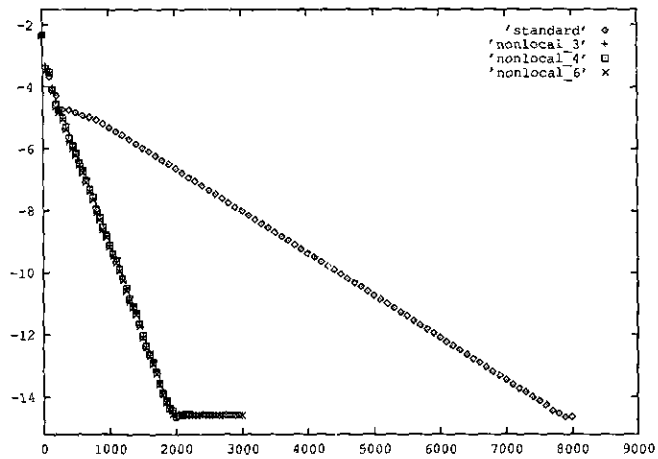

FIG. 9. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 400, grid 224 × 48, "average radius" ≈ 2 chords.

Re = 400, $\alpha = 2°$) computation but for a smaller original computational domain. Namely, we use the basic grid of 256 × 64 nodes with an "average radius" of about 5.5 chords of airfoil. This grid is finer than the previous one, especially near the airfoil surface. Figure 8 represents the convergence dynamics for this case.

Again, one observes that the convergence for nonlocal ABCs (for $Y = 3$, 4, and 6) is more than 3 times faster than for standard conditions. Truncating a certain external part of this basic grid we get a new one which consists of 224 × 48 nodes with an "average radius" of about two chords. The convergence dynamics for flow computations on this truncated grid is presented in Fig. 9.

One can see the same drastic difference in convergence rates for different types of ABCs. Moreover, it is worth mentioning that we observe the same situation as above: for smaller meshes both procedures (multigrid with nonlocal and multigrid with standard ABCs) converge faster, but the relative difference in convergence rates caused by different types of ABCs remains approximately the same.

Now compare the values of force coefficients obtained for different sizes of computational domain and for different types of ABCs on this fine grid. We present these values in Table I.

From Table I we see that standard boundary conditions cause a stronger dependence of force coefficients on the size of the computational domain, whereas only a slight dependence is created by the nonlocal ABC (2.18) with a large period, $Y = 6$ "diameters." Nonlocal ABCs corresponding to smaller periods ($Y = 3$, 4) possess intermediate properties. This behavior of calculated force coefficients seems natural since, the larger the period Y, the closer we get to the original solution (i.e., to the solution of an original unbounded problem linearized in the far field) [19]. We again emphasize here that the period Y is measured in "diameters" of computational domain. Therefore, the operators $\mathbf{T}_Y$, $Y = 3, 4, 6$, calculated for computational domains of different "average radii" (2 and 5.5 chords, respectively) are found to be calculated for different actual values of Y (when measured in airfoil chords). However, our numerical experiments show that convergence properties of an algorithm will be similar for computational domains of different sizes if the corresponding boundary operators $\mathbf{T}$ are calculated for the same values of Y expressed in the domain's "diameters." Indeed, if the period Y is equal to one "diameter" then there is no convergence at all for any size of computational domain. As Y is enlarged, convergence appears to start from Y of about two "diameters."

Note also, that the values of force coefficients obtained on the finer grid (see Table I) slightly differ from the corresponding values obtained on the coarser grid (see Figs. 6 and 7). To clarify this difference we present Table II which corresponds to the coarser grid and is analogous to Table I.

The data presented in Table II (see also Figs. 6, 7) are obtained using the grid which is three times coarser near the airfoil (in the normal direction) than the grid used for obtaining the data presented in Table I. This explains the discrepancy between the corresponding values. Of course, it is natural to assume that the computations on the finer grid (Table I) are more accurate. However, in both cases the dependence of the solution (force coefficients) on the type of external boundary conditions is the same—and that is one of the salient points of this investigation.

