Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 208 (2005) 704–734

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

A shift transformation for fully conservative methods: turbulence simulation on complex, unstructured grids

J.E. Hicken [a,*], F.E. Ham [b], J. Militzer [a], M. Koksal [a]

[a] *Dalhousie University, Department of Mechanical Engineering, J110 "C1" Building, 5269 Morris St., Halifax, Canada NS B3J 1B6*
[b] *Center for Turbulence Research, Stanford University, Stanford, CA 94305, USA*

Received 6 October 2004; received in revised form 16 February 2005; accepted 3 March 2005
Available online 4 May 2005

Abstract

Operator transformations are presented that allow matrix operators for collocated variables to be transformed into matrix operators for staggered variables while preserving symmetries. These "shift" transformations permit conservative, skew-symmetric convective operators and symmetric, positive-definite diffusive operators to be obtained for staggered variables using collocated operators. Shift transformations are not limited to uniform or structured meshes, and this formulation leads to a generalization of the works of Perot (J. Comput. Phys. 159 (2000) 58) and Verstappen and Veldman (J. Comput. Phys. 187 (2003) 343). A set of shift operators have been developed for, and applied to, a time-adaptive Cartesian mesh method with a fractional step algorithm. The resulting numerical scheme conserves mass to machine error and conserves momentum and energy to second order in time. A mass conserving interpolation is used for the variables during mesh adaptation; the interpolation conserves momentum and energy to second order in space. Turbulent channel flow simulations were conducted at $\mathrm{Re}_\tau \approx 180$ using direct numerical simulation (DNS). The DNS results from the adaptive method compare favourably with spectral DNS results despite the use of a (formally) second-order accurate scheme.
© 2005 Elsevier Inc. All rights reserved.

Keywords: Direct numerical simulation; Energy-conserving schemes; Stability; Unstructured grids; Staggered mesh; Grid adaptation

1. Introduction

We are interested in the simulation of turbulent, incompressible flows of Newtonian fluids; hence, we consider the numerical solution of the unsteady, incompressible Navier–Stokes and continuity equations. The non-dimensional form of these equations is

* Corresponding author. Tel.: +1 416 656 0253.
E-mail address: jehicken@alumni.uwaterloo.ca (J.E. Hicken).

0021-9991/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2005.03.002

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_j u_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{1}{Re}\frac{\partial^2 u_i}{\partial x_j \partial x_j}, \tag{1}$$

$$\frac{\partial u_j}{\partial x_j} = 0. \tag{2}$$

For the purposes of this work, the density is assumed constant and is absorbed in the pressure. The existence and uniqueness of general solutions to (1) and (2) remains an open question; nevertheless, solutions of the Navier–Stokes and continuity equations have been verified experimentally for several laminar and unstable flows. Moreover, highly accurate numerical simulations, performed by Kim et al. [12] and others [17,22], suggest these equations are an excellent model for incompressible turbulence.

Early work on the numerical simulation of turbulence relied on methods using orthogonal functions, particularly Fourier series, e.g. [12]. These methods, although accurate and fast, are restricted to simple geometries; thus, during the last decade, many researchers have worked on taking direct numerical simulations (DNS) and large-eddy simulations (LES) beyond spectral methods. Specifically, this paper is concerned with the continued development of finite difference/volume methods for turbulence simulation.

Applying finite difference schemes to the simulation of turbulence presents a challenge: reconciling accuracy and stability. Upwind schemes are generally stable, because they introduce numerical dissipation. Numerical dissipation, however, upsets the subtle balance of forces at the smallest scales in the turbulent flow and introduces errors [15,16]. Conversely, while central difference schemes do not add non-physical damping, these schemes are often not stable.

One solution to the accuracy-stability problem is to ensure kinetic energy is conserved inviscidly by the convective terms. Morinishi et al. [16] reviewed existing conservative, second-order finite difference schemes for structured meshes, and introduced a forth-order conservative discretization. The truncation error for their strictly conservative scheme is not fourth order on a nonuniform mesh, so they developed a "nearly conservative" fourth-order scheme which sacrificed conservation for accuracy. Verstappen and Veldman [22] note that the truncation error is not necessarily the same order as the actual error, and they have developed a fully conservative scheme for structured Cartesian meshes which is fourth-order despite a lower-order truncation error.

Perot [18] derived a conservative staggered mesh scheme for unstructured grids, and thus opened the way for DNS and LES on even more complicated domains (see also [19]). More recently, Mahesh et al. [13] have developed staggered and collocated schemes for complex domains. Their work also suggests that fully conservative methods are necessary for accurate LES simulations. Specifically, they argue that strict conservation is required if the dissipative scales of turbulence are not resolved, since the diffusive terms will not remove sufficient energy at high Reynolds numbers.

The conservative schemes of [18] and [22] fall into the more general field of mimetic finite difference methods pioneered by Hyman et al. [7] and Hyman and Shashkov, see [8–10], for example. The extensive work of Hyman and Shashkov has focused on logically rectangular grids, although the support-operator method they employ can be applied to more general meshes.

The primary aim of this work has been to generalize the methods of Perot [18] and Verstappen and Veldman [22]. By considering conservation from an operator perspective, Verstappen and Veldman demonstrated that the convective operator must be skew-symmetric to guarantee conservation of kinetic energy. This remark is our starting point for developing a more general, conservative discretization of the Navier–Stokes and continuity equations.

A secondary goal of this work, and a logical step after considering unstructured meshes, was the development of a method suitable for turbulence simulation on adaptive meshes. The adaptive mesh method presented here builds on the Cartesian grid method of Ham et al. [4] which uses local anisotropic adaptation.

The paper proceeds as follows. Section 2 relates conservation of energy with differential operator symmetries. The corresponding discrete operators and their symmetries are then discussed. The shift transformation is introduced so staggered variables can take advantage of the symmetries of certain collocated operators. The shift transformation is specialized to an unstructured Cartesian mesh in Section 3. The Cartesian method uses anisotropic adaptation, and aspects of the adaptive mesh are presented in Section 4. Analysis and numerical experiments are used in Section 5 to investigate the accuracy of the method. Section 6 presents the results of a DNS of the turbulent channel obtained with present method. The summary and conclusions can be found in Section 7.

2. Symmetry-preserving discretization

2.1. The governing equations: Skew-symmetry and conservation

The conservative nature of the Navier–Stokes and continuity equations are defining characteristics of these differential equations which allows them to represent physical flows so well. Together with mass and momentum conservation, the incompressible form of these equations also conserves kinetic energy in the absence of viscosity. Energy conservation is not a separate constraint, but rather a consequence of Eqs. (1) and (2).

Consider the equation governing the global kinetic equation. Let $\langle , \rangle$ denote the integral scalar product of two functions on an open and bounded domain Ω. For example, the scalar product of two vectors, $\mathbf{u}$ and $\mathbf{v}$, is given by

$$\langle \mathbf{u}, \mathbf{v} \rangle = \int\int_{\Omega}\int \mathbf{u} \cdot \mathbf{v} \, \mathrm{d}V.$$

The evolution equation for the global kinetic energy on Ω can be obtained by differentiating $\langle \mathbf{u}, \mathbf{u} \rangle$ with respect to time and using the vector form of (1) [22]

$$\frac{\mathrm{d}}{\mathrm{d}t}\langle \mathbf{u}, \mathbf{u} \rangle = -\langle (\mathbf{u} \cdot \nabla)\mathbf{u}, \mathbf{u} \rangle - \langle \mathbf{u}, (\mathbf{u} \cdot \nabla)\mathbf{u} \rangle + \frac{1}{Re}(\langle \nabla \cdot \nabla \mathbf{u}, \mathbf{u} \rangle + \langle \mathbf{u}, \nabla \cdot \nabla \mathbf{u} \rangle) - \langle \nabla p, \mathbf{u} \rangle - \langle \mathbf{u}, \nabla p \rangle. \tag{3}$$

The differential equation (3) contains several terms involving the convective and gradient operators, $(\mathbf{u} \cdot \nabla)$ and ∇, respectively. Assuming boundary terms vanish, or Ω is periodic, integration by parts reveals that these operators satisfy [22]

$$\langle \nabla p, \mathbf{u} \rangle = -\langle p, \nabla \cdot \mathbf{u} \rangle \tag{4}$$

and

$$\langle (\mathbf{u} \cdot \nabla)\mathbf{v}, \mathbf{w} \rangle = -\langle \mathbf{v}, (\mathbf{u} \cdot \nabla)\mathbf{w} \rangle. \tag{5}$$

The first result, Eq. (4), simply states that the adjoint of the divergence operator is the negative gradient operator. Eq. (5) is an analogous relation between the convective operator and its adjoint. It follows that the (differential) convective operator is skew-symmetric.

Returning to the kinetic energy equation (3) the importance of these properties becomes clear; they eliminate the convective and pressure terms from the equation governing the evolution of global energy. Hence, the energy equation reduces to

$$\frac{\partial}{\partial t}\langle \mathbf{u}, \mathbf{u} \rangle = -\frac{2}{Re}\langle \nabla \mathbf{u}, \nabla \mathbf{u} \rangle. \tag{6}$$

The skew-symmetries of the convective and divergence operators guarantee global energy conservation in the absence of viscosity. Moreover, the kinetic energy is monotone decreasing with time when viscosity is present. The relations (4) and (5) are extremely important and will be revisited when constructing the appropriate discretization for the continuity and Navier–Stokes equations.

2.2. Energy conservation and discrete operators

On an arbitrary mesh, the finite volume spatial discretizations of the Navier–Stokes and continuity equations are given by, respectively,

$$\mathbf{\Omega}\frac{d\mathbf{u}_s}{dt} + \mathscr{C}(u)\mathbf{u}_s + \mathscr{D}\mathbf{u}_s + \mathbf{\Omega}\mathscr{G}\mathbf{p}_c = \mathbf{0}_s \tag{7}$$

and

$$\mathscr{M}\mathbf{u}_s = \mathbf{0}_c. \tag{8}$$

The variable $\mathbf{u}_s$ is an m-column vector of velocities and $\mathbf{p}_c$ is an n-column vector of pressures. The matrix $\mathbf{\Omega}$ is a diagonal $m \times m$ matrix of velocity cell control volumes (CVs). The $m \times m$ matrices $\mathscr{C}(u)$ and $\mathscr{D}$ represent the convective and diffusive operators, respectively; the u dependence of the convective operator is a reminder that this operator is non-linear when applied to $\mathbf{u}_s$. Finally, the $m \times n$ matrix $\mathscr{G}$ denotes the gradient operator and the $n \times m$ matrix $\mathscr{M}$ is the divergence operator.

A finite volume discretization ensures mass and momentum are conserved; however, unlike the differential equations, energy conservation is not guaranteed by (7) and (8). Following [22], the global discrete energy is defined as

$$\|\mathbf{u}_s\|^2 \equiv \mathbf{u}_s^*\mathbf{\Omega}\mathbf{u}_s. \tag{9}$$

An equation for the evolution of $\|\mathbf{u}_s\|^2$ can be obtained by left-multiplying Eq. (7) by $\mathbf{u}_s^*$ and summing the resulting equation with its conjugate transpose:

$$\frac{\mathrm{d}}{\mathrm{d}t}\|\mathbf{u}_s\|^2 = -\mathbf{u}_s^*(\mathscr{C}(u) + \mathscr{C}^*(u))\mathbf{u}_s - \mathbf{u}_s^*(\mathscr{D}(u) + \mathscr{D}^*(u))\mathbf{u}_s - \mathbf{u}_s^*\mathbf{\Omega}\mathscr{G}\mathbf{p}_c - \mathbf{p}_c^*\mathscr{G}^*\mathbf{\Omega}^*\mathbf{u}_s. \tag{10}$$

The convective and pressure terms vanish in the differential form of the total energy equation because of their respective operator symmetries. If the analogous terms are to cancel in the discrete energy equation the following two conditions must be met:

$$\mathbf{u}_s^*(\mathscr{C}(u) + \mathscr{C}^*(u))\mathbf{u}_s = 0, \tag{11}$$

$$\mathbf{u}_s^*\mathbf{\Omega}\mathscr{G}\mathbf{p}_c + \mathbf{p}_c^*\mathscr{G}^*\mathbf{\Omega}^*\mathbf{u}_s = 0. \tag{12}$$

The first equation is satisfied, for any velocity, if and only if the discrete convective operator is skew-symmetric. Eq. (12) is satisfied if the negative conjugate transpose of the discrete gradient operator is equal to the divergence operator. Hence,

$$\mathscr{C}(u) = -\mathscr{C}^*(u) \tag{13}$$

and

$$-(\mathbf{\Omega}\mathscr{G})^* = \mathscr{M}. \tag{14}$$

Note, (14) will guarantee (12) only if the velocity field is divergence free ($\mathscr{M}\mathbf{u}_s = 0$).

The skew-symmetries of the differential operators must be retained when discretizing the equations of motion if energy, as defined by (9), is to be conserved inviscidly.[1] The following subsection will examine

[1] Eqs. (13) and (14) are consequences of the discrete energy definition (9). In general, the energy definition will not produce these conditions. We would like to thank one of the referees for bringing this to our attention.

how the symmetries (13) and (14) constrain the choice of variable arrangement and operator discretization.

2.3. Constructing discrete operators

For the sake of generality, assume that the underlying mesh is unstructured and the solution domain is divided into a finite number, n, of non-overlapping volumes. Each volume may have a different number of faces with area A_f and normal $\mathbf{n}_f$. Suppose there are m faces on the computational domain.

