Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 190 (2003) 22–41
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Anisotropic grid adaptation for Navier–Stokes' equations

Lars Ferm, Per Lötstedt *

Department of Information Technology, Scientific Computing, Uppsala University, P.O. Box 337, SE-75105 Uppsala, Sweden

Received 24 June 2002; received in revised form 8 April 2003; accepted 21 April 2003

Abstract

Navier–Stokes' equations are discretized in space by a finite volume method. Error equations are derived which are approximately satisfied by the errors in the solution. The dependence of the solution errors on the discretization errors is analyzed in certain flow cases. The grid is adapted based on the estimated discretization errors. The refinement and coarsening of the grid are anisotropic in the sense that it is different in different directions in the computational domain. The adaptation algorithm is applied to laminar, viscous flow over a flat plate, in a channel with a bump, and around a cylinder and an airfoil.

Keywords: Finite volume method; Anisotropic grid refinement; Error equation; Viscous flow

1. Introduction

The steady-state solution of the incompressible and compressible Navier–Stokes equations is computed by a grid adaptive procedure in two space dimensions in this paper. The equations are discretized by a second-order accurate finite volume method. New cells are introduced and concentrated in areas of the computational domain where an error estimate indicates that the discretization (or truncation) error is too large. In this way, computing time and memory are saved compared to having a fine grid over the whole domain.

The grids are structured and partitioned into blocks. All cells in a block are refined or coarsened. The grid size may change discontinuously between two blocks with a common interface. The alternative is to refine single cells by splitting them into four (2D) or eight (3D) cells. The advantages with adaptivity in blocks of cells are that the administration is reduced in particular for time-dependent problems, it is easier to maintain second-order accuracy in all cells and the parallelization of the code is simplified. The adaptation algorithm here generates anisotropic grids with refinement only in one grid direction. Such grids are well suited for the Navier–Stokes equations where the solution in a laminar boundary layer often varies slowly in the streamwise direction and rapidly normal to the wall. The method is applicable also to quadrilaterals in unstructured grids. Such grids are often organized in a 'structured' way close to solid surfaces.

* Corresponding author.
E-mail addresses: ferm@tdb.uu.se (L. Ferm), perl@tdb.uu.se (P. Lötstedt).

0021-9991/$ - see front matter
doi:10.1016/S0021-9991(03)00250-X

The discretization errors are estimated by comparing the discretization on a fine grid with the discretization on coarser grids. These estimates determine where the original grid is changed. It is more difficult to estimate the errors in the solution. These errors satisfy error equations approximately. The error equations are solved using fundamental solutions and Green's functions in four examples. There the solution errors are bounded by the discretization errors in the supremum norm. While it is possible to construct examples where the discretization errors are propagated and accumulated so that the solution errors are much larger, in many cases in practice they are of the same order of magnitude. Hence, it is often sufficient to decide where to refine and coarsen a grid using the discretization error. This is the strategy when the time-step is selected in adaptive methods for ordinary differential equations [26]. The size of the residual is also the interruption criterion in iterative solvers of systems of linear equations.

The cells of two adjacent blocks overlap at the interfaces. High order interpolation is necessary there for the discretization to be of second order also for second derivatives. The accuracy and stability of interpolation schemes are investigated in [10].

Other work on adaptive methods for flow problems is reviewed briefly. A block adaptive algorithm for inviscid flow is developed in [9]. The discretization errors are estimated and the solution errors in certain variables are controlled. A more complicated data structure for the grids, compared to what we have here and in [9], is found in [3] with examples of inviscid flow including shocks. Similar techniques for viscous flow are described in [15,27]. The grid is refined in patches also in [29] and the Navier–Stokes equation is solved in parallel. Laminar flow around an airfoil is computed in [11]. Other related applications are magnetohydrodynamics [22], porous flow [14], and the plasma fluid equations [5].

Viscous flow problems are solved in [2] by the finite element method and a posteriori error estimation. Weights are computed with an adjoint equation. Introductions to such techniques are found in [7,23]. These weights determine the influence of a discretization error on the error in functionals of engineering interest. The advantage of unstructured grids is their ability to represent geometrically complicated objects. An example of adaptive techniques for the Navier–Stokes equations is [19]. An alternative to adding more cells is to keep the number of cells constant but to move them to areas where more resolution is needed as in [13]. The cells are easily skewed with such a method and crossings of grid lines are difficult to avoid. Time-dependent parabolic problems are solved in [30] and hyperbolic problems in [18] with adaptivity in space and time. Flame fronts in combustion are followed by adaptive grids in [12,25].

The finite volume discretization of the equations is discussed in the following section. The interpolation of the variables in the overlapping ghost cells at the block interfaces is taken from [10]. The error equations satisfied by the solution errors are derived and solved in four special cases in Section 3. Then the algorithm for anisotropic grid adaptation is given in Section 4 including a result for the discretization error estimates. In Section 5 the algorithm is applied to four different examples: viscous flow over a flat plate, in a channel with a bump, around a cylinder and around an airfoil. The range of the Reynolds number Re is between 30 and 5000 and the flow is laminar and incompressible or compressible. In the computational domain, a general norm is denoted by $\|\cdot\|$, the supremum norm by $|\cdot|_\infty$, and the Euclidean vector norm by $\|\cdot\|_2$.

2. Navier–Stokes' equations and their discretization

Let $w = (\rho, \rho u, \rho v, \rho E)^{\mathrm{T}}$ be the state vector, where ρ is the density, u and v are the velocity components in the x- and y-directions, and E is the total energy. Then the Navier–Stokes equations in two dimensions for the time-independent flow of a compressible fluid are in conservation form

$$\frac{\partial f(w, \nabla w)}{\partial x} + \frac{\partial g(w, \nabla w)}{\partial y} = 0. \tag{1}$$

The flux vectors f and g are defined by

$$f = \begin{pmatrix} \rho u \\ \rho u^2 + p - \mu S_{11} \\ \rho uv - \mu S_{21} \\ (\rho E + p)u - \mu(uS_{11} + vS_{21}) \end{pmatrix}, \qquad g = \begin{pmatrix} \rho v \\ \rho uv - \mu S_{12} \\ \rho v^2 + p - \mu S_{22} \\ (\rho E + p)v - \mu(uS_{12} + vS_{22}) \end{pmatrix}, \tag{2}$$

where μ is the viscosity coefficient and S is the strain rate tensor defined by

$$S = \begin{pmatrix} \frac{4}{3}u_x - \frac{2}{3}v_y & u_y + v_x \\ u_y + v_x & \frac{4}{3}v_y - \frac{2}{3}u_x \end{pmatrix}. \tag{3}$$

A subscript x or y denotes differentiation with respect to the variable. The pressure p satisfies

$$p = (\gamma - 1)\rho(E - 0.5(u^2 + v^2))$$

and the gas constant $\gamma = 1.4$. The corresponding fluxes for incompressible flow are

$$f = \begin{pmatrix} u \\ u^2 + p - \mu u_x \\ uv - \mu v_x \end{pmatrix}, \qquad g = \begin{pmatrix} v \\ uv - \mu u_y \\ v^2 + p - \mu v_y \end{pmatrix}, \tag{4}$$

with the solution $w = (p, u, v)^{\mathrm{T}}$.

Eq. (1) is discretized by a finite volume method on a structured grid. It is integrated over a cell ω_j with boundary $\partial\omega_j$, normal (n_1, n_2), and area A_j using Gauss' formula

$$A_j^{-1} \int_{\omega_j} (f_x + g_y)\, \mathrm{d}S = A_j^{-1} \int_{\partial\omega_j} (fn_1 + gn_2)\, \mathrm{d}s. \tag{5}$$

The cell average $\overline{w}_j$ of the solution w in cell j,

$$\overline{w}_j = A_j^{-1} \int_{\omega_j} w\, \mathrm{d}S,$$

is computed with an approximation of w and ∇w on $\partial\omega_j$ in (5). The convective part of f and g is determined by the second-order Jameson scheme [16,28], or the second-order Osher scheme [21,28]. The gradient of a component w_k of w is computed in a dual grid with the midpoints in the primary grid as cell corners. In such a dual cell, the average of the gradient $\overline{\nabla w_k}$ is approximated by invoking Gauss' formula again

$$\overline{\nabla w_k} = A_j^{-1} \int_{\omega_j} \nabla w_k\, \mathrm{d}S = A_j^{-1} \int_{\partial\omega_j} w_k n\, \mathrm{d}s. \tag{6}$$

Using these values in the dual grid the viscous terms in f and g are computed at $\partial\omega_j$ by averaging. A more detailed account is found in [10].

The computational grid is partitioned into blocks. Jumps by a factor two in the grid size are allowed at the block interfaces as in [9,10]. In [9] the grid is refined isotropically in the normal *and* the tangential directions at the block interface. Here, we allow the grid refinement to be anisotropic with finer resolution in the normal *or* the tangential directions (or both).