Let us now consider one more subcritical flow regime corresponding to a higher but still low Reynolds number: $M_0 = 0.63$, $\alpha = 2°$, Re = 4000. Our computations show that this flow turns out to be separated; a small separation zone is located on the upper surface of the airfoil near the trailing edge. Due to

TABLE I

Grid	224 × 48, 2 chords				256 × 64, 5.5 chords			
	Nonlocal				Nonlocal			
ABC	$Y = 3$	$Y = 4$	$Y = 6$	Standard	$Y = 3$	$Y = 4$	$Y = 6$	Standard
C_l	0.1079	0.1105	0.1115	0.1042	0.1136	0.1140	0.1144	0.1113
C_d	0.0620	0.0614	0.0609	0.0591	0.0611	0.0610	0.0609	0.0603

TABLE II

Grid	192 × 32, 2 chords				224 × 48, 5.5 chords			
	Nonlocal				Nonlocal			
ABC	$Y = 3$	$Y = 4$	$Y = 6$	Standard	$Y = 3$	$Y = 4$	$Y = 6$	Standard
C_l	0.1115	0.1125	0.1140	0.1061	0.1151	0.1157	0.1162	0.1158
C_d	0.0644	0.0636	0.0631	0.0617	0.0632	0.0631	0.0629	0.0623

the influence of viscosity and flow separation the corresponding force coefficients (see Table III below) differ strongly from the values relevant to the same flow regime being treated as inviscid (the latter have been calculated by numerous authors: $C_l \approx 0.3325$, $C_d \approx 0.0002$). To ensure that the flow separation is not caused by a violation of conservation laws which may occur for a difference scheme we have calculated the force coefficients by integrating the momentum flux along different closed contours enveloping the airfoil. We used contours located far enough from the airfoil as well as those intersecting the separation zone. For all the contours we obtained close values of coefficients (differing within the accuracy prescribed by the grid) which gives reasons to expect that the finite-difference scheme does not violate the conservation laws.

For this computation we use a 256 × 64 grid corresponding to the computational domain of an "average radius" of 5.5 chords. The convergence dynamics is presented in Fig. 10.

We again observe that the convergence provided by nonlocal ABCs is more than 3 times faster than the convergence provided by standard (extrapolation) conditions. However, it is worth mentioning that both iteration procedures (with standard and nonlocal ABCs) converge about 1.5 times slower for the case Re = 400 (Fig. 8) than for the case Re = 4000 (Fig. 10). This behavior is presumably determined by the code [16, 21, 22]. The convergence rate for the nonlocal ABCs is almost independent of the period Y chosen for calculating the operator $\mathbf{T}$ (see the curves *nonlocal_2*, *nonlocal_3*, and *nonlocal_4* in Fig. 10 for $Y = 2$, 3, and 4, respectively). The corresponding force coefficients are presented in Table III.

We see that these coefficients differ only slightly from each other which means that (at least for this specific case) we may restrict ourselves by using only cheap operators $\mathbf{T}$ (corresponding to small values of Y).

TABLE III

ABC	Standard	Nonlocal		
		$Y = 2$	$Y = 3$	$Y = 4$
C_l	0.02509	0.02455	0.02470	0.02470
C_d	0.03129	0.03147	0.03144	0.03139

We now return to the question of additional CPU time expenditure needed for computing the nonlocal ABCs. We begin with the last example: $M_0 = 0.63$, $\alpha = 2°$, Re = 4000. Due to the nonlocal nature of our boundary conditions their implementation results in matrix–vector multiplication on each iteration (see (2.18)) which makes each iteration about 10% more expensive. Specifically, one iteration of the code [16, 21, 22] on a 256 × 64 grid with four levels of multigrid and W-cycles costs about 14.9 s of CPU time on an IBM RISC 6000/540 workstation for standard external conditions and about 16.4 s for nonlocal ABCs. To estimate efficiency we have to decide what accuracy is sufficient for our purposes. Usually the solution corresponding to 10^{-8} (ρ-residual in L_∞-norm) is already quite satisfactory. In this case we need 4600 iterations with standard conditions (see Fig. 10) which implies about 19 h of CPU time, and only 1500 iterations with nonlocal ABCs (see Fig. 10) which means 6 h 49 min. While evaluating the total time relevant to the case of nonlocal ABCs we, of course, have to add the expenditure for calculating the boundary operator $\mathbf{T}$ itself. It turns out that on the same computer the operator $\mathbf{T}$ for $Y = 4$ costs about 120 min, for $Y = 3$—about 80 min, and for $Y = 2$—about 52 min. We see that our integral gain for the case $Y = 2$, i.e., that part of total CPU time which we can