The discrete variables can be arranged and defined in a variety of ways. The two most common schemes for the incompressible Navier–Stokes equations are the staggered and collocated arrangements. Fig. 1 illustrates the two schemes. The staggered mesh scheme, first proposed by Harlow and Welch [6], stores the pressure at the cell-centres and a face-normal velocity at the faces. In the collocated arrangement, the pressure and velocities are stored at the cell-centre, and a secondary velocity field is stored at the faces; the collocated velocities are used to satisfy the momentum equation while the face-velocities are used to enforce mass conservation. The two velocity fields in the collocated arrangement are coupled using an interpolation method developed by Rhie and Chow [20].

Defining a skew-symmetric convective operator on a collocated mesh is straightforward. Unfortunately, the gradient operator $\mathscr{G}$ used by collocated schemes is not related to the divergence operator via equation (14). If the two operators were related, as required by energy conservation, the pressure Poisson equation used to enforce mass conservation would decouple the pressure field into even and odd modes; the result is the well-known and non-physical checkerboard pressure field [2]. Most collocated methods use a non-conservative pressure gradient term to avoid the decoupling problem.

In contrast, the gradient and divergence operators on a staggered mesh can be related via equation (14) without decoupling the pressure field; however, a conservative discretization of the convective operator is difficult to define on unstructured, staggered meshes. This difficulty will be addressed later, when we will

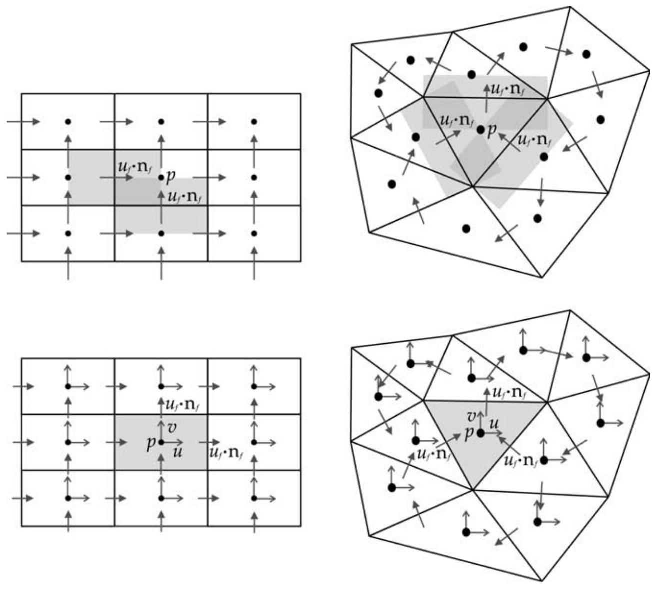

Fig. 1. Variable arrangement for the staggered and collocated schemes. The top two meshes show staggered schemes for a structured mesh (left) and an unstructured, triangular mesh (right). The lower two meshes show the collocated arrangement for the same geometries.

show that collocated convective operators can be used to construct suitable staggered convective operators. For the remainder of this paper, the variables will be assumed to be staggered.

2.3.1. Gradient and divergence operators

Eq. (14) equates the integrated pressure gradient operator, $\mathbf{\Omega}\mathscr{G}$, to the negative conjugate transpose of $\mathscr{M}$, the divergence operator. Hence, specifying one operator determines the other. Since an obvious discretization of the continuity equation is available on staggered meshes, $\mathscr{M}$ is defined first and the gradient operator is allowed to follow. The idea of using one operator to define another is generalized by the support-operators method discussed in [21], for example.

Integrating the continuity equation (2) over an arbitrary pressure cell k of volume ΔV_k and applying the divergence theorem yields

$$\iint_{\partial \Delta V_k} u_j n_j \, \mathrm{d}S = \sum_{f \in F(k)} \iint_{\Delta A_f} u_j n_j \, \mathrm{d}S = 0, \tag{15}$$

where $F(k)$ is the set of all faces bordering the pressure-cell k and ΔA_f is the set of points on face f. If the discrete face normal velocities, $U_f = \mathbf{u}_f \cdot \mathbf{n}_f$, are located at the centroid of the face then a suitable second-order discretization of (15) is given by

$$[\mathscr{M}\mathbf{u}_s]_k = \sum_{f \in F(k)} U_f A_f = 0. \tag{16}$$

where A_f is the area of face f. As usual, the face normal vector $\mathbf{n}_f$ points out of the control volume k.

Since $-\mathscr{M}^* = \mathbf{\Omega}\mathscr{G}$ the following pressure gradient discretization at face f follows immediately from (16):

$$[\mathbf{\Omega}\mathscr{G}\mathbf{p}_c]_f = (p_{c2} - p_{c1})A_f, \tag{17}$$

The cells $c1$ and $c2$ in (17) are adjacent to face f, but their order depends on the nature of the mesh. In general, the accuracy of the discretization (17) is not second-order; the accuracy will be discussed further in Section 5.

2.3.2. Skew-symmetric convective operator

Let $\mathscr{C}_c(u)$ be an $n \times n$ skew-symmetric convective operator on a collocated mesh. The operator $\mathscr{C}_c(u)$ is defined by its action on an arbitrary pressure-centred variable $\boldsymbol{\phi}_c$ at some cell k:

$$[\mathscr{C}_c(u)\boldsymbol{\phi}_c]_k = \sum_{f \in F(k)} \frac{(\phi_{c1} + \phi_{c2})}{2} A_f U_f. \tag{18}$$

The variable values in the cells adjacent to the face f are denoted by ϕ_{c1} and ϕ_{c2} with the convention that the normal vector points from cell $c1$ to cell $c2$, so $c1$ is cell k for each face in this case. Fig. 2 illustrates the face normal convention.

The collocated operator $\mathscr{C}_c(u)$ is skew-symmetric provided the face-velocities are mass conserving. To see this, note that the off-diagonal elements of $\mathscr{C}_c(u)$ are given by

$$[\mathscr{C}_c(u)]_{kj} = \frac{1}{2} A_{kj} U_{kj} = -\frac{1}{2} A_{jk} U_{jk} = -[\mathscr{C}_c(u)]_{jk} \quad (\text{no sum } j, k),$$

where $A_{kj} = A_{jk}$ is the area of the face common to cells k and j and $U_{kj} = -U_{jk}$ is the face normal velocity from cell k to cell j. The diagonal term is simply the sum of the off-diagonal elements of row k:

$$[\mathscr{C}_c(u)]_{kk} = \sum_{f \in F(k)} \frac{1}{2} A_f U_f = 0 \quad (\text{no sum } k), \tag{19}$$

by virtue of the continuity condition, Eq. (16).

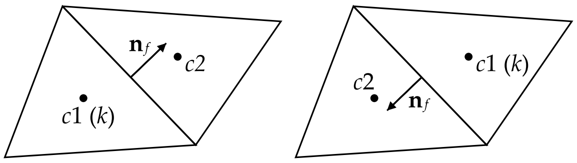

Fig. 2. Face normal and neighbour convention. For cell k the face normal points from cell $c1 = k$ to $c2$, regardless of which cell k is identified with. In other words, k represents a permanent labelling of the cells while $c1$ and $c2$ can be thought of as temporary.

How does a collocated convective operator help us construct a staggered operator? Suppose the staggered velocities could be averaged, or shifted, to the cell-centres. The collocated operator $\mathscr{C}_c(u)$ could then be applied to these shifted velocities. Subsequently, the convective flux resulting from the application of $\mathscr{C}_c(u)$ could be returned to the faces. Therefore, to make use of the collocated operator, two additional operators are needed: one to move face variables to the cells, and one to move cell variables to the faces.

The operators for moving variables between face and cell are constrained by the skew-symmetry condition placed on $\mathscr{C}(u)$. Let $\mathbf{\Upsilon}$ and $\mathbf{\Pi}$ denote the $3n \times m$ face-to-cell and $m \times 3n$ cell-to-face operators, respectively. The $3n$ rows of the face-to-cell operator and the $3n$ columns of the cell-to-face operator are needed because each collocated cell needs three Cartesian velocity components. For simplicity, assume that the action of $\mathbf{\Upsilon}$ on $\mathbf{u}_s$ produces a $3n$-vector of velocities such that all the x-component velocities are first, followed by all of the y-component velocities, and finally all of the z-component velocities. Hence, a staggered convective operator will be given by

$$\mathscr{C}(u) = \mathbf{\Pi}\mathscr{C}_u(u)\mathbf{\Upsilon},$$

where the $3n \times 3n$ matrix $\mathscr{C}_u(u)$ is given by

$$\mathscr{C}_u(u) = \begin{pmatrix} \mathscr{C}_c(u) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathscr{C}_c(u) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathscr{C}_c(u) \end{pmatrix}.$$

Conservation of energy demands that

$$-\mathscr{C}^*(u) = -\mathbf{\Upsilon}^*\mathscr{C}_u(u^*)\mathbf{\Pi}^* = \mathbf{\Upsilon}^*\mathscr{C}_u(u)\mathbf{\Pi}^* = \mathbf{\Pi}\mathscr{C}_u(u)\mathbf{\Upsilon} = \mathscr{C}(u). \tag{20}$$

An obvious solution to the constraint (20) would be to choose $\mathbf{\Pi}^* = \mathbf{\Upsilon} = \mathbf{\Gamma}$ giving $\mathscr{C}(u) = \mathbf{\Gamma}^*\mathscr{C}_u(u)\mathbf{\Gamma}$. In general, a skew-symmetric operator can be constructed using any number of shift operators. Thus, a more general convective operator can be defined as

$$\mathscr{C}(u) = \frac{1}{N}\sum_{i=1}^{N}\mathbf{\Gamma}_i^*\mathscr{C}_u(u)\mathbf{\Gamma}_i. \tag{21}$$

The shift operators $\mathbf{\Gamma}_i$ are further constrained by global momentum conservation. An equation for the evolution of total momentum can be obtained by contracting the discrete momentum equation, Eq. (7), with the constant m-vector $\mathbf{1}_s$:

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{1}_s^*\mathbf{\Omega}\mathbf{u}_s) = -\mathbf{1}_s^*(\mathscr{C}(u) + \mathscr{D})\mathbf{u}_s - \mathbf{1}_s^*\mathbf{\Omega}\mathscr{G}\mathbf{p}_c = 0_s. \tag{22}$$

Momentum will be conserved if $(\mathscr{C}(u) + \mathscr{D})^*\mathbf{1}_s = \mathbf{0}_s$ and $-\mathscr{G}^*\mathbf{\Omega}\mathbf{1}_s = \mathscr{M}\mathbf{1}_s = \mathbf{0}_c$. The latter condition is guaranteed if mass is conserved by a constant velocity field, i.e. conservation of mass is consistently discretized between cells [22]. The former condition can be separated into the following two conditions:

$\mathscr{C}(u)^*\mathbf{1}_s = \mathbf{0}_s$ and $\mathscr{D}^*\mathbf{1}_s = \mathbf{0}_s$ [22]. The momentum conservation of the diffusive operator will be addressed later. The condition for the convective operator can be recast as $\mathscr{C}(u)\mathbf{1}_s = \mathbf{0}_s$ using the skew-symmetry of $\mathscr{C}(u)$. Now, the constant n-vector $\mathbf{1}_c$ lies in the null space of the collocated convective operator, since the row sum of $\mathscr{C}_c(u)$ is equal to two times the diagonal element; see Eq. (19). If the face-to-cell operators maintain the value of a constant it will follow that the staggered convective operator will also conserve momentum. That is, if

$$\mathbf{\Gamma}\mathbf{1}_s = \mathbf{1}_{3c},$$

where $\mathbf{1}_{3c}$ is a $3n$-vector, then

$$\mathscr{C}(u)\mathbf{1}_s = \mathbf{\Gamma}^*\mathscr{C}_u(u)\mathbf{\Gamma}\mathbf{1}_s = \mathbf{\Gamma}^*\mathscr{C}_u(u)\mathbf{1}_{3c} = \mathbf{0}_s.$$

The preceding analysis has shown that the face-to-cell and cell-to-face shift operators are related; furthermore, momentum is conserved provided the face-to-cell operators preserve the value of a constant. Constraining the shift operators further is difficult without specializing to a particular type of unstructured mesh and desired order of accuracy. Several suitable $\mathbf{\Gamma}_i$, including the ones adopted for the current work, are presented in Section 3.

2.3.3. *A positive-definite diffusive operator*

With the appropriate choice of convective and gradient operators, Eq. (10) reduces to

$$\frac{\mathrm{d}}{\mathrm{d}t}\|\mathbf{u}_s\|^2 = -\mathbf{u}_s^*(\mathscr{D}(u) + \mathscr{D}^*(u))\mathbf{u}_s. \tag{23}$$

To mimic the analogous differential result, Eq. (6), the diffusive terms must be dissipative:

$$\mathbf{u}_s^*(\mathscr{D}(u) + \mathscr{D}^*(u))\mathbf{u}_s \geqslant 0, \quad \forall \mathbf{u}_s.$$

This inequality will be satisfied if $\mathscr{D}$ is symmetric and positive semi-definite. Respecting Eq. (23) may be as important as the inviscid conservation of energy, but constructing a symmetric, positive-definite diffusive operator to ensure the monotonicity of $\|\mathbf{u}_s\|^2$ has a large influence on the accuracy of this operator. On unstructured meshes the accuracy of $\mathscr{D}$ may need to take precedence over the properties of the operator. In this work, however, we have adopted the philosophy of Verstappen and Veldman [22]; namely, that the symmetries of the convective and diffusive operators are critical to the dynamics of turbulence and should be respected.