Every block has two rows of ghost cells overlapping the interior cells of the neighboring block. These cells simplify the flux computations in the cells with a face on a block boundary. If the ghost cells coincide with the cells in the neighboring block then the variables in the ghost cells are copied. A block interface with different grid sizes at $x = 0$ is illustrated in Fig. 1. If one ghost cell in a coarse block corresponds to two or four cells in a fine block, then the ghost value is computed by an area weighted average (cells (1,0) and (2,0)

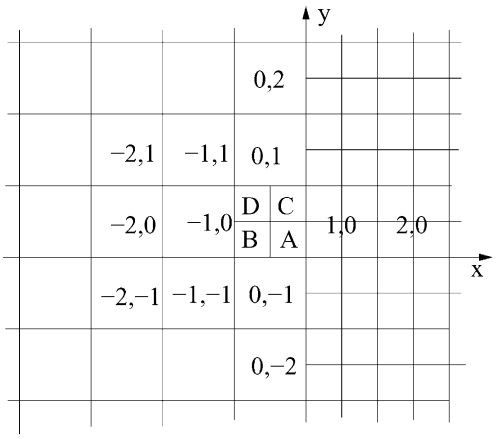

Fig. 1. The grid and a block interface at $x = 0$.

in Fig. 1). When two or four ghost cells overlap one interior coarse cell then the variables there are determined by interpolation between the values in the neighboring block (cells A, B, C, and D in Fig. 1). It is shown in [10] that fourth-order accuracy is needed in the interpolation to obtain a second-order accurate discretization of (1). The stability of the boundary treatment is investigated theoretically and experimentally in [10]. It is found that for a centered difference approximation of the first derivative in a convection–diffusion equation to be stable there is a lower bound on the physical diffusion μ or the added artificial dissipation for all the tested interpolation formulas. An upwind scheme at the block interface stabilizes the discretization.

The solution $w_I,\ I = A, B, C, D$, in a ghost cell of the fine grid to the left of $x = 0$ in Fig. 1 is interpolated from the interior solutions $w_{jk},\ j = -2, -1, 0, k = -2, \ldots, 2$, and the ghost cell solutions $w_{j,0},\ j = 1, 2$, of the coarse grid. The fourth-order formulas for w_A, w_B, w_C, and w_D corresponding to a weighted sum of two symmetric interpolation schemes $0.5H_6^* + 0.5H_7^*$ in [10]. Let

$$
\begin{aligned}
&w_{AC} = (-3w_{2,0} + 17w_{1,0} + 55w_{0,0} - 5w_{-1,0})/64, \quad w_{BD} = 2w_{0,0} - w_{AC},\\
&d_j = w_{-j,1} - w_{-j,-1}, \quad g = w_{0,2} - w_{0,-2},\\
&s_0 = (14d_0 - 4d_1 + d_2 + 1.5g)/64, \quad s_1 = (22d_0 - 3g)/64 - s_0.
\end{aligned}
$$

Then the formulas are

$$
w_A = w_{AC} + s_0, \quad w_B = w_{BD} + s_1, \quad w_C = w_{AC} - s_0, \quad w_D = w_{BD} - s_1. \tag{7}
$$

If the fine grid is refined in only one direction then a ghost cell variable is obtained by a suitable summation of the interpolation formulas for the cells A, B, C, and D.

3. Error analysis

A discretization error τ_j in a cell ω_j is caused by the approximation of the integral (5) by averaged variables $\overline{w}$ in adjacent cells. Let $F_j(w)$ denote the integral in (5) and $\Psi_j(\overline{w})$ its numerical approximation. Then for any smooth w we have

$$
\tau_j(w) = F_j(w) - \Psi_j(\overline{w}).
$$

A smooth reconstruction $\tilde{w}$ of the numerical solution from $\overline{w}$ satisfies $\tau_j(\tilde{w}) = F_j(\tilde{w})$ since $\overline{\tilde{w}} = \overline{w}$ and $\Psi_j(\overline{w}) = 0$. Let $\hat{w}$ be the analytical solution. Then the solution error $\delta w = \tilde{w} - \hat{w}$ fulfills

$$F_j(\hat{w} + \delta w) - F_j(\hat{w}) = F_j(\tilde{w}) = \tau_j(\tilde{w}). \tag{8}$$

This equation is written in differential form and linearized to obtain the error equation. The error equation is investigated for incompressible flow (4) in this section. A derivation and a discussion of the error equation for finite volume methods are found in [9].

The equation for incompressible flow in non-conservative form with flux functions in (4) corresponding to (8) is

$$\begin{aligned} &uu_x + vu_y + p_x - \mu(u_{xx} + u_{yy}) = \tau_1, \\ &uv_x + vv_y + p_y - \mu(v_{xx} + v_{yy}) = \tau_2, \\ &u_x + v_y = \tau_3. \end{aligned} \tag{9}$$

Insert $u = \hat{u} + \delta u$, $v = \hat{v} + \delta v$, and $p = \hat{p} + \delta p$ into (9) and subtract (9) with $\tau = 0$ satisfied by $\hat{u}$, $\hat{v}$, and $\hat{p}$. After dropping quadratic terms in the error we arrive at the error equation for incompressible flow

$$\begin{aligned} &\hat{u}\delta u_x + \hat{u}_x\delta u + \hat{v}\delta u_y + \hat{u}_y\delta v + \delta p_x - \mu(\delta u_{xx} + \delta u_{yy}) = \tau_1, \\ &\hat{u}\delta v_x + \hat{v}_x\delta u + \hat{v}\delta v_y + \hat{v}_y\delta v + \delta p_y - \mu(\delta v_{xx} + \delta v_{yy}) = \tau_2, \\ &\delta u_x + \delta v_y = \tau_3. \end{aligned} \tag{10}$$

The solution errors δu, δv, δp depend in a complicated way on the errors τ_j in (10). Due to a transport of the discretization error τ, they may be affected by errors generated far away. In general, diffusion terms cause the errors to decay slowly and the convection terms propagate the discretization errors in the flow direction but their effect in the crosswind direction vanishes quickly. These observations are made in the following four simple flow examples.

3.1. Poiseuille flow

Parallel flow through a straight channel is particularly simple. The flow direction is along the x-coordinate and the y-direction is normal to the walls. The pressure gradient in the flow direction is constant, $\hat{v} = 0$ and $\hat{u} = \hat{u}(y)$ in the analytical solution [24]. The resulting velocity $\hat{u}$ is parabolic in y. The error equation is

$$\begin{aligned} &\hat{u}\delta u_x + \hat{u}_y\delta v + \delta p_x - \mu(\delta u_{xx} + \delta u_{yy}) = \tau_1, \\ &\hat{u}\delta v_x + \delta p_y - \mu(\delta v_{xx} + \delta v_{yy}) = \tau_2, \\ &\delta u_x + \delta v_y = \tau_3. \end{aligned} \tag{11}$$

Assume that the discretization is such that the numerical solution shares the properties of the analytical solution so that $v = 0$ and $u = u(y)$ and that the linear variation of p in the channel is exactly represented. Then $\delta p = 0$ and τ_1 originates from the discretization of μu_{yy}. The error equation (11) is then further simplified to

$$-\mu\delta u_{yy} = \tau_1, \tag{12}$$

with boundary conditions $\delta u(0) = \delta u(1) = \delta u_0$ at the channel walls. The symmetric errors in the boundary conditions are modeled by δu_0. The solution of (12) is

$$\delta u = \delta u_0 + \mu^{-1}\int_0^1 g(y,\eta)\tau_1(\eta)\,\mathrm{d}\eta, \quad g(y,\eta) = \begin{cases} y(1-\eta), & 0 < y < \eta, \\ \eta(1-y), & \eta \leqslant y < 1, \end{cases} \tag{13}$$

with the Green's function g. It follows from (13) that if $|\tau_1|_\infty \leqslant \epsilon$ then

$$|\delta u|_\infty \leqslant |\delta u_0| + \epsilon/8\mu. \tag{14}$$

3.2. Convective flow with diffusion

Away from the boundary layers we assume that $\hat{u}$ and $\hat{v}$ are constant and that the gradient of the error in the pressure $\nabla\delta p$ is small. Then (11) is simplified to two convection–diffusion equations

$$\hat{u}\delta u_x + \hat{v}\delta u_y - \mu(\delta u_{xx} + \delta u_{yy}) = \tau_1, \tag{15a}$$

$$\hat{u}\delta v_x + \hat{v}\delta v_y - \mu(\delta v_{xx} + \delta v_{yy}) = \tau_2. \tag{15b}$$

The solution of (15a) in free space is given by the fundamental solution [20]

$$\delta u(x,y) = \frac{1}{2\pi\mu}\int_S \exp(\mathbf{U}\cdot\mathbf{r}/2\mu)K_0(Ur/2\mu)\tau_1\,\mathrm{d}S, \tag{16}$$

where S is the domain of support of τ_1 and $(\xi,\eta)\in S$, K_0 is a modified Bessel function, $\mathbf{U}^{\mathrm{T}} = (\hat{u},\hat{v})$, $\mathbf{r}^{\mathrm{T}} = (x-\xi, y-\eta)$, $U = \|\mathbf{U}\|_2$, and $r = \|\mathbf{r}\|_2$. The error δu can be estimated from (16) in the following way.