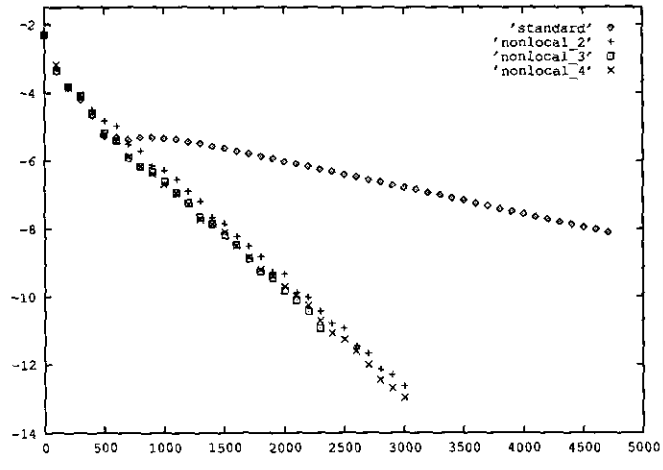

FIG. 10. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.63$, $\alpha = 2°$, Re = 4000, grid 256 × 64, "average radius" ≈5.5 chords.

save using nonlocal ABCs (in comparison with standard ones) is found to be about 60%. Indeed, the total CPU time using nonlocal ABCs is 7.7 h (for $Y = 2$) compared to 19 h for the standard boundary conditions. This is a very significant gain.

Obviously, the cost of one iteration as well as the cost of boundary operator $\mathbf{T}$ does not depend on flow parameters M_0 and Re. These costs actually depend only on geometric factors: the number of nodes, the shape of artificial boundary, the value of Y. Therefore, the estimates of CPU time for other computed variants will be approximately the same. For example, from Fig. 8 (low Reynolds number Re = 400) we see that to achieve 10^{-8} accuracy one needs 7500 standard iterations and 2500 iterations with nonlocal ABCs which implies 31 h and 11 h 23 min, respectively. The last value should be increased by a cost of $\mathbf{T}$ which is not greater than 210 min—for $Y = 6$. The integral gain of CPU time is again around 60%. Another example refers to smaller grids. From Fig. 9 (224×48 grid for low Re = 400) we conclude that one needs 800 iterations with nonlocal ABCs and 3000 standard iterations to achieve 10^{-8} accuracy which is 2 h 25 min and 7 h 54 min, respectively (one iteration costs 10.9 s for nonlocal conditions and 9.5 s for standard ones on this grid). Adding the cost of $\mathbf{T}$ for $Y = 3$ (80 min) to the "nonlocal" time (2 h 25 min) we find that we save about 53% of the total time for this specific case. Note, that here we do not need to use more expensive operators $\mathbf{T}$. Generally, it is more advantageous to use the nonlocal ABC (2.18) for larger grids (i.e., for grids with a larger total number of nodes) although even for smaller ones we can save more than half of the original CPU time.

3.2. *Supercritical Regime*

So far we have made less effort to investigate the flow: $M_0 = 0.85$, $\alpha = 1°$. The numerical study of such a regime for the case of low laminar Reynolds number (Re = 4000) presents essential difficulties because of the strong flow separation and the existence of local supersonic zones. We will describe the results obtained while computing this supercritical flow on two different grids of 256×64 nodes with "average radii" of 11 and 5.5 chords, respectively.

The convergence dynamics for the computational domain of an "average radius" of 11 chords is shown in Fig. 11. Again, nonlocal ABCs provide faster convergence. Taking into account all the specific computational costs, it turns out that one can save about half of the total CPU time here.

Our numerical experiments confirm that this supercritical case is more difficult from the point of view of convergence, and that numerical algorithms become more sensitive to the method of treating the external boundary. In practice this sensitivity necessitates choosing larger periods Y for nonlocal ABCs and consequently more expensive operators $\mathbf{T}$. For standard conditions it turns out that for small computational domains the convergence may simply fail. This is clearly seen from Fig. 12, where we present convergence dynamics for the computational domain of "average radius" of about 5.5 chords. Here the iteration procedure with the nonlocal ABC (2.18) still converges rather rapidly, whereas standard conditions provide no convergence at all. This implies that the nonlocal ABCs (2.18) may exert a certain stabilizing and regularizing influence on a numerical algorithm.