Again, as with the convective operator, a diffusive operator is easily constructed on a collocated mesh. The diffusive flux is the divergence, $\mathscr{M}$, of a gradient, $\mathscr{G}$, which suggests the finite volume discretization

$$\mathscr{D}_c = -\frac{1}{Re}\mathscr{M}\mathscr{G} = \frac{1}{Re}\mathscr{M}\mathbf{\Omega}^{-1}\mathscr{M}^*, \tag{24}$$

where the Reynolds number has been introduced. The negative sign in (24) arises because of the form of Eq. (7) where all the operators are on the left-hand side and positive. The collocated diffusive operator (24) is clearly symmetric and positive-definite. The action of the operator on the variable $\boldsymbol{\phi}_c$ at cell k is

$$[\mathscr{D}_c\boldsymbol{\phi}_c]_k = \frac{1}{Re}\sum_{f\in F(k)}\frac{(\phi_{c2} - \phi_{c1})A_f}{\delta n_f}, \tag{25}$$

where $\delta n_f = \Delta V_f / A_f$ and $\Delta V_f = [\mathbf{\Omega}]_{ff}$ is the volume of the velocity cell CV. The length δn_f is an approximation of the distance between the centroids of cell $c1$ and $c2$. On general unstructured meshes, the discretization (25) contains an error term which is O(1), so this operator is unsuitable for many meshes. For the adaptive Cartesian mesh used in the present study, these low order terms are produced in symmetric pairs which appear to cancel. This will be discussed further in Section 5.

Of course, a collocated diffusive operator cannot be directly applied to a staggered mesh. Appropriate face-to-cell and cell-to-face operators are needed to make use of $\mathscr{D}_c$. Fortunately, the shift operators used for the convective matrix can be reused for the diffusive operator. The general expression for a staggered diffusive operator is

$$\mathscr{D} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\Gamma}_i^* \mathscr{D}_u \boldsymbol{\Gamma}_i, \tag{26}$$

where $\mathscr{D}_u$ is a $3n \times 3n$ matrix defined by

$$\mathscr{D}_u = \begin{pmatrix} \mathscr{D}_c & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathscr{D}_c & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathscr{D}_c \end{pmatrix}.$$

The symmetry and positive-definiteness of the collocated diffusive operator are preserved, because the face-to-cell and cell-to-face operators are the conjugate transpose of each other. Thus, the operator $\mathscr{D}$ will ensure that the total kinetic energy is monotone decreasing with time.

The momentum conservation properties of $\mathscr{D}$ also need to be addressed. Recall that the diffusive operator will conserve momentum if and only if $\mathscr{D}^*\mathbf{1}_s = \mathbf{0}_s$. This requirement can be rewritten as

$$\mathscr{D}\mathbf{1}_s = \mathbf{0}_s, \tag{27}$$

since the diffusive operator is symmetric. To ensure the convective terms conserve momentum, the shift operators have been constructed such that

$$\boldsymbol{\Gamma}_i \mathbf{1}_s = \mathbf{1}_{3c},$$

so if the collocated diffusive operator satisfies $\mathscr{D}_c\mathbf{1}_c = \mathbf{0}_c$ then Eq. (27) will follow. But the constant vector $\mathbf{1}_c$ is clearly in the null space of $\mathscr{D}_c$ by construction; see Eq. (25). Thus, the staggered operator $\mathscr{D}$ conserves momentum as required.

3. Specializing to a cartesian grid method

3.1. Unstructured cartesian mesh with anisotropic adaptation

The method presented here was built directly upon the Cartesian method of Ham et al. [4]. Their method uses anisotropic refinement which provides some control over the local truncation error. Although the original scheme uses collocated variables, the conversion to staggered variables is straightforward. The simplicity with which unstructured collocated solvers can be converted to conservative staggered solvers is a favourable property of the present numerical method.

The shift operators depend on the geometry of the CVs, so a brief review of the mesh used in [4] is necessary before the operators can be defined. The mesh is time-adaptive, so the resulting geometry can be explained by "walking-through" an example with a few cell refinements. For simplicity, the example is restricted to two dimensions; the extension to three dimensions is straightforward. Referring to Fig. 3, consider several refinements of the rectangular domain with dimensions of L_x and L_y. The initial domain is covered by a single coarse cell: Fig. 3(a). Fig. 3(b) shows the domain after one anisotropic refinement in the y-direction, and Fig. 3(c) is the mesh after several refinements. Each cell refinement is anisotropic and involves the bisection of the cell in either the x- or y- (or z-) directions. An isotropic refinement is achieved by two (or three) adaptations in each of the coordinate directions. Each cell is defined uniquely by its indices (i,j) and refinement levels $[l_i, l_j]$. For a particular cell, the levels keep track of the number

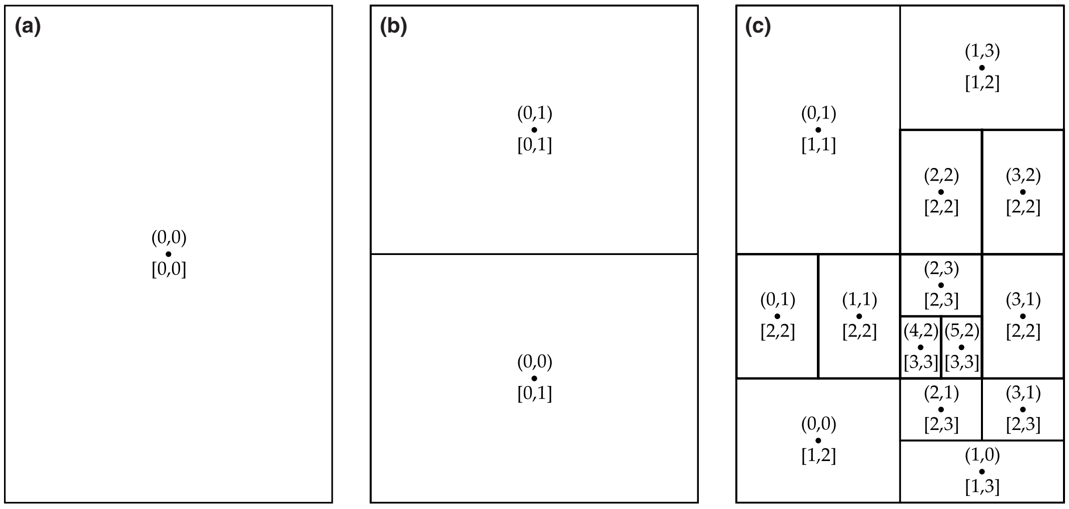

Fig. 3. Cartesian adaptive mesh example.

of refinements necessary to get from the global dimensions to the cell's dimensions. The indices represent the structured Cartesian indices the cell would have if all cells were refined to its level.

One restriction on the refinement and coarsening algorithms is the number of neighbours permitted in each direction: four neighbours in three dimensions, two in two dimensions. More precisely, for an arbitrary cell C, each neighbour in the x_i-direction cannot have an x_j- or x_k-refinement level which differs from C's corresponding level by more than 1, for $i \neq j \neq k$. Note that the x_i-refinement level of cell C does not restrict the x_i-refinement level of its neighbours in the x_i-direction. For example, if cell C has a y-refinement level of 5 then its east, west, top, and bottom neighbours may have y-refinement levels of 4, 5, or 6, but the north and south neighbours have no such restriction. For further details on the implementation of the refinement and coarsening algorithms the reader is directed to the original paper [4].

3.2. Shift operators for unstructured Cartesian meshes

Fig. 4 shows the various geometric dimensions of the unstructured Cartesian staggered mesh. Consider the velocity cell enclosing face f in this figure. If the normal vector for the velocity cell f is $n_{i,f} = 1$, then $\delta n_f = x_{i,c2} - x_{i,c1}$ and the velocity-cell's area is

$$A_f = \min(\Delta x_{j,c1}, \Delta x_{j,c2}) \min(\Delta x_{k,c1}, \Delta x_{k,c2}),$$

where $i \neq j \neq k$.

Consider only local face-to-cell shifts involving immediate faces of a cell. Even with this restriction, several possible shift operators are available. One of the more obvious ways to get the velocities from the face to the cell is to use a face-area weighted average. Such a shift operator is defined by

$$[\mathbf{\Gamma u}_s]_{ik} = \frac{\sum_{f \in F(k)} U_f n_{i,f} A_f}{2 \sum_{f \in F(k)} A_f \mid n_{i,f} \mid} = \left(\frac{1}{2\Delta V}\right)_k \sum_{f \in F(k)} U_f n_{i,f} A_f (\Delta x_i)_k, \quad i = 1, 2, 3, \tag{28}$$

where $n_{i,f}$ is the ith component of the outward face normal. Recall that $U_f = \mathbf{u}_f \cdot \mathbf{n}_f$, so $U_f n_{i,f}$ defines the ith Cartesian component of the face velocity. The shift operator (28) is similar to the interpolation used by Perot [18].

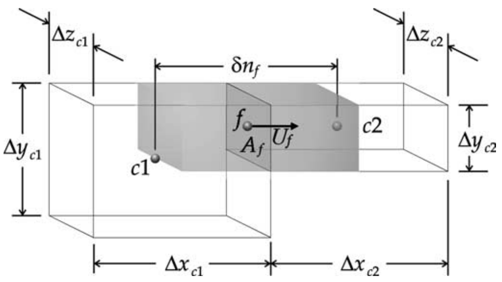

Fig. 4. Cartesian mesh geometry definitions.

For Cartesian meshes the shift operator (28) has an undesirable property. Consider the shift operator applied to the diffusive operator (26) on a structured Cartesian mesh. For a generic x-velocity cell P, in two dimensions, the diffusive flux discretization reduces to

$$[\mathscr{D}\mathbf{u}_s]_P = \frac{A}{4\Delta x}(U_{EE} - 2U_P + U_{WW}) + \frac{A}{\Delta y}\left[\frac{1}{2}(U_N - 2U_P + U_S) + \frac{1}{4}(U_{NE} - 2U_E + U_{SE}) + \frac{1}{4}(U_{NW} - 2U_W + U_{SW})\right]. \tag{29}$$

The averaging has decoupled adjacent velocity nodes in the x-direction and introduced an unnecessary average in the y-direction. A similar cancellation results when the interpolation (28) is applied to the convective operator.

Structured Cartesian meshes are a special case of the unstructured Cartesian mesh adopted for this work. Hence, to avoid the problem presented by the interpolation (28), the numerical method should reduce to the standard Harlow and Welsh scheme [6] when the mesh, either locally or globally, refines to a structured grid. If the unstructured method is to reduce to the structured one, the convective and diffusive operators must use only local neighbours in their stencils.

Eqs. (21) and (26) do not restrict the number of shift operators applied to the collocated convective and diffusive operators; hence, the method developed for this work uses two shift operators, defined as follows:

$$(\mathbf{\Gamma}_1\mathbf{u}_s)_{ik} = \frac{\sum_{f\in F(k)} U_f \max(n_{i,f},0)A_f}{\sum_{f\in F(k)} A_f \max(n_{i,f},0)} = \left(\frac{1}{\Delta V}\right)_k \sum_{f\in F(k)} U_f \max(n_{i,f},0)A_f(\Delta x_i)_k \tag{30}$$

and

$$(\mathbf{\Gamma}_2\mathbf{u}_s)_{ik} = \frac{\sum_{f\in F(k)} U_f \min(n_{i,f},0)A_f}{\sum_{f\in F(k)} A_f \max(-n_{i,f},0)} = \left(\frac{1}{\Delta V}\right)_k \sum_{f\in F(k)} U_f \min(n_{i,f},0)A_f(\Delta x_i)_k. \tag{31}$$

On an unstructured Cartesian mesh, the operators $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ have a simple geometric interpretation. Operator (30) is a face-area weighted average of face velocities to the east, north, or top of cell k depending on the velocity component desired. Similarly, $\mathbf{\Gamma}_2$ is an average of west, south, or bottom velocities. Alternatively, since $A_f(\Delta x_i)_k$ defines a volume, the shift operators may also be considered a volume-weighted average.

The shift operators defined above satisfy the momentum-conservation requirement, namely $\mathbf{\Gamma}\mathbf{1}_s = \mathbf{1}_c$. It follows that $\mathbf{\Gamma}_1$ and 15 $\mathbf{\Gamma}_2$ can be used to construct convective operators which conserve momentum and energy, and they can be used to define symmetric, positive-definite diffusive operators.

The convective and diffusive matrix operators used for the present numerical scheme are obtained by substituting the shift operators $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ into Eqs. (21) and (26). Specifically,

$$\mathscr{C}(u) = \frac{1}{2}\left[\mathbf{\Gamma}_1^* \mathscr{C}_u(u)\mathbf{\Gamma}_1 + \mathbf{\Gamma}_2^* \mathscr{C}_u(u)\mathbf{\Gamma}_2\right] \tag{32}$$

and

$$\mathscr{D} = \frac{1}{2}\left[\mathbf{\Gamma}_1^* \mathscr{D}_u \mathbf{\Gamma}_1 + \mathbf{\Gamma}_2^* \mathscr{D}_u \mathbf{\Gamma}_2\right]. \tag{33}$$

The gradient operator is identical to the general operator (17) used for arbitrary unstructured meshes.

The convective and diffusive operators, (32) and (33), do not decouple the velocity field into even-odd modes, in contrast with the operators produced with the interpolation (28).