Proposition 3.2. *Assume that* $|\tau_1|_\infty \leqslant \epsilon$, *$S$ is a circle of radius Δr with center at the origin*, $\mathbf{r}_0^{\mathrm{T}} = (x,y)$, $r_0 = \|\mathbf{r}_0\|_2$, *and that $\Delta r/r_0$ is sufficiently small. Let $\phi_0 \in (-\pi,\pi]$ be the angle between* $\mathbf{U}$ *and* $\mathbf{r}_0$, $\Delta\phi = \arcsin \Delta r/r_0$, *and* $\rho_0 = Ur_0/2\mu$. *Then δu in* (16) *satisfies*

$$|\delta u(x,y)| \leqslant \frac{\epsilon}{\mu}\sqrt{\frac{\pi}{8\rho_0}}\exp(\rho_0(1-\Delta r/r_0)(\cos\hat{\phi}-1))f(\rho_0,\Delta r/r_0),$$

where $f = \Delta r^2(1+\mathcal{O}(1/\rho_0)+\mathcal{O}(\Delta r/r_0))$ *and*

$$\hat{\phi} = \begin{cases} 0, & |\phi_0| \leqslant \Delta\phi, \\ \phi_0 - \Delta\phi, & \phi_0 > \Delta\phi, \\ \phi_0 + \Delta\phi, & \phi_0 < -\Delta\phi. \end{cases}$$

Proof. With $\rho = Ur/2\mu$, an upper bound on δu is

$$|\delta u(x,y)| \leqslant \frac{\epsilon}{2\pi\mu}\int_S \exp(\rho\cos\phi)K_0(\rho)\,\mathrm{d}S, \tag{17}$$

where ϕ is the angle between $\mathbf{U}$ and $\mathbf{r}$. For ρ we have

$$\rho \geqslant \frac{U}{2\mu}(r_0 - \Delta r) = \rho_0(1-\Delta r/r_0). \tag{18}$$

The angle ϕ varies in $[\phi_0 - \Delta\phi, \phi_0 + \Delta\phi]$ (see Fig. 2). Hence,

$$\max_\phi(\cos\phi - 1) \leqslant \cos\hat{\phi} - 1 \leqslant 0. \tag{19}$$

Combine (18) and (19) to obtain

$$\max\exp(\rho(\cos\phi-1)) \leqslant \exp(\rho_0(1-\Delta r/r_0)(\cos\hat{\phi}-1)). \tag{20}$$

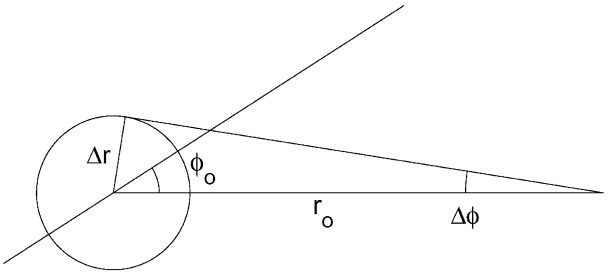

Fig. 2. Illustration of the definitions in Proposition 3.2.

The asymptotics for $K_0(\rho)$ for large ρ is [1]

$$K_0(\rho) = \exp(-\rho)\sqrt{\frac{\pi}{2\rho}}\left(1 - \frac{1}{8\rho} + \cdots\right). \tag{21}$$

Expand the square root in (21) using (18) to arrive at

$$\sqrt{\frac{\pi}{2\rho}} \leqslant \sqrt{\frac{\pi}{2\rho_0}}\frac{1}{\sqrt{1-\Delta r/r_0}} = \sqrt{\frac{\pi}{2\rho_0}}(1 + \mathcal{O}(\Delta r/r_0)). \tag{22}$$

Insert (22) into (21). Multiply (21) with the upper bound in (20) and introduce the expression into (17). Integrate over S to obtain the proposition. □

We find that in the flow direction for small ϕ_0 the influence of the discretization error τ_1 decreases with the distance as $1/\sqrt{r_0}$. The error decays exponentially in all other directions. When $\hat{\phi} \neq 0$ and r_0 is fixed, δu vanishes as $\exp(-c/\mu)/\sqrt{\mu}$ for some $c > 0$ with decreasing μ. The same result is valid for δv and τ_2 in (15b).

3.3. Stokes' flow

In fluids with slowly varying flow at low speeds, $\hat{u}, \hat{v}$, and their derivatives are small. For slow flow, μ is usually large. If we ignore small terms then the Navier–Stokes equations boil down to Stokes' equations. At a stagnation point in 2D $\hat{u} = \hat{v} = 0$, but their derivatives can be large and Stokes' flow is not necessarily a good approximation there.

The error (10) for Stokes' equations are

$$\nabla\delta p - \mu\Delta\delta U = \tau_{12}, \tag{23a}$$

$$\nabla \cdot \delta U = \tau_3, \tag{23b}$$

with $\delta U^{\mathrm{T}} = (\delta u, \delta v)$ and $\tau_{12}^{\mathrm{T}} = (\tau_1, \tau_2)$. Apply $\nabla\cdot$ to (23a) to obtain

$$\Delta(\delta p - \mu\tau_3) = \nabla \cdot \tau_{12}. \tag{24}$$

Suppose that the support of τ is inside S so that $\tau = 0$ and $\nabla\tau = 0$ on ∂S and that $\mathbf{r}$ and r are defined as in Section 3.2. With the fundamental solution to Poisson's equation $\log r/2\pi$ and Gauss' theorem we derive

$$\delta p = \mu\tau_3 + \frac{1}{2\pi}\int_S \nabla \cdot \tau_{12} \log r \, \mathrm{d}S = \mu\tau_3 + \frac{1}{2\pi}\int_S \frac{\tau_{12} \cdot \mathbf{r}}{r^2}\, \mathrm{d}S. \tag{25}$$

With the same assumptions as in Proposition 3.2 and $|\tau_k|_\infty \leqslant \epsilon,\ k = 1, 2$, a bound on δp is

$$|\delta p(x,y)| \leqslant \mu|\tau_3| + \frac{1}{2\pi}\int_S \frac{\|\tau_{12}\|_2\|\mathbf{r}\|_2}{r^2}\,\mathrm{d}S \leqslant \mu|\tau_3(x,y)| + \frac{\epsilon\Delta r^2}{\sqrt{2}r_0}(1+\mathcal{O}(\Delta r/r_0)). \tag{26}$$

The equation satisfied by δU is derived by operating with Δ on (23a), with ∇ on (24), and replacing $\nabla\Delta$ by $\Delta\nabla$ and inserting $\Delta\nabla\delta p$ into (23a) to obtain

$$\mu\Delta\Delta\delta U = \mu\nabla\Delta\tau_3 - \Delta\tau_{12} + \nabla\nabla\cdot\tau_{12}. \tag{27}$$

The fundamental solution of the biharmonic equation in (27) is $g = r^2(\log r - 1)/8\pi$ [6]. The dependence of δU on τ_3 is determined by repeated application of Gauss' theorem

$$\delta U = \int_S g\nabla\Delta\tau_3\,\mathrm{d}S = -\int_S \nabla\Delta g\tau_3\,\mathrm{d}S = \frac{1}{2\pi}\int_S \mathbf{r}\frac{\tau_3}{r^2}\,\mathrm{d}S.$$

As in (25) and (26) we find with $|\tau_3|_\infty \leqslant \epsilon$ that

$$\|\delta U(x,y)\|_2 \leqslant \frac{\epsilon\Delta r^2}{2r_0}(1+\mathcal{O}(\Delta r/r_0)). \tag{28}$$

The relation between τ_{12} and δU is more complicated but can be obtained as above. However, δU derived from the fundamental solution does not decay as $1/r$ for increasing r in 2D.