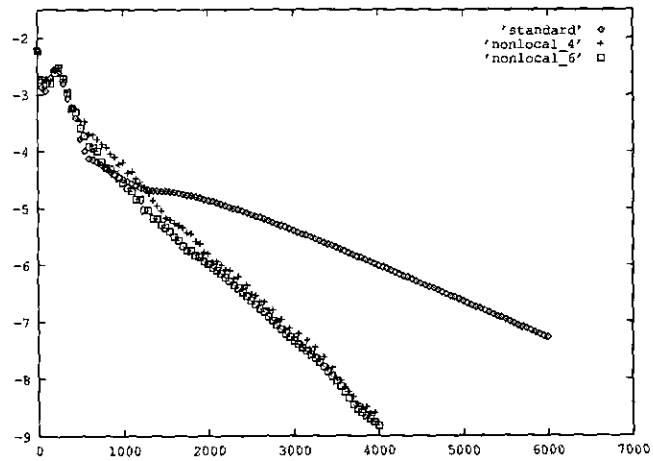

FIG. 11. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.85$, $\alpha = 1°$, Re = 4000, grid 256×64, "average radius" $\approx$11 chords.

Now consider some flow patterns corresponding to these computations. In Fig. 13 we present level lines of u-velocity in a certain (close-to-the-airfoil) subregion of an original domain. One can easily observe a large separation zone. This zone develops more strongly near the upper surface than near the lower one since the flow is not symmetric: $\alpha = 1°$. Due to a rather thick boundary layer (Re = 4000) and, in particular, its separation one may believe that the "effective shape" of an immersed body has now changed which in turn strongly influences the shape and

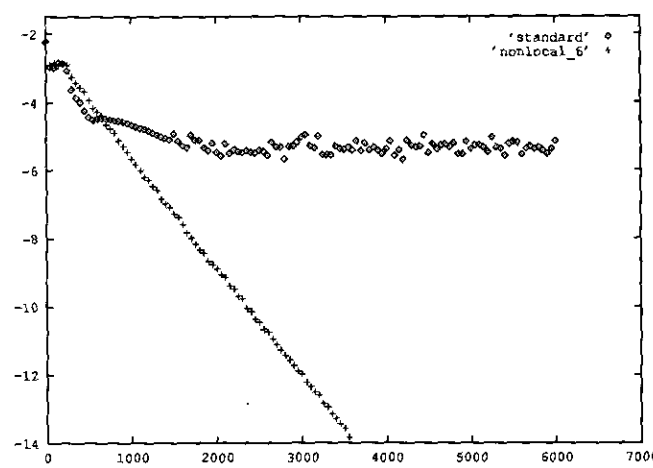

FIG. 12. Logarithm of ρ-residual in L_∞-norm versus number of iterations; NACA0012, $M_0 = 0.85$, $\alpha = 1°$, Re = 4000, grid 256×64, "average radius" $\approx$5.5 chords.

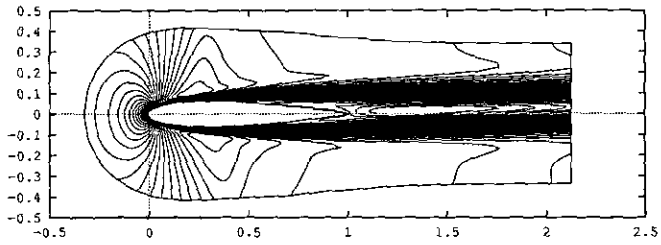

FIG. 13. u-velocity contours (fragment), $u_{\min} = -0.1314$, $u_{\max} = 1.238$, $\Delta u = 0.0351$; NACA0012, $M_0 = 0.85$, $\alpha = 1°$, Re = 4000, grid 256 × 64, "average radius" ≈5.5 chords.

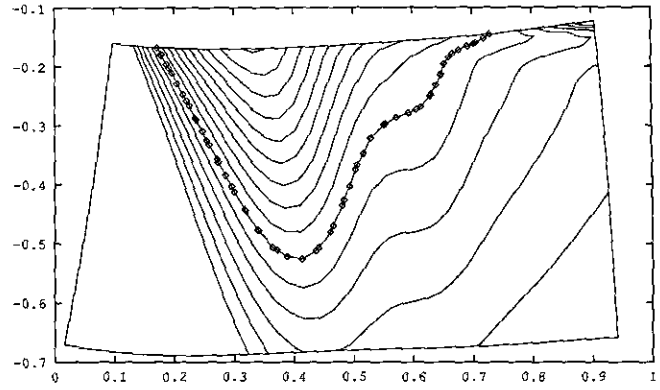

FIG. 15. A fragment of Mach contours near the lower surface of airfoil, $M_{\min} = 0.95$, $M_{\max} = 1.09$, $\Delta M = 0.01$, ◇ corresponds to the contour $M = 1.00$; NACA0012, $M_0 = 0.85$, $\alpha = 1°$, Re = 4000, grid 256 × 64, "average radius" ≈5.5 chords.

structure of local supersonic regions relevant to this supercritical flow. These regions are located near both (upper and lower) surfaces of the airfoil; their configuration is shown in Fig. 14. Obviously, the boundary of such a region is simply the contour $M = 1$. These contours are shown in Fig. 14.