Until now, we have focused on the interpretation of $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ as face-to-cell operators. Recall, however, that the conjugate transpose of either matrix is a cell-to-face shift operator. Are the definitions of $\mathbf{\Gamma}_1$ and $\mathbf{\Gamma}_2$ consistent with this second interpretation? To answer this question consider an arbitrary face f with adjacent cells $c1$ and $c2$. The discretization of the convective term at velocity cell f has the form

$$\begin{aligned}
[\mathscr{C}(u)\mathbf{u}_s]_f &= \left[\frac{1}{2}\mathbf{\Gamma}_1^*(\mathscr{C}_u(u)\mathbf{\Gamma}_1\mathbf{u}_s) + \frac{1}{2}\mathbf{\Gamma}_2^*(\mathscr{C}_u(u)\mathbf{\Gamma}_2\mathbf{u}_s)\right]_f \\
&= \frac{1}{2}\left(\frac{A_f(\Delta x_f)_{c1}}{\Delta V_{c1}}\right)\left(\Delta V\frac{\delta(u_j u_i)}{\delta x_j}\right)_{c1} + \frac{1}{2}\left(\frac{A_f(\Delta x_f)_{c2}}{\Delta V_{c2}}\right)\left(\Delta V\frac{\delta(u_j u_i)}{\delta x_j}\right)_{c2} \\
&= \frac{1}{2}\left(A_f(\Delta x_f)_{c1}\frac{\delta(u_j u_i)}{\delta x_j}\bigg|_{c1}\right) + \frac{1}{2}\left(A_f(\Delta x_f)_{c2}\frac{\delta(u_j u_i)}{\delta x_j}\bigg|_{c2}\right),
\end{aligned}$$

where $(\Delta x_f)_{c1}$ and $(\Delta x_f)_{c2}$ are the lengths of $c1$ and $c2$, respectively, in the direction of $\mathbf{n}_f$. We see that $\mathbf{\Gamma}_1^*$ and $\mathbf{\Gamma}_2^*$ scale the cell convective fluxes to account for the difference in volume between the velocity cell and its adjacent collocated cells. Therefore, the cell-to-face interpretation of the shift operators is appropriate although the accuracy is still uncertain.

3.3. Boundary conditions

No-slip and periodic boundary conditions (BCs) were implemented in the numerical method. The implementation of the periodic BCs, while requiring careful "bookkeeping", was straightforward. Application of the no-slip condition was not so clear. Indeed, there are several possible discretizations of this boundary condition; only two are mentioned here.

The shifting of the convective and diffusive operators suggests a similar strategy for the no-slip BCs. That is, shift the velocities to the pressure cells, apply the no-slip condition, and return the resulting shear stress to the velocity cells. This implementation, however, does not guarantee a negative source term linearization for some velocity cells, e.g. those in Fig. 5. When used for simulations of the turbulent channel, this BC was found to produce negative wall velocities on the order of $-U_{\max}$; although, stability did not seem to be compromised.

Alternatively, the shear stress can be applied directly to the velocity cells without averaging through a shift operator. Applying the BC directly raises the question of which velocity cells should receive the shear stress. Clearly, any u- or w-velocity cell adjacent to the wall at $y = 0$ should be subjected to the no-slip condition. However, restricting the BC to wall-adjacent cells would neglect a viscous stress term in some cells, such as P in Fig. 5. For example, if cell C_1 was in the interior of the domain, the viscous stress resulting from cells to the south would get added to velocity cell P. Hence, because cell C_1 has a wall to its south,

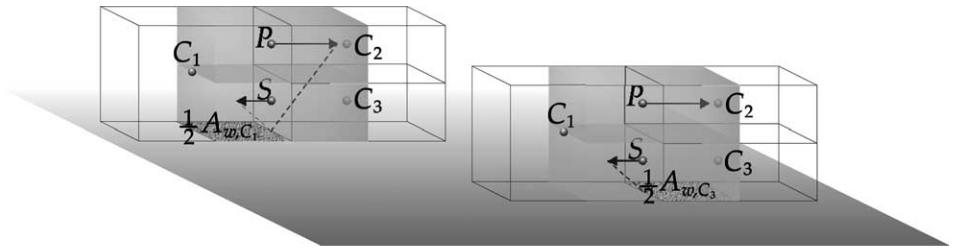

Fig. 5. No-slip boundary conditions.

a wall shear stress term should be added to cell P. The shear stresses for cells P and S in the second approach are

$$\tau_{w,P} = -\frac{A_{w,C_1}/2}{Re(\Delta y_{w,P})} u_P, \tag{34}$$

$$\tau_{w,S} = -\frac{(A_{w,C_1}/2 + A_{w,C_3}/2)}{Re(\Delta y_{w,S})} u_S, \tag{35}$$

where $\Delta y_{w,P}$ and $\Delta y_{w,S}$ are the perpendicular distances from the wall to cells P and S, respectively. Notice that the wall areas are divided by two; cell S gets a second wall stress term from cell C_3 and cell P will get a viscous stress from its east collocated neighbour C_2.

The BCs were implemented using the direct approach, (34), (35), in the final version of the method. This implementation is absolutely stable, since the boundary source term is linearized using a negative coefficient. Note that the no-slip BCs reduce to the desired second-order method on a structured mesh.

3.4. Time-integration method

Until now, the analysis has focused exclusively on the spatial discretization of the Navier–Stokes equations. Convective, gradient, and diffusive operators have been constructed that conserve momentum and provide for stability through inviscid conservation of energy. Of course, the time derivative in (7) must also be treated carefully if these properties are to be retained for discrete time integration.

In order to conserve energy and momentum the time advancement must be implicit [18]. The simplest second-order time integration scheme, that conserves energy, is the midpoint implicit method. Let Δt be the time step size between time t^{n+1} and t^n. Hence, the midpoint method applied to (7) gives

$$\mathbf{\Omega}\frac{\mathbf{u}_s^{n+1} - \mathbf{u}_s^n}{\Delta t} + \mathscr{C}\left(\frac{u^{n+1} + u^n}{2}\right)\left(\frac{\mathbf{u}_s^{n+1} + \mathbf{u}_s^n}{2}\right) + \mathscr{D}\left(\frac{\mathbf{u}_s^{n+1} + \mathbf{u}_s^n}{2}\right) - \mathscr{M}^*\left(\frac{\mathbf{p}_c^{n+1} + \mathbf{p}_c^n}{2}\right) = \mathbf{0}. \tag{36}$$

The discretization (36) conserves momentum and energy in the absence of viscosity, and it ensures the energy is monotone decreasing if viscosity is non-zero.

The system (36) satisfies our requirements for conservation and could be solved as it stands; see, for example [5] for its solution on a structured grid. However, most turbulent flows require a small time step to resolve the fastest scales, and, if the time steps are sufficiently small, the discrete Navier–Stokes equation can be linearized and decoupled to reduce the computation time without compromising accuracy or conservation. Although many authors adopt this approach or variations of it, for example [22,16,1], the work of Ham et al. [5] suggests that the added expense of solving the fully coupled and non-linear equation may be justified since the time step can be increased substantially. Obviously, further work is needed in this area.

For this work, the discrete Navier–Stokes equations are linearized and decoupled. An appropriate linearization of the convective term, which retains the second-order time accuracy, is given by [4]

$$\mathscr{C}\left(\frac{u^{n+1}+u^n}{2}\right)\left(\frac{\mathbf{u}_s^{n+1}+\mathbf{u}_s^n}{2}\right)=\mathscr{C}\left(\frac{1}{2}\delta u\right)\mathbf{u}_s^n+\mathscr{C}(u^n)\frac{1}{2}\boldsymbol{\delta}\mathbf{u}_s+\mathscr{C}(u^n)\mathbf{u}_s^n+\mathrm{O}(\Delta t^2),$$

where $\boldsymbol{\delta}\mathbf{u}_s=\mathbf{u}_s^{n+1}-\mathbf{u}_s^n$. A fractional step method [11] was used to decouple the pressure and velocity. The momentum equations are advanced using the old pressure p^n to obtain the first intermediate velocity $\hat{\mathbf{u}}_s$. The effect of the old pressure gradient is subsequently removed from the velocity field to produce $\tilde{\mathbf{u}}_s$. The new velocity, $\mathbf{u}_s^{n+1}$, is obtained by adding the new pressure gradient; since the new velocity is divergence free, a Poisson equation for the new pressure results. These steps are summarized below in their mathematical form.

$$\boldsymbol{\Omega}\frac{\boldsymbol{\delta}\hat{\mathbf{u}}}{\Delta t}+\mathscr{C}\left(\frac{1}{2}\delta\hat{u}\right)\mathbf{u}_s^n+\mathscr{C}(u^n)\frac{1}{2}\boldsymbol{\delta}\hat{\mathbf{u}}_s+\mathscr{C}(u^n)\mathbf{u}_s^n+\mathscr{D}\left(\frac{1}{2}\boldsymbol{\delta}\hat{\mathbf{u}}_s+\mathbf{u}_s^n\right)-\mathscr{M}^*\mathbf{p}_c^n=\mathbf{0}, \tag{37}$$

$$\tilde{\mathbf{u}}_s-\hat{\mathbf{u}}_s=-\Delta t\boldsymbol{\Omega}^{-1}\mathscr{M}^*\mathbf{p}_c^n, \tag{38}$$

$$\mathscr{M}\boldsymbol{\Omega}^{-1}\mathscr{M}^*\mathbf{p}_c^{n+1}=-\frac{1}{\Delta t}\mathscr{M}\tilde{\mathbf{u}}_s, \tag{39}$$

$$\mathbf{u}_s^{n+1}-\tilde{\mathbf{u}}_s=\Delta t\boldsymbol{\Omega}^{-1}\mathscr{M}^*\mathbf{p}_c^{n+1}, \tag{40}$$

where $\boldsymbol{\delta}\hat{\mathbf{u}}_s=\hat{\mathbf{u}}_s-\mathbf{u}_s^n$. This approximate factorization is similar to the collocated fractional step method of [4]. Eq. (37) was solved using a Jacobi solver, and, as with the collocated version, the residual was reduced by about six orders of magnitude with approximately 15 iterations. Unlike the collocated version, some relaxation of Eq. (37) was necessary to obtain convergence: a relaxation constant of 0.8 was sufficient. Finally, notice that the Laplacian operator in (39) is discretized using a symmetric, positive-definite matrix; hence, the pressure equation was solved using a diagonally-preconditioned conjugate-gradient method.

A splitting error is associated with using $\hat{\mathbf{u}}_s$ instead of $\mathbf{u}_s^{n+1}$ in the momentum equation (37). The error can be found by adding Eqs. (38) and (40):

$$\mathbf{u}_s^{n+1}-\hat{\mathbf{u}}_s=\Delta t\boldsymbol{\Omega}^{-1}\mathscr{M}^*(\mathbf{p}_c^{n+1}-\mathbf{p}_c^n)=\Delta t^2\boldsymbol{\Omega}^{-1}\mathscr{M}^*\left[\frac{\mathrm{d}\mathbf{p}_c}{\mathrm{d}t}+\mathrm{O}(\Delta t)\right]. \tag{41}$$

Hence, despite the linearization and splitting errors, the overall numerical method remains second-order accurate in time.

The linearization and fractional step errors also affect the discrete energy equation and, therefore, global energy conservation. The effect of these errors on the energy equation can be shown to be second-order in time. For example, the convective operator $\mathscr{C}(\delta\hat{u}/2)$ is not perfectly skew-symmetric because $\hat{\mathbf{u}}_s$ does not satisfy the continuity condition; therefore the matrix diagonal does not vanish. However, the difference velocities used by this operator may be replaced by δu^{n+1} with only a Δt^2 penalty, see Eq. (41) – alternatively, the convective operator could be replaced by the skew-symmetric operator $\mathscr{C}(\delta\hat{u}/2)-\mathrm{diag}(\mathscr{C}(\delta\hat{u}/2))$ as in [22].

4. Adaptation

4.1. Adaption criterion

The adaptive mesh was introduced in Section 3. Beginning with a uniform, structured Cartesian mesh, the mesh is refined and coarsened at each time step using anisotropic bisection or binary agglomeration of the cells. This subsection presents the criterion used to determine which cells are refined and which are coarsened.

The adaptation criterion is based on the criterion used by Ham et al. [4]. Their criterion leads to the following target cell sizes:

$$\Delta x_{\text{target}} = \left(\frac{64c^2 F_{yy} F_{zz}}{F_{xx}^4}\right)^{1/10}, \tag{42}$$

$$\Delta y_{\text{target}} = \left(\frac{64c^2 F_{xx} F_{zz}}{F_{yy}^4}\right)^{1/10}, \tag{43}$$

$$\Delta z_{\text{target}} = \left(\frac{64c^2 F_{xx} F_{yy}}{F_{zz}^4}\right)^{1/10}, \tag{44}$$

where

$$F_{x_i x_i} = \sqrt{\left(\frac{\partial^2 u}{\partial x_i^2}\right)^2 + \left(\frac{\partial^2 v}{\partial x_i^2}\right)^2 + \left(\frac{\partial^2 w}{\partial x_i^2}\right)^2} \quad \text{(no sum } i\text{)}. \tag{45}$$

In [4], the collocated velocities are used in Eq. (45) to find the $F_{x_i x_i}$. For the present method, the velocities in (45) are obtained using a face-area weighted average to interpolate the staggered velocities to the cell centres, specifically, Eq. (28). These interpolated velocities are then used to find the target cell sizes. A criterion can be derived for the staggered velocities using the Laplacian operator (33), but the appropriate refinement for individual velocity cells is unclear; how would a single velocity cell refine without altering the cuboid shape of its adjacent pressure cells? Further work on the adaptation criterion for staggered meshes is warranted.