3.4. Boundary layer flow

In a laminar boundary layer over a flat plate at $y = 0$ assuming that p is constant, an accurate approximation is the boundary layer equation

$$\begin{aligned} &\hat{u}\hat{u}_x + \hat{v}\hat{u}_y - \mu\hat{u}_{yy} = 0,\\ &\hat{u}_x + \hat{v}_y = 0,\\ &\hat{u}(x,0) = \hat{v}(x,0) = 0, \hat{u}(x,\infty) = \hat{u}_\infty. \end{aligned} \tag{29}$$

Blasius obtained a solution by transforming the equation to a nonlinear ordinary differential equation with solution f and expressing f in a new independent variable $\eta = y\sqrt{\hat{u}_\infty/\mu x}$, see [17,24]. For a given small μ, downstream of the leading edge at $x > 0$ and close to the plate with η and y approaching 0, the solution of (29) satisfies

$$\begin{aligned} &\hat{u} \sim \eta, \quad \hat{u}_x \sim \eta, \quad \hat{u}_y \sim 1/\sqrt{\mu}, \quad \hat{u}_{yy} \sim \eta^2/\sqrt{\mu},\\ &\hat{v} \sim \sqrt{\mu}\eta^2, \quad \hat{v}_x \sim \sqrt{\mu}\eta^2, \quad \hat{v}_y \sim \eta. \end{aligned}$$

Ignore terms of order η and μ in (10) and we arrive at an approximate error equation

$$\hat{u}_y\delta v + \delta p_x = \tau_1, \tag{30a}$$

$$\delta p_y = \tau_2, \tag{30b}$$

$$\delta u_x + \delta v_y = \tau_3. \tag{30c}$$

Assume that the discretization of the boundary conditions is such that $\delta v(x,0) = 0$ and $\delta p_y(x,0) = 0$. Then we derive from (30a) and (30b) that

$$\delta v(x,y) = -\hat{u}_y^{-1}\mathscr{T} = \mathcal{O}(\sqrt{\mu}), \quad \mathscr{T}(x,y) = \int_0^y \nabla \times \tau_{12}(x,\zeta)\,\mathrm{d}\zeta, \tag{31}$$

and from (30c) that

$$\begin{aligned}\delta u(x,y) &= \delta u(x_0,y) + \int_{x_0}^{x} \left(\tau_3 + \hat{u}_y^{-1}\nabla \times \tau_{12}\right)(\xi,y)\,\mathrm{d}\xi,\\ &\approx \delta u(x_0,y) + \int_{x_0}^{x} \tau_3(\xi,y)\,\mathrm{d}\xi,\end{aligned} \tag{32}$$

for some upstream x_0. The error in v is small in (31) and δu is an accumulation in the streamwise direction of the error in the divergence equation and as in (14) with $|\tau_3|_\infty \leqslant \epsilon$

$$|\delta u(x,y)| \lesssim |\delta u(x_0,y)| + (x-x_0)\epsilon. \tag{33}$$

The equation for δp is

$$\Delta \delta p = \nabla \cdot \begin{pmatrix} \tau_1 + \mathscr{T} \\ \tau_2 \end{pmatrix}. \tag{34}$$

The Green's function is

$$g = (2\pi)^{-1}(\log r + \log r'), \quad r' = \sqrt{(x-\xi)^2 + (y+\zeta)^2},$$

and as in (25) and (26) δp decreases with the distance from the source of the error τ_{12}.

We have obtained upper bounds on δu, δv, and δp in (14), Proposition 3.2, (26), (28), and (33) provided $|\tau_k|_\infty \leqslant \epsilon$, $k = 1,2,3$. In many cases the effect of τ on δU and δp is local. The errors decay with the distance from the source and are bounded by τ (and not its derivatives) supporting the view that it often suffices to control the discretization error and in this way also have some control of the solution error. There is one notable exception in Section 3.2. Outside the boundary layer ρ_0 is large but in the boundary layer diffusion dominates over convection and a more damped behavior is noticed, e.g., in Figs. 10 and 11 in Section 5.

The bounds are different for different flow situations and a bound for a general case would probably be very pessimistic and not very useful in practical calculations. Computable bounds relate the discretization errors to the solution errors in a posteriori error estimation at the expense of solving an adjoint problem [2,7,23]. In our adaptation algorithm for anisotropic grids, we control the discretization error, but the algorithm can be combined with weights obtained a posteriori for the influence on the solution error.

4. Anisotropic grid adaptation

When solving the Navier–Stokes equations we need high resolution in one of the space dimensions in the boundary layers. An adaptive algorithm should hence be able to choose different grid sizes h_k and h_l in different directions k and l. Therefore, different terms will be identified in the discretization error τ. The choice of grid size will be based on these terms.

Assume that the domain Ω is covered by quadrilaterals. Let the coarse cell ω_0 consist of the fine cells ω_i, $i = 1,2,3,4$, and have twice the grid size of the fine cells as in Fig. 3. The area of a cell ω_i is A_i. Compute w_0 in all coarse cells in a block by the area weighted sum $\sum_{i=1}^4 \gamma_i w_i$, $\gamma_i = A_i/A_0$, of w_i in the original cells. Furthermore, let w_{ij} be the solution in the semi-coarsened cells ω_{ij} created by coarsening in only one direction so that ω_i and ω_j form one cell for $i,j = 1,2,3,4$. Then $w_{ij} = \alpha_i w_i + \alpha_j w_j$, $\alpha_\beta = A_\beta/(A_i + A_j)$. The discretization of (5) in ω_0 is $\psi_0 = \Psi_0(w_0)$ and $\psi_{ij} = \Psi_{ij}(w_{ij})$ in ω_{ij}. The following proposition identifies different terms in τ in the fine grid and a way to estimate them.

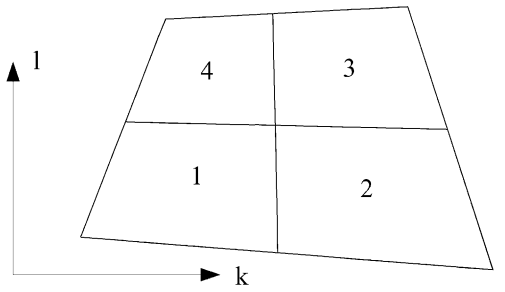

Fig. 3. A coarse cell composed of four fine grid cells.

Proposition 4. *Let h_k, h_l be the grid sizes in the k and l directions. Assume that Ψ in cell i satisfies*

$$\Psi_i(\overline{w}) = \psi_i = F_i(w) - \tau(w). \tag{35}$$

Furthermore, assume that the discretization error consists of three terms

$$\tau = \tau^k + \tau^l + \tau^{kl}, \quad \tau^k = c_k h_k^2 + d_k h^3, \quad \tau^l = c_l h_l^2 + d_l h^3, \quad \tau^{kl} = c_{kl} h_k h_l + d_{kl} h^3, \tag{36}$$

where $h = \max(h_k, h_l)$, c_k, c_l, c_{kl}, are constants for each coarse cell ω_0, and d_k, d_l, d_{kl}, are bounded independently of h for each cell ω_i. Let $\gamma_{ij} = \gamma_i + \gamma_j$ and introduce the linear combinations

$$b_1 = \gamma_{14}\psi_{14} + \gamma_{23}\psi_{23} - \psi_0, \quad b_2 = \gamma_{12}\psi_{12} + \gamma_{34}\psi_{34} - \psi_0, \quad b_3 = \sum_{j=1}^{4} \gamma_j \psi_j - \psi_0. \tag{37}$$

Then in a cell ω_i

$$\tau^k = \tfrac{1}{3}(2b_3 - 2b_2 - b_1) + \mathcal{O}(h^3), \quad \tau^l = \tfrac{1}{3}(2b_3 - 2b_1 - b_2) + \mathcal{O}(h^3), \quad \tau^{kl} = b_1 + b_2 - b_3 + \mathcal{O}(h^3). \tag{38}$$

Proof. Since $\gamma_{14} + \gamma_{23} = 1$ and $\gamma_{14}F_{14} + \gamma_{23}F_{23} = F_0$, we have

$$\begin{aligned} b_1 &= \gamma_{14}F_{14} + \gamma_{23}F_{23} - \gamma_{14}\tau_{14} - \gamma_{23}\tau_{23} - F_0 + \tau_0 \\ &= c_k h_k^2(4 - (\gamma_{14} + \gamma_{23})) + 4c_l h_l^2(1 - (\gamma_{14} + \gamma_{23})) + 2c_{kl}h_k h_l(2 - (\gamma_{14} + \gamma_{23})) + \mathcal{O}(h^3) \\ &= 3c_k h_k^2 + 2c_{kl}h_k h_l + \mathcal{O}(h^3) \end{aligned}$$

Similarly,

$$b_2 = 3c_l h_l^2 + 2c_{kl}h_k h_l + \mathcal{O}(h^3),$$

and for b_3 we derive

$$b_3 = \sum_{j=1}^{4} \gamma_j F_j - F_0 + \tau_0 - \sum_{j=1}^{4} \gamma_j \tau_j = 3c_k h_k^2 + 3c_l h_l^2 + 3c_{kl}h_k h_l + \mathcal{O}(h^3).$$

A system of linear equations is satisfied by the leading terms of τ^j, τ^k, and τ^{jk}. The solution of that system is (38). □

Remark. For the proposition to be useful in the estimation of discretization errors, the assumption is that h_k and h_l are so small that the first quadratic term in τ^k, τ^l, and τ^{kl} dominates in (36) and τ behaves in the asymptotically expected way. This is not the case, e.g., if d is larger than the corresponding c and h is not sufficiently small. Such a situation may occur at shocks and the discretization error estimates are not reliable there. Still, they act as a sensor for refinement since they are large at such points. Examples of shock solutions are found in [9].