The configuration of supersonic zones differs markedly from that one which may be obtained for the same flow being treated as inviscid. First, both zones are much smaller than in the inviscid case. Second, which may seem very unusual, downstream boundaries of the zones are not of the shock type. Indeed, we have here a certain kind of continuous transition from supersonic to subsonic flow which is shown in Fig. 15: a fragment of Mach contours near the lower surface of the airfoil.

We would like to mention that the intensity of zones is very low. The maximal value of the Mach number inside the supersonic regions is about $M_{\max} \approx 1.1$. This low intensity may be the reason why the finite-difference scheme does not resolve the very weak shock and smoothes it over, which numerically results in the continuous transition mentioned above.

Anyway, we have obtained a solution of a very non-typical structure for the case of supercritical flow past an airfoil. We explain this unusual character of the solution by the influence of large viscosity (low Re laminar flow), which seems rather natural. However, we have no reference to any experiments conducted with the similar flow parameters. Therefore, in the meantime we may consider this solution only as a product of the specific numerical solver [16, 21, 22]. We note that we have the same solution for both types of ABCs (for standard conditions we have obtained the same result for a larger domain when convergence takes place), which means that the method of treating the external boundary may influence the properties of the numerical procedure (convergence) but it does not exert the essential influence on the solution itself.

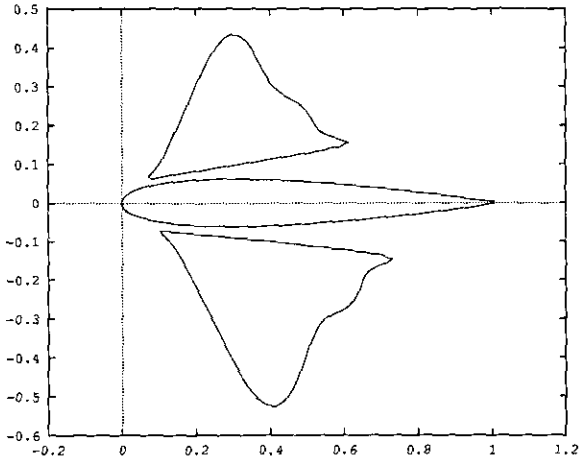

FIG. 14. Configuration of supersonic zones for viscous flow past NACA0012, $M_0 = 0.85$, $\alpha = 1°$, Re = 4000, grid 256 × 64, "average radius" ≈5.5 chords.

Moreover, it turns out that for a larger Reynolds number (Re = 40,000) convergence fails for both types of external boundary conditions. The iteration procedure does not blow up, but the solution has an explicit oscillatory character while being developed in "time" ("·" because of multigrid) which presumably implies that the steady-state solution simply does not exist in this case.

A further increase in Reynolds number finally leads to turbulent flows. This case is undoubtedly the most interesting for practical applications. The usual modern approach to computation of turbulent flows requires certain models of turbulence. For example, the code [16, 21, 22] involves an algebraic model (of Baldwin–Lomax type). The turbulence models are essentially nonlinear and non-isotropic. They are most important for computations in a close-to-the-body region. On the other hand, such a sophisticated treatment is presumably not relevant to the flow in the far field, where it is sufficient to use a concept of "effective turbulent viscosity" which dates back to Boussinesq [18]. We already have some reasonable initial results of turbulent flow computations using nonlocal ABCs, and they will be reported in a future paper.

4. CONCLUSIONS

We developed and numerically implemented the nonlocal ABCs for external viscous flow computations. These ABCs are

equally easy to apply to computational domains of irregular shape. They provide very fast convergence to a steady state (in comparison with standard conditions) and also enable shrinkage of the computational domain without loss of accuracy.

The nonlocal ABCs are based on the linearization of the governing equations (full Navier–Stokes equations) in the far field and then on the application of the difference potentials method. Comparison of the results obtained while using these conditions with those obtained from standard (extrapolation) conditions justifies the use of far-field linearization in all the cases under study.