The criterion used to derive Eqs. (42)–(44) assumes non-vanishing second derivatives. In the event that all the variables are locally, or globally, linear in some direction, an appropriate two-dimensional criterion is adopted. For example, if the flow is locally linear in the z-direction then the target cell sizes become

$$\Delta x_{\text{target}} = \left(\frac{144c^2 F_{yy}}{F_{xx}^3}\right)^{1/8}, \tag{46}$$

$$\Delta y_{\text{target}} = \left(\frac{144c^2 F_{xx}}{F_{yy}^3}\right)^{1/8}, \tag{47}$$

$$\Delta z_{\text{target}} = \Delta z_{\text{current}}. \tag{48}$$

Similarly, if the flow is locally linear in two coordinates, say z and y, then we use

$$\Delta x_{\text{target}} = \left(\frac{24c}{F_{xx}}\right)^{1/3}, \tag{49}$$

$$\Delta y_{\text{target}} = \Delta y_{\text{current}}, \tag{50}$$

$$\Delta z_{\text{target}} = \Delta z_{\text{current}}. \tag{51}$$

4.2. Adaptation and conservation

When a pressure cell is refined a new face is added which bisects the cell into two new cells. If the refinement is in the x-direction, for example, the faces in the y- and z-directions are split if necessary. The addition of a new face and the splitting of old faces requires an interpolation of the velocities stored at these faces. Similarly, an interpolation is also needed for some faces during coarsening. Various interpolations are available, but if mass, momentum, or energy are to be conserved the choices are severely restricted.

In the Cartesian method of [4], the face velocities are interpolated to conserve mass and the collocated velocities are interpolated to conserve momentum. In the present staggered method the face velocities are the only velocities available. Ideally, the face velocities would be interpolated to conserve mass, momentum, and energy, but this may not be possible in general. Since mass conservation is important for both momentum and energy conservation, the interpolation was constructed to ensure that discrete continuity is not violated.

Errors may be introduced by the mass conserving interpolation into the other quantities, such as momentum and energy. Taylor series expansions can be used to show that these errors are second order in the local grid spacing; the analysis is tedious and straightforward, and, therefore, it is omitted. To validate the second-order behaviour of the interpolation, the flow in a periodic box with unit dimensions was computed. The initial conditions were constructed using a random, divergence-free velocity field:

$$u_0 = \sum_{i=1}^{4}\sum_{j=1}^{4}\sum_{k=1}^{4} A_{ijk}\sin(2i\pi x)\sin(2j\pi y)\sin(2k\pi z),$$

$$v_0 = \sum_{i=1}^{4}\sum_{j=1}^{4}\sum_{k=1}^{4} B_{ijk}\sin(2i\pi x)\sin(2j\pi y)\sin(2k\pi z),$$

$$w_0 = \sum_{i=1}^{4}\sum_{j=1}^{4}\sum_{k=1}^{4}\left[A_{ijk}\frac{i}{k}\cos(2i\pi x)\sin(2j\pi y) + B_{ijk}\frac{j}{k}\sin(2i\pi x)\cos(2j\pi y)\right]\cos(2k\pi z),$$

where the coefficients A_{ijk} and B_{ijk} were random numbers in the range $[-0.1, 0.1]$. The viscosity was set to $\nu = 1/2000$ initially and subsequently turned off at $t = 1.5$. The total simulation time was $t = 3.0$ which provided an equal sample size for the viscous and inviscid flows. A constant time step was used, and it was calculated such that the initial CFL number would equal two.

The local kinetic energy of a cell k was defined as

$$\mathrm{KE}_{\mathrm{local}} = \sum_{f\in F(k)} u_f^2 A_f(\Delta x_{i,k}|n_{i,f}|). \tag{52}$$

Using the above definition, an error measurement for the kinetic energy conservation was constructed as follows: for each cell that was refined or coarsened, the difference between the local kinetic energy before and after adaptation was calculated and squared. The squared errors were then summed and divided by the total number of adaptations; refinement of a cell or coarsening of two cells were each counted as one adaptation. Finally, a relative error was found by taking the square root of the summed errors and dividing the result by the total energy before adaptation.

An average cell dimension, Δ_{avg}, was calculated by summing the volumes of all cells which were adapted; the volume before adaptation was added for refining cells and the volume after adaptation was added for

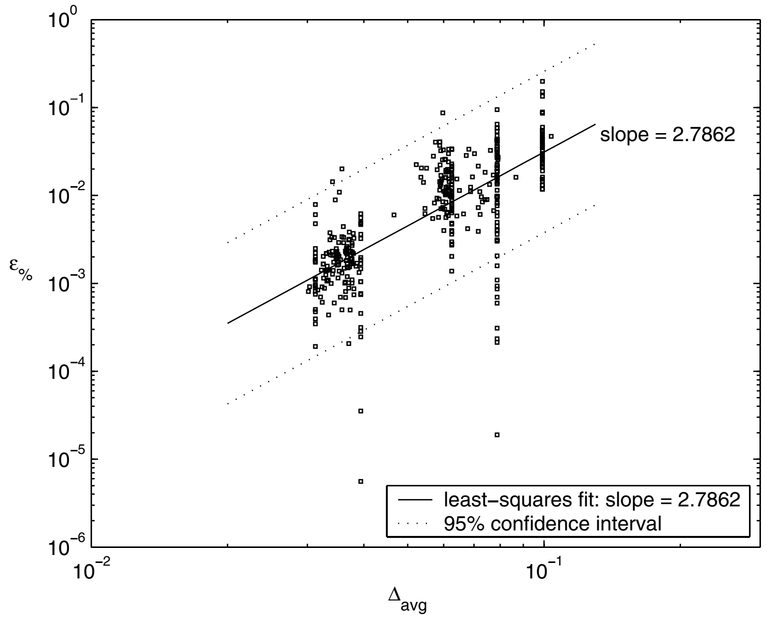

Fig. 6. Energy conservation error in the adaptive interpolation.

coarsening cells. The summed volumes were divided by the total number of adaptations, and the cubic root of the result was taken.

Fig. 6 plots the adaptive interpolation error versus the average adapted cell dimension. A least-squares-fit shows that the error is approximately second order as predicted. The fit was obtained by transforming the data to logarithmic space, finding a linear least-squares-fit, and then transforming back. Fig. 6 also shows the 95% confidence interval for the transformed data.

5. Accuracy

5.1. Error analysis

Traditionally, the truncation error is used to find the accuracy of a method; however, Manteuffel and White [14] have shown that the truncation error only provides a lower bound for a method's accuracy. The resulting lower bound may not reflect the true error. This is the case for the present method.

The truncation error for the Navier–Stokes discretization can be found by substituting the analytical solution into Eq. (7). For the unstructured, staggered Cartesian method the truncation error derivation is tedious and has been omitted. The truncation error contains, in general, low order terms which suggest that the method is inconsistent. Practically, however, the low-order terms are paired because of the way the mesh adapts using bisection, and these paired terms cancel. When the mesh is locally uniform the method reduces to the standard staggered scheme and becomes second-order accurate. This raises the question: which errors dominate the solution?

There are two distinct types of low-order terms present in the general form of the truncation error. On a one-dimensional mesh, the first type of error is proportional to $\Delta V_f(\Delta x_{i+1} - \Delta x_i)$ and is the result of the cell dimension changing in the x-direction. This error will be referred to as the one-dimensional error and is analyzed in Appendix A. The second type of error accounts for the remaining low-order terms and is primarily due to misalignments between the face and cell centroids; such errors will be labelled unstructured errors.

The analysis in Appendix A shows that the present method is second order on a one-dimensional domain; the presence of first order, one-dimensional errors does not result in a first-order scheme. Generalizing this remark to higher dimensions is difficult because of the unstructured errors. The example below provides anecdotal evidence that the order of accuracy is also not reflected by the unstructured errors. Clearly an example does not constitute a proof of the method's order of accuracy, but it does reveal the naivety of using the truncation error to deduce the actual error.

On a two- or three-dimensional Cartesian adaptive mesh, changes in cell dimension will usually occur over surfaces because of the continuous criterion. Therefore, for simplicity, consider a two-part two-dimensional uniform mesh with cell dimensions Δx and Δy for $x < \hat{x}$ and Δx and $\Delta y/2$ for $x \geqslant \hat{x}$. A portion of this mesh is shown in Fig. 7 where several u-velocity cells have been labelled. Let $\Delta V = \Delta x \Delta y \Delta z$ and $A_i = \Delta V/\Delta x_i$.

The individual truncation errors for the cells N and S in Fig. 7 can be shown to be O(1). Instead of the smaller cells, consider an imaginary cell P created by the agglomeration of N and S, and let $u_P = (u_N + u_S)/2$. Approximating u_P by the average of the velocities at N and S produces a second order error, but, of more interest, is the accuracy of the Navier–Stokes discretization at P created by summing the discretizations at N and S. Each of the terms in the discretization are examined below.

Adding the convective discretizations at N and S yields an approximation for the convective flux at P over a volume of size ΔV:

$$\begin{aligned}\left[(\mathscr{C}(u)\mathbf{u}_s)_N + (\mathscr{C}(u)\mathbf{u}_s)_S\right] &= \underbrace{\left[\left(\frac{u_{NE}+u_N}{2}\right)\left(\frac{u_{NE}+u_N}{2}\right) + \left(\frac{u_{SE}+u_S}{2}\right)\left(\frac{u_{SE}+u_S}{2}\right)\right]\frac{A_x}{2}}_{\text{convective flux through east face of } P} \\ &\quad - \underbrace{\left[\left(\frac{u_N+u_S}{4}+\frac{u_W}{2}\right)\left(\frac{u_N+u_S}{4}+\frac{u_W}{2}\right)\right]A_x}_{\text{convective flux through west face of } P} \\ &\quad + \underbrace{\left[\left(\frac{v_{nw}+v_{ne}}{2}\right)\left(\frac{u_N+u_S}{8}+\frac{u_{NNN}+u_{NN}}{8}+\frac{u_N+u_{NN}}{4}\right)\right]A_y}_{\text{convective flux through north face of } P} \\ &\quad - \underbrace{\left[\left(\frac{v_{sw}+v_{se}}{2}\right)\left(\frac{u_N+u_S}{8}+\frac{u_{SSS}+u_{SS}}{8}+\frac{u_S+u_{SS}}{4}\right)\right]A_y}_{\text{convective flux through south face of } P} + R_P,\end{aligned}$$

where R_P denotes the unused terms

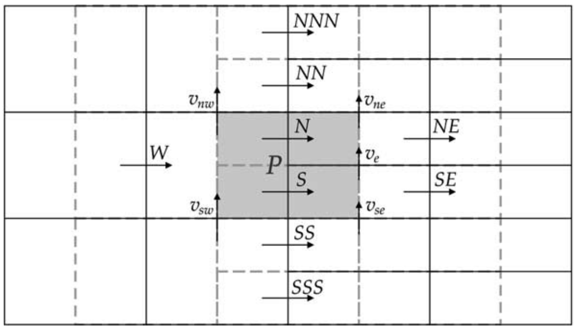

Fig. 7. Example of mesh with unstructured type errors.

$$R_P = \left(\frac{A_y v_{nw}}{4} - \frac{A_y v_{ne}}{4}\right)\left(\frac{u_N + u_S}{4} + \frac{u_{NNN} + u_{NN}}{4} - \frac{u_N + u_{NN}}{2}\right)$$
$$- \left(\frac{A_y v_{sw}}{4} - \frac{A_y v_{se}}{4}\right)\left(\frac{u_N + u_S}{4} + \frac{u_{SSS} + u_{SS}}{4} - \frac{u_S + u_{SS}}{2}\right) - \frac{2A_x}{16}(u_N - u_S)^2.$$

If we ignore the term R_P, summing the convective fluxes at N and S produces a second-order accurate discretization of the convective flux at P. Moreover, closer inspection of the remaining terms reveals that $R_P = \mathrm{O}(\Delta y^2 A)$. It follows that

$$\iiint_{\Delta V_P} \frac{\partial}{\partial x_j}(u_j u)\mathrm{d}V = \left[(\mathscr{C}(u)\mathbf{u}_s)_N + (\mathscr{C}(u)\mathbf{u}_s)_S\right] + \mathrm{O}(\Delta y^2 A). \tag{53}$$

Eq. (53) shows that error in convective term is $\mathrm{O}(\Delta x)$, after the flux is divided by ΔV_P.

Next, consider the diffusive flux at P. Combining the diffusive flux at N and S gives

$$\left[(\mathscr{D}\mathbf{u}_s)_N + (\mathscr{D}\mathbf{u}_s)_S\right] = \frac{1}{Re}\left[u_W + \left(\frac{u_{NE} + u_{SE}}{2}\right) - 2\left(\frac{u_N + u_S}{2}\right)\right]\frac{A_x}{\Delta x}$$
$$+ \frac{1}{2Re}\left[\left(\frac{u_{NN} + u_{NNN}}{2}\right) + \left(\frac{u_{SS} + u_{SSS}}{2}\right) - 2\left(\frac{u_N + u_S}{2}\right)\right]\frac{A_y}{\Delta y}$$
$$+ \frac{1}{2Re}\left[(u_{NN} - u_N) + (u_{SS} - u_S)\right]\frac{A_y}{\frac{1}{2}\Delta y}.$$

The above discretization is a second-order accurate approximation of the diffusive flux at P. Note how the diffusive fluxes on the north and south faces use an average of two face derivatives; one with a Δy wide stencil and the other with a $\Delta y/2$ wide stencil.

Finally, the combined pressure gradient is given by

$$-\left[(\mathscr{M}\mathbf{u}_s)_N + (\mathscr{M}\mathbf{u}_s)_S\right] = A_y\left(\frac{p_n + p_s}{2} - p_w\right) \approx \iiint_{\Delta V_P} \frac{\partial p}{\partial x}\,\mathrm{d}V. \tag{54}$$

The resulting pressure gradient discretization (54) is also second-order accurate for the cell at P.