The solution is computed first on the original grid with even number of cells in both directions. Then we construct one grid with doubled grid size in the k-direction, one grid with doubled grid size in the l-direction, and one grid with doubled grid size in both directions. The solution on the original grid is restricted by area weighted summation to the three coarser grids. We estimate the discretization errors on the three grids by first computing ψ_0, ψ_j, $j = 1, 2, 3, 4$, ψ_{12}, ψ_{14}, ψ_{23}, and ψ_{34} following Proposition 4 and then calculating the three leading error terms in τ valid for the four grid cells forming one coarse cell. The cost for this is small compared to the many evaluations of Ψ in the iterative solution of the nonlinear equations. The error terms for a second-order method after refinement in the k- and l-directions ($\tilde{\tau}_k$ and $\tilde{\tau}_l$) and in both directions ($\tilde{\tau}_{kl}$) are then estimated to be

$$\tilde{\tau}_k = \tau^k/4 + \tau^l + \tau^{kl}/2, \quad \tilde{\tau}_l = \tau^k + \tau^l/4 + \tau^{kl}/2,$$
$$\tilde{\tau}_{kl} = (\tau^k + \tau^l + \tau^{kl})/4 = \tau/4.$$

Compute $\tau, \tilde{\tau}_k$, and $\tilde{\tau}_l$ for all cells in a block in the coarse grid using $\tau^k, \tau^l, \tau^{kl}$. Choose a norm $\|\cdot\|$ in a block j, denote it by $\|\cdot\|_j$, and assume that a discretization error below a certain tolerance ϵ is required. Then the adaptation algorithm for refinement is

while $\max_j \|\tau\|_j > \epsilon$
  $m = \arg\max_j \|\tau\|_j$
  if $\|\tilde{\tau}_k\|_m \leqslant \min(\|\tilde{\tau}_l\|_m, \|\tau\|_m)$
    then refine in k direction in block m
  elseif $\|\tilde{\tau}_l\|_m \leqslant \|\tau\|_m$
    then refine in l direction in block m
  else
    then refine in both directions in block m
  endif
  recalculate $\tau, \tilde{\tau}_k, \tilde{\tau}_l$ in block m
  refine neighboring blocks recursively if necessary
end while

In the algorithm we try to reduce the measured error by refining the grid in one direction where it has the best effect. If that is not possible then the cells in a block are divided into four new cells. Then $\tau, \tilde{\tau}_k$, and $\tilde{\tau}_l$ are updated using the same τ^k, τ^l, and τ^{kl} without recomputing the ψs. Then adjacent coarser blocks are refined recursively at block interfaces where there are jumps in grid size by a factor greater than two. Cancellation between error terms in (36) can be lost when a block is refined in only one direction. It may be necessary to refine in both directions even if there is a jump in only one direction. This can lead to occasional unnecessary refinements. This procedure is repeated until the maximum τ is sufficiently small and a new adapted grid has been found.

If h_k and h_l vary smoothly inside a block in the initial grid then the same variation is inherited by the grid sizes in the refined grid. By starting the adaptation on a coarsened initial grid, the final grid may actually be coarser than the initial grid in some blocks. Repeated application of the algorithm is possible after solving the equations for a new w and decreasing ϵ. This adaptation method can also be used to improve an unstructured grid consisting of quadrilaterals and partitioned into blocks.

5. Numerical results

The Navier–Stokes equations are solved in four test problems with the adaptation algorithm in Section 4. The second-order Osher scheme [21,28], is used for compressible flows, and incompressible flows are approximated by the Jameson scheme with artificial dissipation [16,28].

The steady-state solution is computed by first adding a time derivative $\partial w/\partial t$ to (1) and then solving the time-dependent equations. The equations are integrated to a stationary solution by a three-stage Runge–Kutta scheme accelerated by multigrid iteration as in [8]. Three or four multigrid levels are used on the initial and adapted grids. The initial grids are not partitioned into blocks as the adapted grids and the computational work there is always relatively small. The time-dependent incompressible equations (1) and (4) are transformed to hyperbolic equations by adding the term $c^{-2}\partial p/\partial t$ to the divergence equation as in [4]. In the scaling factor c^{-2}, c can be regarded as an artificial speed of sound. It is chosen to be $c = 3$ in the experiments.

The initial grid and the block partitioning are chosen such that there is at least one tangential block interface in the boundary layer. The consequence of this is that also the initial grids are stretched with high aspect ratios at solid walls. This is sometimes necessary for convergence of the initial steady-state solution and without initial stretching the adaptation method would refine only in the direction normal to the wall. Since large jumps in the grid size between adjacent blocks are not allowed and with few blocks normal to the boundary layer, the grid sizes in outer blocks are sometimes bounded by the grid sizes in the boundary layer and not by the error. This waste of cells can be mitigated by introducing more blocks by automatically splitting them or by permitting larger jumps in grid size at the block interfaces.

The flow is entering the computational domain from the left in all the examples. One adapted grid is determined by the algorithm in Section 4 starting from the initial grid with a uniform grid size and its solution. The norm in the algorithm is $\|\tau\|_j = \max(|\tau_1|_\infty, |\tau_2|_\infty, |\tau_3|_\infty)$ computed over all cells in a block j. The maximum norm of the residual in the nonlinear equations measures the progress of the multigrid iterations.

The adapted grids are displayed in the figures such that all blocks look like squares to visualize the grids in the boundary layers. Only every fourth grid line is plotted and therefore no grid lines are shown inside the coarsest blocks. Isolines of the solution are included in the grid plots. The solutions are also plotted in the physical space coordinates.

5.1. Flat plate

Incompressible flow over a flat plate at $x \geqslant 0$ is computed in the first example. The solution at $Re = 5000$ is displayed in Fig. 4. The computational domain Ω in Fig. 4 is divided into 9×4 blocks. The adapted grid and the solution are plotted in Fig. 5 excluding the three leftmost columns of blocks. The singularity at the

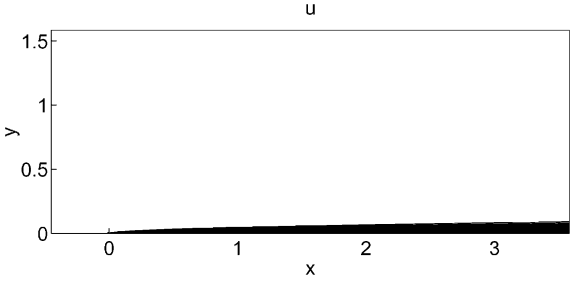

Fig. 4. Contour plots of the velocity component u over a flat plate.

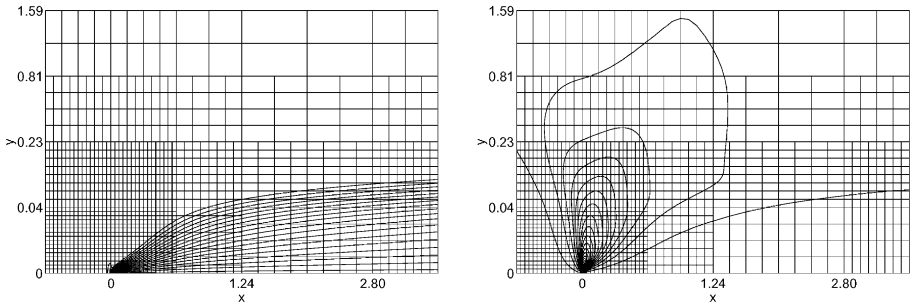

Fig. 5. The adapted grid and isolines of u and v in the transformed domain.

leading edge $x = 0$ of the plate leads to extremely fine grids in the two blocks there if a tolerance ϵ should be satisfied. Since we are not interested in trying to resolve the singularity at the leading edge, the adaptive method is turned off in the two blocks there and they have 64×64 cells independent of the error estimates.

The velocity component u on the adapted grids is extracted at the centers of the four rightmost block columns along lines in the direction normal to the flat plate. This anisotropic solution is compared to the Blasius solution (29) to the left in Fig. 6. The block boundaries in Fig. 6 are at $y \approx 0.04$ and $y \approx 0.23$.

There are 2304 cells in the initial grid and 21,904 cells in the adapted grid in Fig. 5. For comparison an isotropically adapted grid is generated by refining the initial grid in a block as many times in both directions as in the finest direction in the anisotropically adapted grid. The isotropic grid has 30,000 cells. A uniform fine grid with the finest cell size in every block would have 147,456 cells.