When comparing the behavior of two curves (corresponding to standard and nonlocal ABCs, respectively) on any graph representing convergence dynamics (Figs. 2–12) one can easily see that the *standard* curve has a breakpoint not far from the beginning and at this point its slope decreases, whereas the *nonlocal* curve has a constant slope almost everywhere. This presumably means that the influence of external boundary conditions reveals itself not at the very beginning of the iteration process but after some time, i.e., after some number of iterations, since before the breakpoint the slopes of the two curves coincide. The existence of a breakpoint on a *standard* curve apparently implies that after this moment spurious reflections from the external boundary drastically slow down the convergence for standard conditions. The nonlocal ABCs have presumably far better non-reflecting properties, and the convergence rate therefore remains constant (constant slope) and very fast, which is seen from the figures.

We plan to generalize this technique for more complicated cases, such as transonic (more careful study) and/or turbulent, as well as for other geometries: ducts, nozzles, etc. Moreover, we believe that it is possible to substantially reduce the CPU time expenditure for computing the operator **T** (on the basis of parabolized Navier–Stokes equations), which is important for practical applications.

ACKNOWLEDGMENTS

I am very grateful to Professor Victor Ryaben'kii in cooperation with whom the work on nonlocal ABCs for viscous flow computations was started. Professor Saul Abarbanel and Professor Eli Turkel are gratefully acknowledged for numerous valuable and fruitful discussions which undoubtedly helped to substantially improve this paper. I am also very thankful to Professor Eli Turkel for giving me the opportunity to use the Navier–Stokes code.

REFERENCES

1. A. Bayliss and E. Turkel, *Commun. Pure Appl. Math.* **33,** 707 (1980).
2. A. Bayliss and E. Turkel, *SIAM J. Sci Statist. Comput.* **3,** 250 (1982).
3. A. Bayliss and E. Turkel, *J. Comput. Phys.* **48,** 182 (1982).
4. A. Bayliss, M. Gunzburger, and E. Turkel, *SIAM J. Appl. Math.* **42,** 430 (1982).
5. B. Engquist and A. Majda, *Math. Comput.* **31,** 629 (1977).
6. B. Engquist and A. Majda, *Commun. Pure Appl. Math.* **32,** 313 (1979).
7. B. Engquist and A. Majda, *J. Comput. Phys.* **40,** 91 (1981).
8. L. Ferm, *J. Comput. Phys.* **91,** 55 (1990).
9. L. Ferm, Non-reflecting accurate open boundary conditions for the steady Euler equations, Technical report, Department of Scientific Computing, Uppsala University, Uppsala, Sweden, September 14, 1992 (unpublished).
10. L. Ferm and B. Gustafsson, *Comput. Fluids* **10,** 261 (1982).
11. D. Givoli and J. B. Keller, *Comput. Meth. Appl. Mech. Eng.* **76,** 41 (1989).
12. D. Givoli, *J. Comput. Phys.* **94,** 1 (1991).
13. D. Givoli, *Numerical Methods for Problems in Infinite Domains* (Elsevier, Amsterdam, 1992).
14. L. Halpern, *Math. Comput.* **46,** 425 (1986).
15. G. W. Hedstrom, *J. Comput. Phys.* **30,** 222 (1979).
16. A. Jameson, W. Schmidt, and E. Turkel, Numerical solutions of the Euler equations by finite volume methods using Runge–Kutta time-stepping schemes, AIAA Paper 81-1259, 1981 (unpublished).
17. H. Jiang and Y. S. Wong, *J. Comput. Phys.* **88,** 205 (1990).
18. L. G. Loytsyansky, *Mechanics of Fluid and Gas* (Nauka, Moscow, 1987). [Russian]
19. V. S. Ryaben'kii and S. V. Tsynkov, *SIAM J. Numer. Anal.*, to appear; Tech. Rpt. 5-93, Dept. of Appl. Math., Tel-Aviv University, Jan. 1993, (unpublished).
20. V. S. Ryaben'kii, *Difference Potentials Method for Some Problems of Continuous Media Mechanics* (Moscow, Nauka, 1987). [Russian]
21. R. C. Swanson and E. Turkel, A multistage time-stepping scheme for the Navier–Stokes equations, AIAA Paper 85-0035, 1985 (unpublished).
22. R. C. Swanson and E. Turkel, Artificial dissipation and central difference schemes for the Euler and Navier–Stokes equations, AIAA Paper 87-1107-CP, 1987 (unpublished).
23. K. W. Thompson, *J. Comput. Phys.* **68,** 1 (1987).
24. L. N. Trefethen and L. Halpern, *Math. Comput.* **47,** 421 (1986).