For the mesh in Fig. 7, the above analysis has shown that a second-order accurate discretization of the Navier–Stokes equations can be obtained for the cell P by summing the discretizations at the cells N and S. Now, if a smooth analytical solution exists then u_N and u_S must converge to u_P as $\max(\Delta x_i) \to 0$. Specifically,

$$u_N = u_P + \left(\frac{\partial u}{\partial y}\right)_P \frac{\Delta y}{4} + \mathrm{O}(\Delta y^2)$$

and

$$u_S = u_P - \left(\frac{\partial u}{\partial y}\right)_P \frac{\Delta y}{4} + \mathrm{O}(\Delta y^2),$$

so the error at N and S must be bounded by $\mathrm{O}(\Delta y)$ on this particular mesh, despite a truncation error of $\mathrm{O}(1)$.

In general, the surfaces where the mesh changes from one size to another will neither be aligned with the coordinate axes nor planar. However, the above argument does demonstrate that the accuracy of the method cannot be inferred from a truncation error analysis. Numerical experiments were conducted to further assess the accuracy of the method; these are discussed in the next subsection.

5.2. Tests of numerical accuracy

5.2.1. The convective and diffusive operators on random meshes

The orders of accuracy of the convective term, $\mathscr{C}(u)\mathbf{u}_s$, and the diffusive term, $\mathscr{D}\mathbf{u}_s$, were assessed by evaluating these terms on random meshes and comparing with analytical solutions. The mesh spacing was effectively changed by varying the frequency of the input functions. This approach was used by Perot in [18] for triangular unstructured meshes.

The input velocities for the convective term were produced from a stream function on a square domain with dimensions of 100×100. The input velocities were

$$u_c(x,y) = -\sin\left(\frac{2\pi N}{100}x\right)\sin\left(\frac{2\pi N}{100}y\right),$$

$$v_c(x,y) = -\cos\left(\frac{2\pi N}{100}x\right)\cos\left(\frac{2\pi N}{100}y\right).$$

The parameter N determines the number of modes on the domain and was varied from 1 to 10. For the diffusive term, the input velocities were

$$u_d(x,y) = u_c(x,y) + \sin\left(\frac{2\pi N}{100}x\right),$$

$$v_d(x,y) = v_c(x,y) + \cos\left(\frac{2\pi N}{100}y\right).$$

The random meshes were constructed using a 32×32 uniform mesh as an initial mesh and performing a random refinement process either once or twice. In the refinement process, each cell was randomly refined, with each direction being refined independently. If two random refinements were used, the maximum possible ratio between $\max(\Delta x_i)$ and $\min(\Delta x_i)$ was 4. Fig. 8 shows one possible random mesh. An average mesh size, Δx_{avg}, was calculated for each random mesh. The relative mesh size was then set equal to the quotient

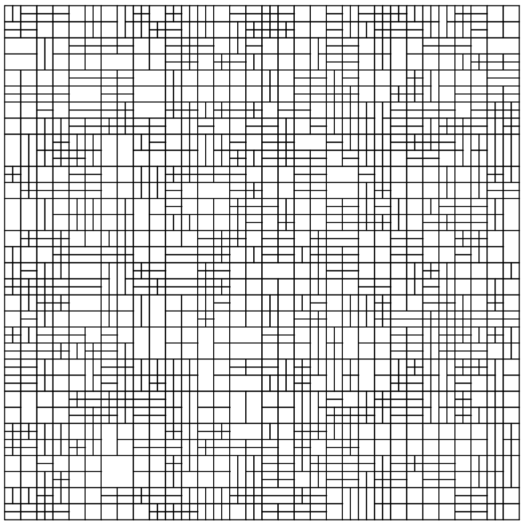
Fig. 8. Example of a random mesh used to assess the accuracy of the convective and diffusive terms.

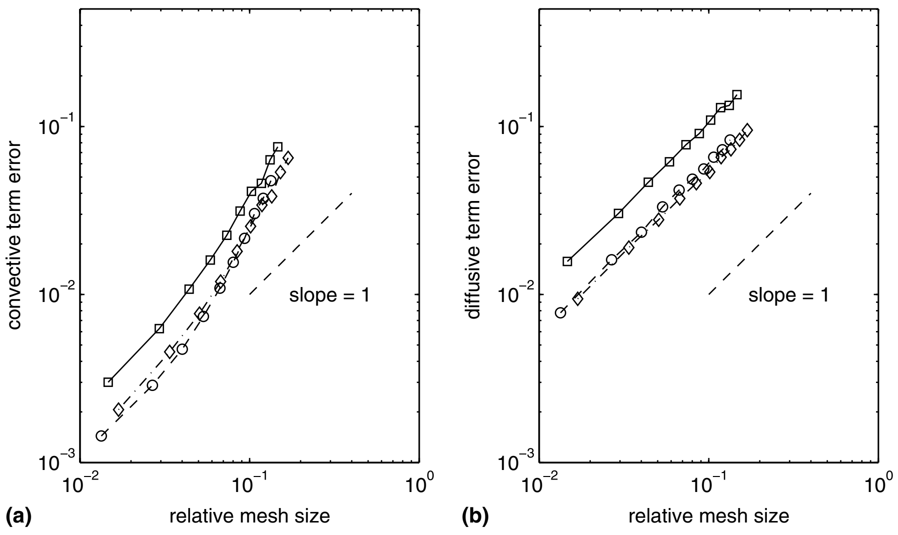

Fig. 9. RMS error in the convective term (a), and the diffusive term (b), for three randomly refined meshes.

of the average mesh size and the wave length 100/*N*; hence the relative mesh size is inversely proportional to the number of grid points per wave length [18].

The root-mean-square (RMS) errors for the convective and diffusive terms are shown in Fig. 9(a) and (b), respectively. The results from three randomly adapted meshes are plotted. These results show that the convective and diffusive terms are first-order accurate for anisotropic Cartesian meshes. First order accuracy for the diffusive term is perhaps surprising given the O(1) approximation for the gradient used on some faces; however, as argued in Section 5.1, these low-order terms appear to cancel for anisotropic Cartesian meshes.

For practical problems the mesh adapts in an effort to minimize the error in the solution, so the first-order accuracy displayed by the convective and diffusive terms on random meshes is an upper bound on the accuracy. Indeed, when the adaptive criteria were applied to the velocities in this test case, a uniform mesh was produced. To assess the accuracy of the adaptive method, a more challenging flow was studied.

5.3. Lid-driven cavity

The two-dimensional lid-driven cavity flow was also used to evaluate the accuracy of the method. The error tolerance *c*, used for the adaptive criterion, was dynamically changed to force the mesh to a certain size. The adapted mesh was then fixed and not allowed to change. Subsequently, the numerical method was advanced until a steady solution was obtained. A Reynolds number of 400 was used.

The error for the cavity flow was estimated using the numerical solution of Ghia et al. [3]. The adaptive mesh solution was interpolated to the 15 interior points, along the cavity's vertical mid-line, reported in [3]. The RMS error between the interpolated solution and the solution of [3] was then calculated:

$$\varepsilon_{\mathrm{rms}} = \sqrt{\frac{1}{15}\sum_{i=1}^{15}(u_{\mathrm{Ghia}} - u_{\mathrm{adapt}})^2}.$$

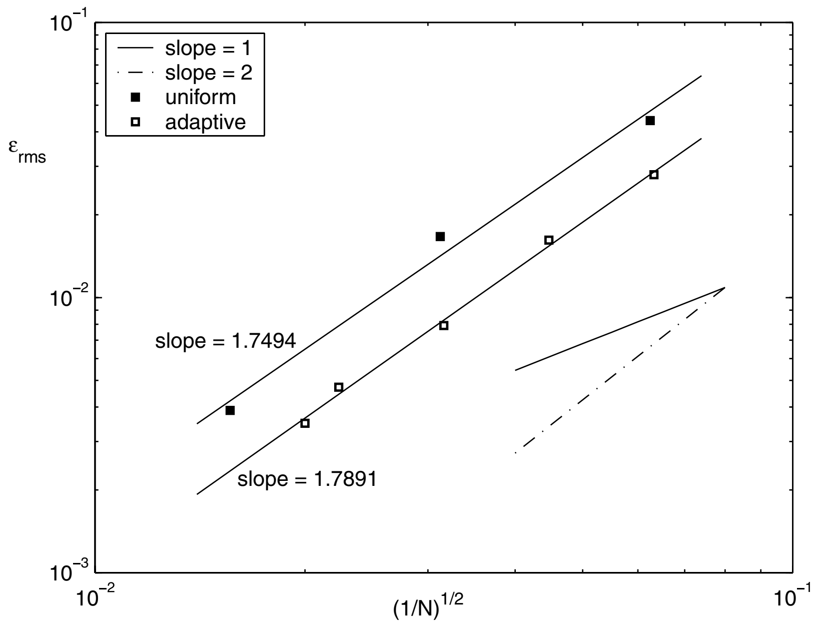

Fig. 10. RMS error for the lid-driven cavity flow.

Fig. 10 shows the RMS error for several adaptive and uniform meshes. Unlike the previous test case, the accuracy is closer to second-order despite the greater complexity of the flow.

6. Direct numerical simulations of the turbulent channel flow

6.1. Channel flow characteristics and simulation set-up

The channel flow consisted of a rectangular hexahedral with dimensions $2\pi\delta \times 2\delta \times \pi\delta$ where δ is the channel half-width. Periodic boundary conditions were imposed in the x- and z-directions, and no-slip boundary conditions were set at $y = 0$ and $y = 2\delta$. The equations of motion were non-dimensionalized using δ and the friction velocity u_τ. The flow was forced in the positive x-direction by a pressure gradient; the pressure gradient was updated dynamically to keep the mass flux constant.

The simulations were performed using a Reynolds number of $Re_b = 2800$ based on the channel half-width, the kinematic viscosity ($v = 1/180$), and the bulk velocity U_b:

$$U_b = \frac{1}{2}\int_{y=0}^{2} u_m \, \mathrm{d}\left(\frac{y}{\delta}\right),$$

where u_m is the time averaged streamwise velocity. The bulk Reynolds number corresponds to a friction velocity Reynolds number of $Re_\tau \approx 180$. Many researchers have used this Reynolds number for turbulent channel simulations, including Kim et al. [12] in their pioneering work. The present results are compared with the spectral DNS of Moser et al. [17], since their results are relatively recent and have benefited from

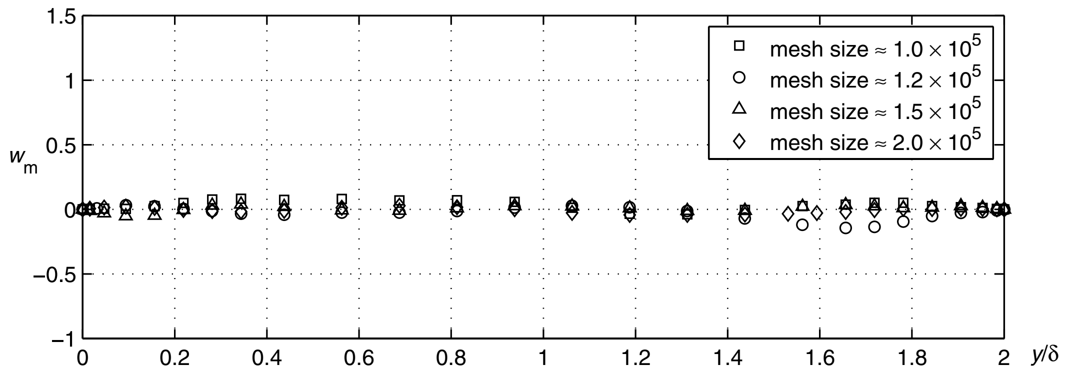

Fig. 11. Mean spanwise velocity profile.

more than a decade of research. The DNS statistics reported in [17] were obtained using $128 \times 129 \times 128$ grid points on a $4\pi\delta \times 2\delta \times 2\pi\delta$ domain.

Simulations were performed with four mesh sizes of approximately 1.0×10^5, 1.2×10^5, 1.5×10^5, and 2.0×10^5 cells based on the number of internal pressure cells. A feedback mechanism was used to adjust the error tolerance c such that the mesh sizes remained close to the specified number of cells. No other restrictions were placed on the adaptation criterion; in particular, the mesh was not forced to assume a particular size near the wall.

A constant time step of 1.25×10^{-3} was adopted for the simulations. This same value was used in the simulations of Verstappen and Veldman [22]. The time step corresponds to a CFL number of approximately one; the results of Choi and Moin [1] and Ham et al. [5] suggest that, in channel flow simulations, the CFL number should not exceed one when fractional step methods are used.

The simulations were carried out for approximately nine non-dimensional time units, $t\, u_\tau/\delta$. This averaging time is slightly smaller than the 10 units sample time of Kim et al. [12]. The mean spanwise velocity can be used to assess the sample size, since, by symmetry, this mean velocity should be zero. Fig. 11 shows the mean spanwise velocity for the four simulations. The spanwise velocity is approximately 3 orders of magnitude smaller than U_b, but not as close to zero as the w_m profile reported by Moser et al. [17]; the latter results have $w_m/U_b \sim 10^{-4}$.

6.2. Statistics gathering on the adaptive mesh

When using structured Cartesian meshes or spectral methods, the statistics are usually averaged over homogeneous planes to increase the sample size. On an adaptive Cartesian mesh, however, averaging over homogeneous planes must be done with care. Suppose the statistics are independently averaged at each unique (discrete) y-coordinate. This approach has two drawbacks: (1) each y-coordinate may have a different sample size, and; (2) certain y-coordinates may be correlated with the velocity fluctuations. Both (1) and (2) were observed in the channel flow if the statistics were gathered at each y-coordinate. For example, the u-velocity fluctuations were found to be correlated with the mesh size in the y-direction. This was likely caused by low-speed fluid being ejected from the wall toward the centre of the channel and causing the mesh to refine. Hence, some mesh points were associated with high u values and others were associated with low u values.