The solution is calculated on the uniform grid with 64×64 cells in all blocks. The solutions on the initial grid and the adapted anisotropic and isotropic grids are compared to the uniform grid solution. The differences are plotted in the right panel in Fig. 6 at the same x-stations as the Blasius solution. The error in the initial solution grows as we move from the left to the right over the plate. The accuracy with the adapted grids is much higher than with the initial grid. The refinement was interrupted when $\max_j \|\tau\|_j = 0.0066$. The maximum difference in a component u, v, or p in any cell in the grid was then 0.00159 for the anisotropic grid and 0.00146 for the isotropic grid. Almost the same accuracy is achieved with a grid with fewer cells if it is refined anisotropically.

In Fig. 7, the isolines of the estimated discretization error in the momentum equation in the x-direction τ_2 is shown together with the error δu in u on the adapted grid. The error in u is computed by comparing with

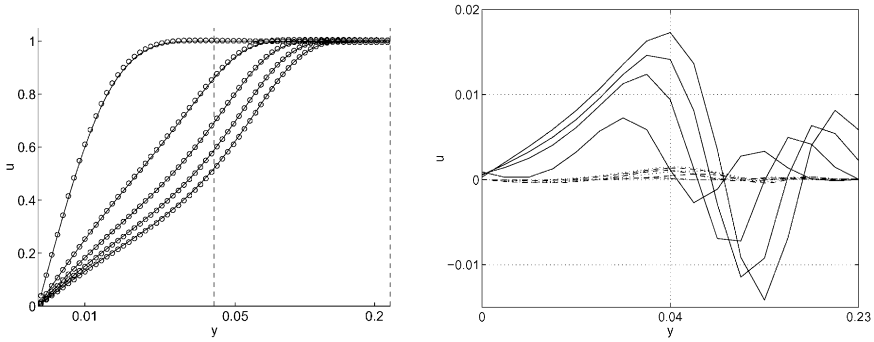

Fig. 6. The solution on the adapted grid (○) is compared to the Blasius solution (solid) (left). Solution errors in u along the plate on the initial grid (solid line), on the adapted anisotropic grid (dotted line), and on the adapted isotropic grid (dashed line) (right).

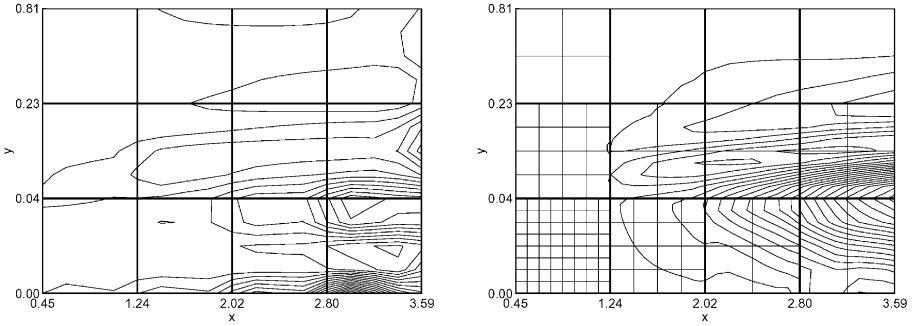

Fig. 7. The estimated discretization error τ_2 (left) and the solution error in u on the adapted grid (right).

a fine grid solution where all blocks have 64×64 cells as in Fig. 6. The relation between δu and τ is given by the error equation (10).

With exact boundary conditions in the numerical approximation, the boundary conditions of the error (10) are homogeneous and $\delta u, \delta v$, and δp are directly proportional to τ. If τ is decreased by a stricter error tolerance, then $\delta u, \delta v$, and δp should be reduced in the same manner. This is also the result for three different ϵ in Fig. 8. The solution error is estimated by comparing with a fine grid solution as in Figs. 6 and 7. How the errors decay with the number of cells N in the blocks in the boundary layer (see Fig. 7) depends on the initial solution. Suppose that the τ^k-term dominates in τ in (36). Then the grid will be refined in the k-direction, $h_k \sim N^{-1}$, and $\tau \sim N^{-2}$. On the contrary, if the refinement is isotropic, then $h_k, h_l \sim N^{-1/2}$ and $\tau \sim N^{-1}$. The inverses of N and N^2 properly scaled by c_1 and c_2 are also plotted in Fig. 8 for comparison. The discretization error and the solution error seem to depend on N like N^{-q} with q between 1 and 2.

A solution is computed on an initial grid. Then the grid is refined in two different ways using the estimated discretization error τ. Either the refinement is isotropic with the same refinement in the x- and y-directions or it is anisotropic as described in Section 4. The decrease of $\max \|\tau\|$ and the increase of the number of cells are recorded in Fig. 9 as a function of the number of iterations in the adaptation algorithm in Section 4. The number of cells grows much faster with isotropic refinement. The number of iterations is larger for anisotropic refinement but the cost of one iteration is low since no new solution is computed in the algorithm.

In Figs. 10 and 11 the solution is first computed on a uniform grid. Then the right-hand side of one of the equations is perturbed by adding a term $\tau \neq 0$ at (0.5, 0.1) outside and at (2, 0.02) inside the boundary layer

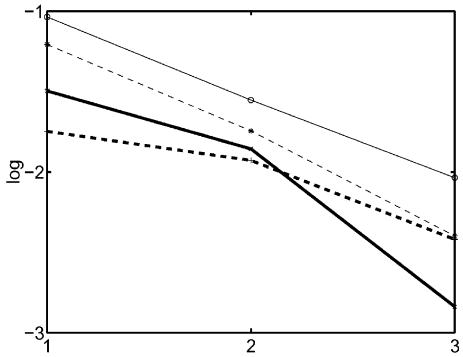

Fig. 8. The logarithms of the maxima of the estimated discretization error (thin solid line, o) and the maximum solution difference in u, v, and p (thin dashed line, $*$) are compared with the logarithms of c_1/N (thick dashed line) and c_2/N^2 (thick solid line).

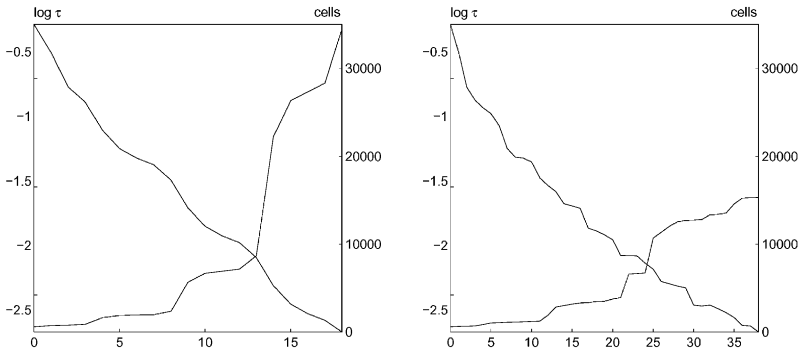

Fig. 9. The reduction of the estimated τ and the increase of the number of cells in a grid refined isotropically (left) and a grid refined anisotropically (right) as the number of iterations in the adaptation algorithm grows.

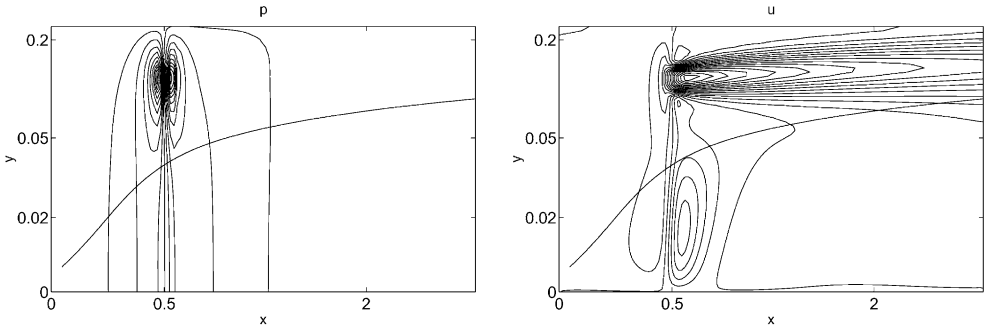

Fig. 10. The changes in p (left) and u (right) after a perturbation has been added to the right-hand side of the divergence equation above the boundary layer.

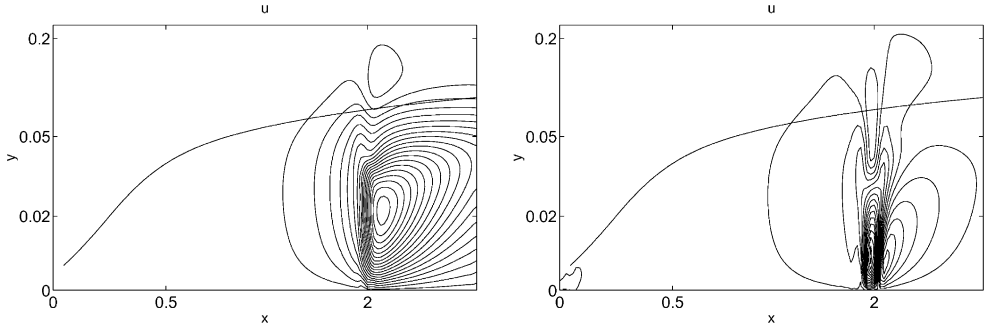

Fig. 11. The change in u after a perturbation has been added to the right-hand side of the x momentum (left) and the y momentum (right) equations in the boundary layer.

simulating a discretization error. These solutions are subtracted from each other and the isolines of the difference is plotted. The ordinate has a logarithmic scale, explaining the elongated level curves in the y-direction. The outer edge of the boundary layer is indicated by a solid line in the figures. As is found for inviscid flow in [9], the difference in the solution outside the boundary layer in Fig. 10 is localized to the

perturbation for p but for u there is an influence far downstream (cf. Proposition 3.2). A grid refinement where δu is large, e.g., at $(x,y) = (2, 0.1)$ will have little effect on δu in contrast to a reduction of τ at $(x,y) = (0.5, 0.1)$. The difference has support closer to the perturbation in the boundary layer in Fig. 11 where the viscous terms dominate.

5.2. Bump

The compressible flow over a bump on a solid wall is computed at Mach number 0.5 and $Re = 5000$. The left, right, and upper boundaries are open, and the data for u in the characteristic variables at the inflow boundary are obtained from a Blasius profile. The adaptive algorithm is turned off there because of disturbances in higher derivatives. Isolines of the velocity component u are shown in Fig. 12. A recirculation zone appears downstream of the bump. The grid and isolines in the transformed domain are plotted in Fig. 13. When the adaptation is interrupted, $\max_j \|\tau\|_j = 0.0216$ in the algorithm. The number of cells of the initial and adapted grids is 3072 and 9856. A corresponding isotropically refined grid has 15,264 cells.

5.3. Cylinder

Incompressible flow around a cylinder is computed in this example. The Reynolds number is 30 and a stationary solution exists. The isolines of the velocity component u and the pressure p are plotted in Fig. 14.

The transformed adapted grid of O-type is plotted in Fig. 15. The lower side of the grid is the surface of the cylinder and the upper side is the outer boundary. The leading edge of the cylinder is at $\theta = \pi$. The aspect ratio is increased by the anisotropic adaptive method in most blocks at the surface of the cylinder.

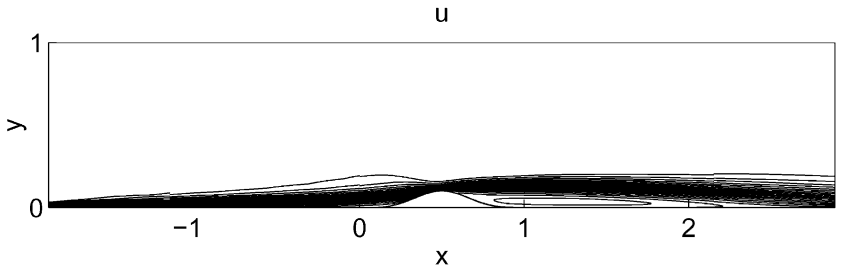

Fig. 12. Isolines of u over a wall with a bump.

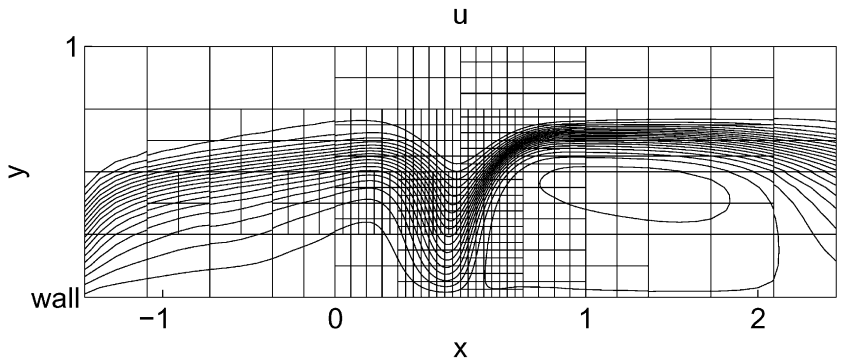

Fig. 13. The adapted grid and isolines of u in the transformed domain over the bump.

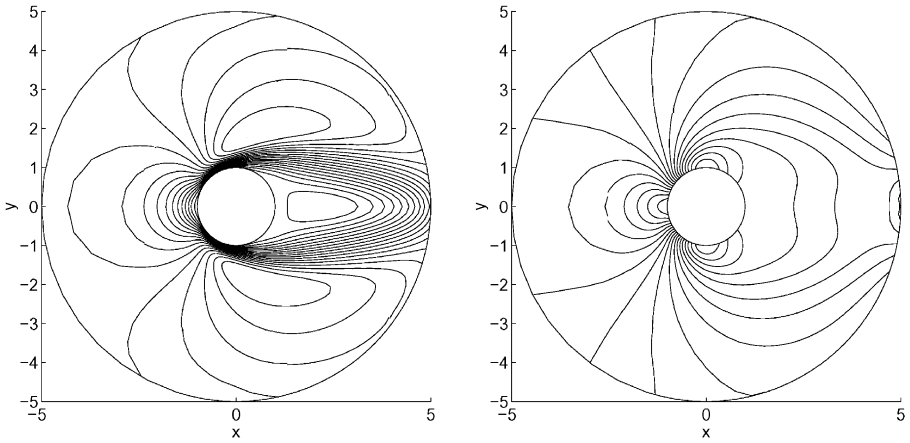

Fig. 14. Isolines of u (left) and p (right) around a cylinder.

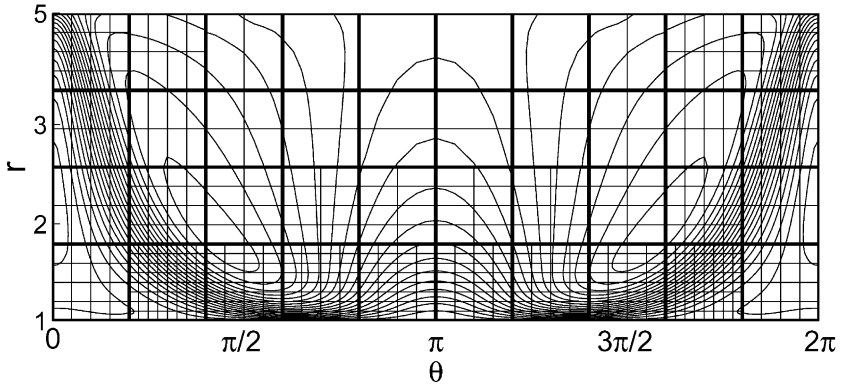

Fig. 15. The grid and isolines of u in the transformed domain around the cylinder.

Tangential refinement is obtained further downstream such that the wake is resolved. Note that the symmetric solution leads to a symmetric grid. We have $\max_j \|\tau\|_j = 0.058$ when the adaptation is terminated. The number of cells are 2560 in the initial grid and 8832 in the adapted grid.

The anisotropic refinement of the grid in different directions is exemplified by computing the flow at $Re = 30$ over a half cylinder. The boundary conditions at $y = 0$ and along the surface of the cylinder in Fig. 16 are the no-slip conditions. The grid is displayed in physical coordinates in the figure. The inner part of the computational domain with the adapted grid obtained with $\max_j \|\tau\|_j = 0.0092$ and isolines of the computed u solution are shown in the figure. The dominant direction of refinement is normal to the walls. The grid is refined isotropically in the leading and trailing corners of the wall. The number of cells in the initial grid is 5120, in the adapted anisotropic grid 24,528, and 54,224 cells with isotropic refinement and the same error estimates.

5.4. Airfoil

In the last example, the subsonic flow around a NACA0012 airfoil is considered at 1.25° angle of attack, Mach number 0.5, and $Re = 3000$. Isolines of u and p are plotted in Fig. 17. The grid is wrapped around the

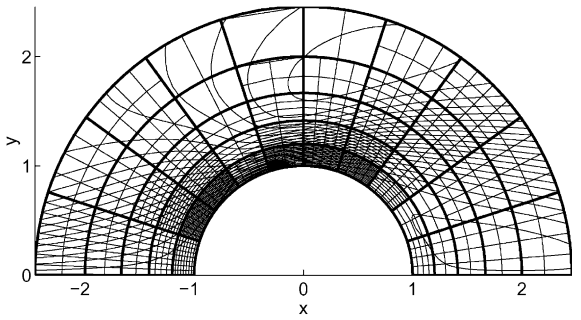

Fig. 16. The isolines of u of the flow over a half cylinder with anisotropic grid refinement in the boundary layer.