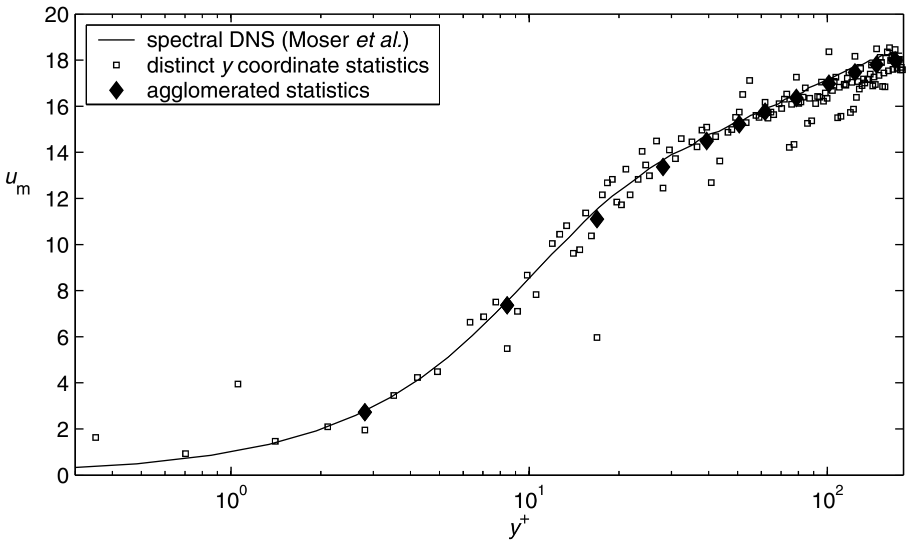

Fig. 12. Agglomerated and raw statistics comparison.

For the present channel flow results, intermediate statistics were gathered by integrating in time and space. The intermediate spatial integration was obtained by multiplying the relevant quantities by the cells's *xz* area. For a given time step, an intermediate statistic may not represent an integral over the entire *xz* plane. Hence, these intermediate statistics were recursively agglomerated, if necessary, until the area of integration was equal to $2(\pi\delta)^2$, i.e. the entire *xz* plane. This procedure was found to eliminate the drawbacks

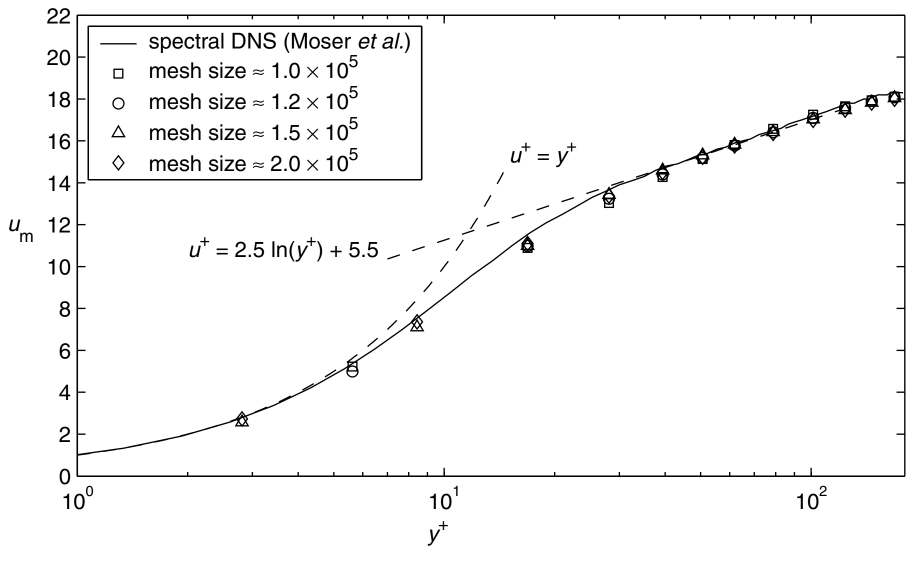

Fig. 13. Mean streamwise velocity in wall units.

discussed above. Fig. 12 compares the distinct y-coordinate statistics with the agglomerated statistics; the spectral data of [17] is included as a reference.

6.3. Mean and RMS velocity profiles

Mean streamwise profiles are plotted in Fig. 13 together with the spectral data of [17]. Fig. 13 is scaled using wall units on a log scale. Note that the friction velocity was taken equal to its exact, non-dimensional value of one; it was not approximated using a discrete wall gradient, since this approach reflects the convergence of u_τ rather than the mean or RMS profiles. The results show that the simulations properly obey the law of the wall and the log-law. Furthermore, all of the mesh sizes follow the spectral data closely, and only underestimate the profile slightly. The agglomerated statistics may give the impression that a relatively coarse, uniform mesh was used; however, recall that the agglomerated statistics are compiled from the raw, distinct y-coordinate statistics.

Using the present method, the finest and coarsest meshes capture the mean profile well using fewer than 2/5 and 1/5, respectively, the number of grid points of the spectral method – the spectral mesh size has been divided by four to account for the larger domain. To emphasize the significance of this result, note that the truncation error for a second-order method is accurate for only 1/4 the Nyquist wave number range. Hence, on a structured mesh, a second-order method would need a mesh approximately 64 times larger than a spectral mesh to ensure an equally accurate solution. From this perspective, the current method requires a mesh two orders of magnitude smaller than a uniform mesh.

The rate of solution convergence is difficult to observe with the mean streamwise profile, since all the mesh sizes seem to perform equally well. Convergence is more obvious with the streamwise fluctuating velocity, u_{rms}. Fig. 14(a) plots u_{rms} for the present simulations and compares these with the results of [17]. The RMS streamwise velocity improves significantly with mesh refinement. The coarser meshes overestimate u_{rms} over the entire range, particularly near the wall; see Fig. 14(b). The RMS velocity from the 2.0×10^5 cell simulation follows the spectral profile well, but shows some poor performance close to the wall where it overestimates the profile more than the next coarser mesh. The source of this discrepancy is not clear; possible reasons are the sample size, domain size, or boundary conditions. Despite its performance

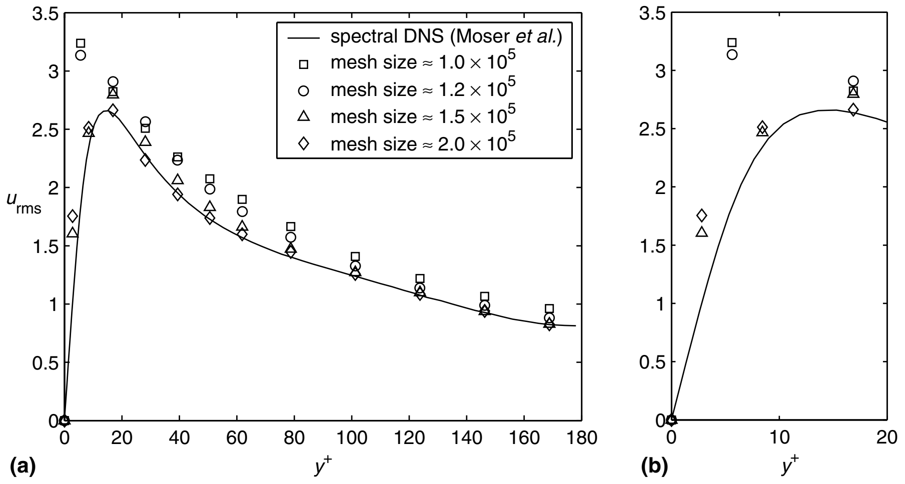

Fig. 14. (a) streamwise RMS fluctuating velocity in wall units: (b) near wall.

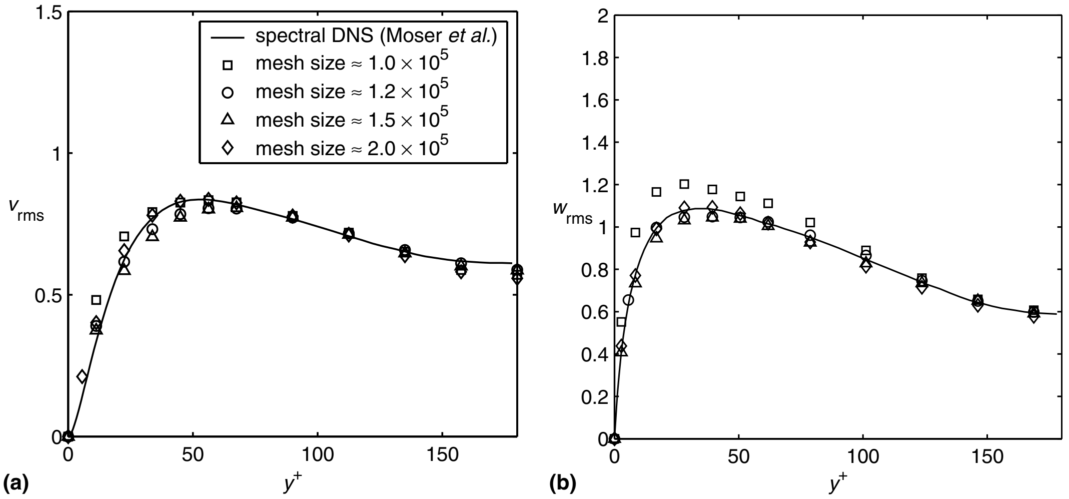

Fig. 15. (a) Wall-normal RMS fluctuating velocity and (b) spanwise RMS fluctuating velocity.

near the wall, the finest mesh result compares well with the spectral results considering the accuracy of the method.

The wall-normal and spanwise fluctuating velocities are plotted in Fig. 15(a) and (b), respectively. Although both profiles show improvement with refinement, the convergence of w_{rms} is more dramatic than that of v_{rms}. Unlike the streamwise fluctuating velocity, the wall-normal and spanwise RMS velocities do not converge monotonically to the spectral data; for example, the profile of w_{rms} from the coarsest mesh overestimates the spectral profile while the other mesh sizes underestimate the profile. One possible explanation of this convergence behaviour is the presence of turbulence structures in the simulations with mesh sizes greater than 1.0×10^5 that are not resolved by the coarsest mesh.

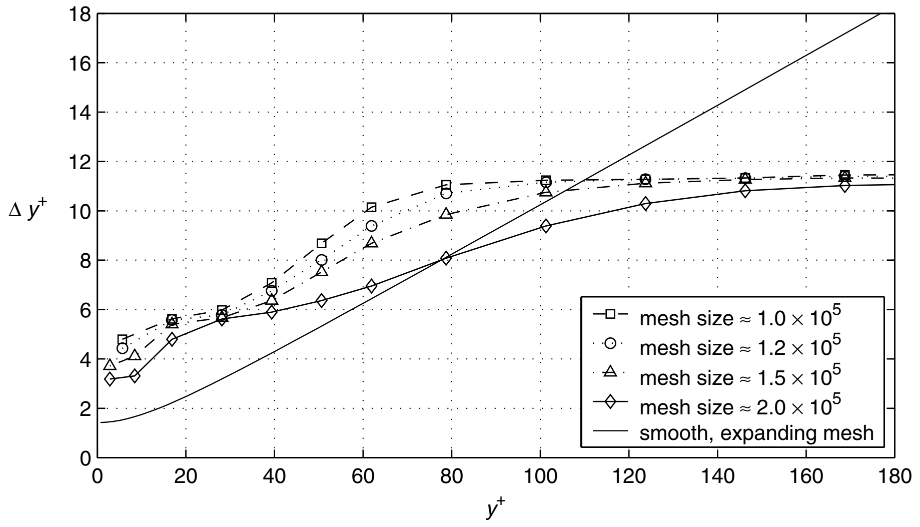

Fig. 16. Mean wall-normal mesh size.

6.4. Mean mesh size statistics

Since the present method uses a time-adaptive mesh, additional statistics were gathered on the mean cell dimensions. The dimensions for the *u*-velocity cells were treated like flow variables and sampled in time and space. Gathering cell size statistics for all the velocity cells and the pressure cells was deemed unnecessary, since, locally, the dimensions are related by a constant factor of no more than 2. Fig. 16 shows the average wall-normal cell size, in wall units, as a function of y^+. As expected, the mesh is most refined near the wall and monotone increasing toward the centre of the channel. Surprisingly, perhaps, the finer meshes do not spread their added degrees of freedom uniformly across the range; rather, the increased resolution tends to be focused close to the wall and in the log-law region. The smooth, expanding mesh used in Verstappen and Veldman's simulations [22], with 64 points in the wall normal direction, is also plotted in Fig. 16. Their smooth mesh emphasizes the near wall resolution at the expense of the channel centre. The adaptive mesh results suggest, however, that the wall region may only need to be resolved periodically, and that the mesh used in [22] may not be sufficiently fine toward the channel centre. This may explain why the fluctuating velocities of the present, second order, method compare better with the spectral data than the fourth-order method of [22].

The mean streamwise and spanwise cell dimensions are plotted in Fig. 17(a) and (b). As the number of cells increases, the streamwise cell dimension converges to a relatively uniform size of $\Delta x^+ \approx 18$, except near the wall where coarser cells are more frequent. The near wall region is dominated by long, low-speed streaks aligned with the *x*-direction, so high streamwise resolution may not be as necessary close to the wall. The spanwise cell dimension also seems to converge to a relatively uniform size, in this case $\Delta z^+ \approx 9$. Unlike the streamwise dimension, however, the spanwise cell size shows the need for some occasional refinement near the wall; this is likely caused by the variation of the low speed streaks in the *z*-direction.