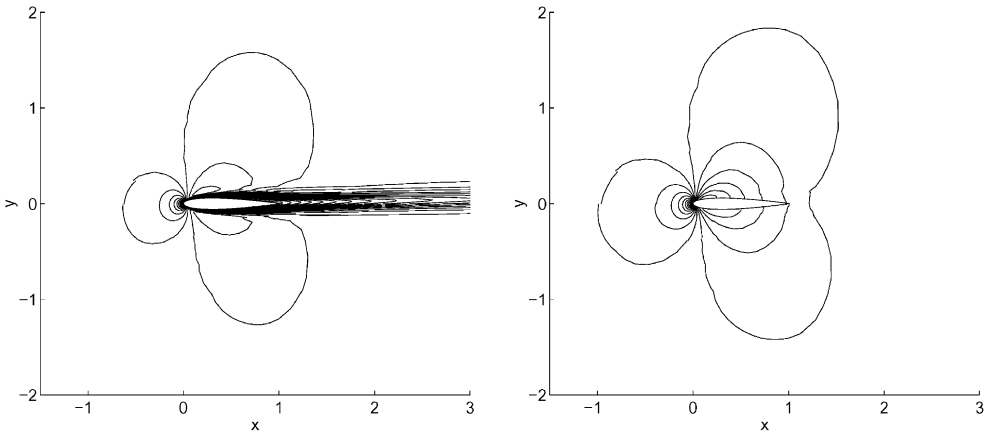

Fig. 17. Isolines of u (left) and p (right) around a wing profile.

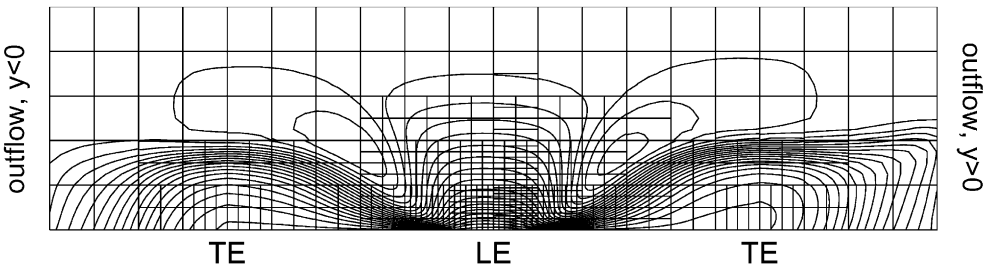

Fig. 18. The adapted grid with isolines of the velocity component u in the transformed domain around the airfoil. The leading and trailing edges of the airfoil are labeled LE and TE.

airfoil as a C-grid so that the solid wall and the wake are at the lower boundary in Fig. 18. The downstream outflow boundary is composed of the left and right sides in Fig. 18. We observe very fine grids at the leading edge of the profile. This is in agreement with the results in [9]. The adaptation is stopped with $\max_j \|\tau\|_j = 0.0234$. There are 1600 cells in the initial grid and 9600 cells in the adapted grid. An isotropic grid has 13888 cells and a uniform grid with the finest grid size in the adapted grid would have 102,400 cells.

6. Conclusions

A method for adaptive anisotropic refinement of structured grids partitioned into blocks has been developed. The method is suitable for problems with rapid variation of the solution in one direction such as boundary layer problems in the Navier–Stokes equations. The error in the solution approximately satisfies a differential equation, the error equation, with the discretization error as the driving right-hand side. The refinement is based on theoretically supported estimates of the discretization error. The numerical experiments show that the accuracy with the anisotropically refined grid is as good as it is with an isotropically refined grid but the number of cells is smaller (Figs. 6 and 9). The grid is refined mainly in the direction normal to the solid walls (Figs. 15 and 16). When the tolerance on the discretization error decreases, then the solution error also decreases (Fig. 8) as expected from the error equation.

Acknowledgements

Financial support has been obtained from the Swedish National Network for Applied Mathematics and the Swedish Research Council for Engineering Science.

References

[1] M. Abramowitz, I.A. Segun, Handbook of Mathematical Functions, Dover, New York, 1968.
[2] R. Becker, M. Braack, R. Rannacher, Adaptive finite element methods for flow problems, in: R. Devore, A. Iserles, E. Süli (Eds.), Foundations of Computational Mathematics, Cambridge University Press, Cambridge, 2001, pp. 21–44.
[3] M. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1989) 64–84.
[4] A.J. Chorin, A numerical method for solving incompressible viscous flow problems, J. Comput. Phys. 2 (1967) 12–36.
[5] P. Colella, M.R. Dorr, D.D. Wake, Numerical solution of plasma fluid equations using locally refined grids, J. Comput. Phys. 152 (1999) 550–583.
[6] D.G. Duffy, Green's Functions with Applications, Chapman & Hall, Boca Raton, 2001.
[7] K. Eriksson, D. Estep, P. Hansbo, C. Johnson, Introduction to adaptive methods for differential equations, Acta Numer. (1995) 105–158.
[8] L. Ferm, P. Lötstedt, Blockwise adaptive grids with multigrid acceleration for compressible flow, AIAA J. 37 (1999) 121–123.
[9] L. Ferm, P. Lötstedt, Adaptive error control for steady state solutions of inviscid flow, SIAM J. Sci. Comput. 23 (2002) 1777–1798.
[10] L. Ferm, P. Lötstedt, Accurate and stable grid interfaces for finite volume methods, Technical Report 2002-012, Department of Information Technology, Uppsala University, Uppsala, Sweden, 2002.
[11] J. Fischer, Self-adaptive mesh refinement for the computation of steady, compressible, viscous flow, Z. Flugwiss. Weltraumforsch. 18 (1994) 241–252.
[12] P. Haldenwang, D. Pignol, Dynamically adapted mesh refinement for combustion front tracking, Comput. Fluids 31 (2002) 589–606.
[13] R. Hagmeijer, Grid adaption based on modified anisotropic diffusion equations formulated in the parametric domain, J. Comput. Phys. 115 (1994) 169–183.
[14] R.D. Hornung, J.A. Trangenstein, Adaptive mesh refinement and multilevel iteration for flow in porous media, J. Comput. Phys. 136 (1997) 522–545.
[15] L.H. Howell, J.B. Bell, An adaptive mesh projection method for viscous incompressible flow, SIAM J. Sci. Comput. 18 (1997) 996–1013.
[16] A. Jameson, W. Schmidt, E. Turkel, Numerical solutions of the Euler equations by finite volume methods using Runge–Kutta time-stepping schemes, AIAA-paper 81-1259, American Institute of Aeronautics and Astronautics, 1981.
[17] A.M. Kuethe, C.-Y. Chow, Foundations of Aerodynamics: Bases of Aerodynamic Design, Wiley, New York, 1976.
[18] P. Lötstedt, S. Söderberg, A. Ramage, L. Hemmingsson-Frändén, Implicit solution of hyperbolic equations with space-time adaptivity, BIT 42 (2002) 134–158.
[19] D.J. Mavriplis, Adaptive meshing techniques for viscous flow calculations on mixed element unstructured meshes, Int. J. Numer. Meth. Fluids 34 (2000) 93–111.

[20] K.W. Morton, Numerical Solution of Convection–Diffusion Problems, Chapman & Hall, London, 1996.
[21] S. Osher, F. Solomon, Upwind schemes for hyperbolic systems of conservation laws, Math. Comp. 38 (1982) 339–377.
[22] K.G. Powell, P.L. Roe, T.J. Linde, T.I. Gombosi, D.L. De Zeeuw, A solution adaptive upwind scheme for ideal magnetohydrodynamics, J. Comput. Phys. 154 (1999) 284–309.
[23] R. Rannacher, Adaptive Galerkin finite element methods for partial differential equations, J. Comput. Appl. Math. 128 (2001) 205–233.
[24] H. Schlichting, Boundary-Layer Theory, seventh ed., McGraw-Hill, New York, 1979.
[25] B. Sjögreen, P. Skogqvist, J. Tegnér, Accuracy in computation of combustible flows, Comp. Fluid Dyn. J. 9 (2001) 535–539.
[26] G. Söderlind, The automatic control of numerical integration, CWI Quarterly 11 (1998) 55–74.
[27] K. Srinivasan, S.G. Rubin, Solution-based grid adaptation through segmented multigrid domain decomposition, J. Comput. Phys. 136 (1997) 467–493.
[28] P. Wesseling, Principles of Computational Fluid Dynamics, Springer, Berlin, 2001.
[29] J. Wu, H. Ritzdorf, K. Oosterlee, B. Steckel, A. Schüller, Adaptive parallel multigrid solution of 2D incompressible Navier–Stokes equations, Int. J. Numer. Meth. Fluids 24 (1997) 875–892.
[30] H. Yu, A local space-time adaptive scheme in solving two-dimensional parabolic problems based on domain decomposition methods, SIAM J. Sci. Comput. 23 (2002) 304–322.