The aim of using an adaptive mesh is to reduce the total number of cells, relative to a structured mesh, necessary for a given accuracy. Unfortunately, no results for the turbulent channel flow using a structured Cartesian mesh and second-order method were available for direct comparison. However, the results of Ham et al. [5] on a smaller domain can be used for an indirect comparison. The second-order results in [5] are obtained on a $16 \times 128 \times 32 \approx 6.5 \times 10^4$ mesh with a domain size of $\pi\delta \times 2\delta \times 0.286\pi\delta$; hence, 10^6 cells would be needed on the full domain. The present results are in good agreement with 2.0×10^5 cells

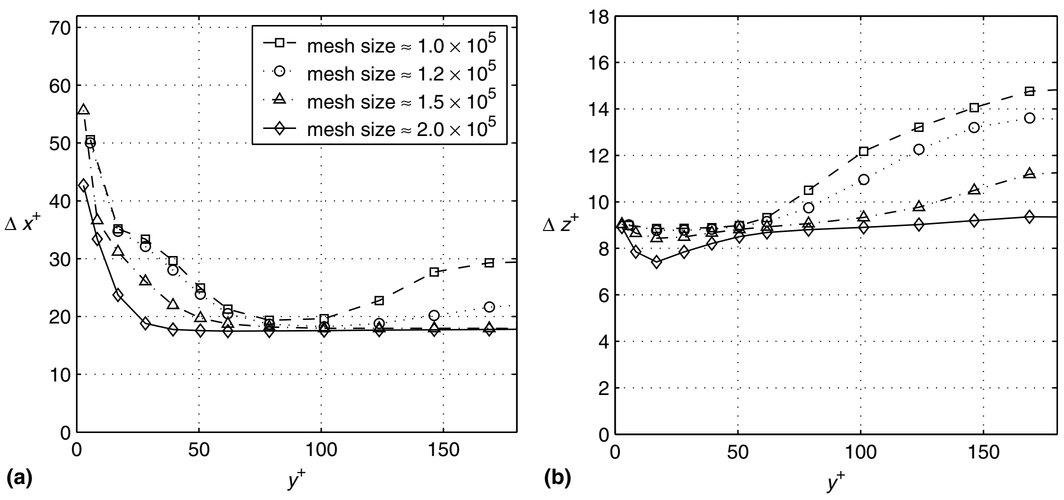

Fig. 17. (a) Mean streamwise and (b) spanwise cell sizes.

on a domain of $2\pi\delta \times 2\delta \times \pi\delta$; however, as the RMS velocities show, even this number of cells is slightly small. Assuming 2.5×10^5 cells would produce excellent agreement with the spectral results, then the present method would also need 10^6 cells on the full domain.

The preceding heuristic analysis suggests that the adaptive mesh does not substantially reduce the number of cells for turbulent channel simulations compared with structured meshes. The channel flow has two directions of homogeneity in which a uniform mesh size may be sufficient; indeed, this is implied by the mean cell size results in the streamwise and spanwise directions. For more general, anisotropic flows we expect the adaptive mesh will reduce the number of grid points relative to a structured mesh. The channel flow has also been studied extensively, so the appropriate cell sizes are well documented. This latter point underlines an important advantage the adaptive mesh has over fixed meshes; in general, the appropriate cell size will not be known a priori.

7. Conclusions

A shift transformation has been introduced which allows collocated operators to be used for staggered variables. The transformation preserves the symmetry properties of discrete collocated operators – symmetry properties that help produce a fully conservative spatial discretization of the Navier–Stokes equations. Various shift operators can be constructed depending on the mesh used and the stencil size. A set of shift transformations have been successfully applied to an unstructured Cartesian adaptive mesh, and the resulting convective and diffusive operators do not decouple the solution. A mass conserving interpolation has been used during mesh adaptation and was shown to conserve momentum and energy to second order in space.

The present method cannot be used to produce a conservative discretization for collocated variables, and the treatment of the pressure term for these schemes remains an open problem; however, the present method can be combined with suitable collocated methods to produce a conservative staggered scheme.

Accuracy remains a difficult issue for conservative methods on general unstructured meshes. The present method has been shown to exhibit second-order convergence when an adaptive mesh is used, and first-order convergence in general. The accuracy of the Cartesian method benefits from the symmetry of the refinements, and more general unstructured meshes may suffer larger errors due to their irregularity. For these meshes, shift operators with larger stencils and higher-order collocated operators may be necessary to obtain second-order accuracy.

Acknowledgements

This work represents part of a M.A.Sc. thesis produced by the first author, whom was supported by a Natural Sciences and Engineering Research Council of Canada (NSERC) scholarship; the first author thanks NSERC for their financial support. The authors wish to thank Dr. Andrew Rau-Chaplin, of Dalhousie University's Computer Science Department, for giving them access to his multi-computer cluster. Finally, the authors thank the referees for their helpful comments and insights.

Appendix A. Errors on a one-dimensional "Cartesian" mesh

Suppose a numerical solution is obtained on a one-dimensional domain of size L_x using an adaptive mesh and the criterion of Section 4.1. Furthermore, suppose the truncation error of the scheme is second order where adjacent cells are uniform and first order where the mesh changes from one size to another. The following argument shows that the global error of such a method is second order.

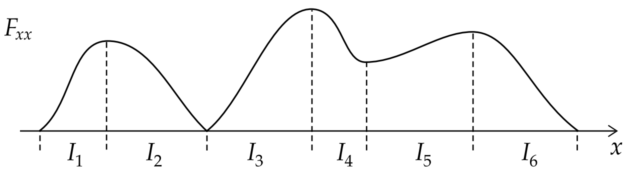

Fig. A.1. Mesh partition into monotonic F_{xx} intervals.

The global order of accuracy is not obvious, since the number of cells with first-order errors may tend to infinity. A common argument used for non-uniform meshes assumes that the mesh spacing is a smooth function of position which implies $\Delta x_{i+1} - \Delta x_i = \mathrm{O}(\Delta x_i^2)$; see, for example [5]. The smooth-mesh argument cannot be used for the present method, since cell sizes change abruptly. An alternate argument is presented below. It is based on the following idea: if the number of cells with second-order truncation errors tends to infinity faster than the number of cells with first-order errors, then the global error will be second order.

Recall from Section 4.1 that, for a one-dimensional problem, the local mesh size satisfies

$$\Delta x_k \leqslant \Delta x_{\text{target}} = \left(\frac{24c}{F_{xx}}\right)^{1/3}. \tag{49}$$

Assume that the second-order derivative is bounded above and vanishes only at a finite number of locations. Therefore, the domain can be divided into a finite number of intervals I_n, $n = 1,\ldots,N$, such that F_{xx} is monotonic on each interval. Fig. A.1 illustrates one possible F_{xx} and the resulting intervals.

The mesh adapts by successive bisections of existing cells, so each cell k must satisfy $\Delta x_k = L_x/2^{l_k}$ for some l_k. Using this restriction and the target cell size (49) yields

$$l_k \geqslant \frac{1}{\ln 2} \ln\left[L_x\left(\frac{F_{xx}(x_k)}{24c}\right)^{1/3}\right], \tag{A.1}$$

$$l_k - 1 \leqslant \frac{1}{\ln 2} \ln\left[L_x\left(\frac{F_{xx}(x_k)}{24c}\right)^{1/3}\right]. \tag{A.2}$$

It follows from (A.2) that the maximum refinement level on I_n is bounded above:

$$\max_{k\in I_n} l_k \leqslant 1 + \frac{1}{\ln 2} \ln\left[L_x\left(\frac{\max_{x\in I_n} F_{xx}}{24c}\right)^{1/3}\right]. \tag{A.3}$$

For intervals satisfying $F_{xx} \neq 0$, Eq. (A.1) can be used to find a lower bound for the minimum refinement level. However, many intervals will have $F_{xx} = 0$ at one of their end points. The algorithm does not permit coarsening beyond the domain, so

$$\min_{k\in I_n} l_k \geqslant 0. \tag{A.4}$$

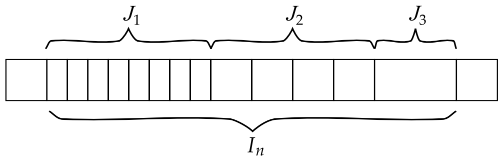

Fig. A.2. constant Δx subintervals.

Eq. (A.4) does not, necessarily, imply that some cells fail to refine as $c \to 0$. For a given $F_{xx}(x_k)$ there always exists a value of c small enough to force another refinement. A possible exception is when a cell exactly satisfies $F_{xx} = 0$. Practically speaking this case should not pose a problem. If necessary, a simple solution would be to restrict the difference in refinement level to one between adjacent cells.

Using the above results, an upper bound can be found for the difference between maximum and minimum refinement levels on each interval I_n:

$$\Delta l_n = \max_{k \in I_n} l_k - \min_{k \in I_n} l_k \leqslant 1 + \frac{1}{\ln 2} \ln \left[L_x \left(\frac{\max_{x \in I_n} F_{xx}}{24c} \right)^{\frac{1}{3}} \right]. \tag{A.5}$$

Hence, for a fixed c, each interval I_n contains a finite number of subintervals $J_{n,m}$, $m = 1, \ldots, M_n = \Delta l_n + 1$, which have constant $\Delta x = \Delta x_m$; refer to Fig. A.2. As $c \to 0$, however, the number of subintervals may tend to infinity; see Eq. (A.5).

If there are K cells on the mesh in total, the total error will have the form

$$\text{Error} = \sum_{n=1}^{N} \sum_{m=1}^{M_n} T_m \Delta x_m \Delta V_m + \mathrm{O}\left(\sum_{k=1}^{K} \Delta x_k^2 \Delta V_k \right), \tag{A.6}$$

where $\Delta V_m = A\Delta x_m$ is the cell volume. The terms $T_m \Delta x_m$ are the first-order errors which occur at the interface of two subintervals and include contributions from the 3 faces surrounding the interface. Let Δx_n be the largest cell size on the interval I_n, so $\Delta x_n = L_x/2^{l_{\min}}$. The subinterval containing Δx_n must be at the beginning or end of I_n, since F_{xx} is monotone increasing or decreasing on I_n. Furthermore, the subinterval adjacent to the one containing Δx_n must have cells with size $\Delta x \leqslant \Delta x_n/2$, and the next subinterval must have $\Delta x \leqslant \Delta x_n/2^2$, and so on. Hence, the global error satisfies

$$\begin{aligned} \text{Error} &\leqslant \sum_{n=1}^{N} \left(\Delta x_n^2 A \sum_{m=1}^{M_n} \frac{T_m}{2^{m-1}} \right) + \mathrm{O}\left(\sum_{k=1}^{K} \Delta x_k^2 \Delta V_k \right) \\ &\leqslant A\tilde{T} \sum_{n=1}^{N} \left(\Delta x_n^2 \sum_{m=1}^{M_n} \frac{1}{2^{m-1}} \right) + \mathrm{O}\left(\sum_{k=1}^{K} \Delta x_k^2 \Delta V_k \right), \end{aligned} \tag{A.7}$$

where $\tilde{T} = \max_n(\max_m |T_m|)$. As $c \to 0$ the number of subintervals on each I_n may tend to infinity, but the geometric sum in (A.7) remains bounded by 2. Thus, continuing from (A.7),

$$\text{Error} \leqslant 2A\tilde{T} \sum_{n=1}^{N} \Delta x_m^2 + \mathrm{O}\left(\sum_{k=1}^{K} \Delta x_k^2 \Delta V_k \right) = \mathrm{O}(\Delta x^2), \tag{A.8}$$

where $\Delta x = \max \Delta x_k$. The inequality (A.8) shows that the global error on a one-dimensional domain is indeed second order.

References

[1] H. Choi, P. Moin, Effects of the computational time step on numerical solutions of turbulent flow, J. Comput. Phys. 113 (1994) 1–4.

[2] J.H. Ferziger, M. Perić, Computational Methods for Fluid Dynamics, Springer, Berlin, 2002.

[3] U. Ghia, K.N. Ghia, C.T. Shin, High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method, J. Comput. Phys. 48 (1982) 387–411.

[4] F.E. Ham, F.S. Lien, A.B. Strong, A Cartesian grid method with transient anisotropic adaptation, J. Comput. Phys. 179 (2002) 469–494.

[5] F.E. Ham, F.S. Lien, A.B. Strong, A fully conservative second-order finite difference scheme for incompressible flow on nonuniform grids, J. Comput. Phys. 177 (2002) 117–133.

[6] F.H. Harlow, J.E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, Phys. Fluids 8 (12) (1965) 2182–2189.
[7] J.M. Hyman, R.J. Knapp, J.C. Scovel, High order finite volume approximations of differential operators on nonuniform grids, Physica D 60 (1992) 112–138.
[8] J.M. Hyman, M. Shashkov, Natural discretizations for the divergence, gradient, and curl on logically rectangular grids, Int. J. Comput. Math. Appl. 33 (1997) 81–104.
[9] J.M. Hyman, M. Shashkov, The orthogonal decomposition theorems for mimetic finite difference methods, SIAM J. Numer. Anal. 36 (1999) 788–818.
[10] J.M. Hyman, M. Shashkov, Mimetic discretizations for Maxwell's equations, J. Comput. Phys. 151 (1999) 881–909.
[11] J. Kim, P. Moin, Application of a fractional-step method to incompressible Navier–Stokes equations, J. Comput. Phys. 59 (1985) 308–323.
[12] J. Kim, P. Moin, R. Moser, Turbulence statistics in fully developed channel flow at low Reynolds number, J. Fluid Mech. 177 (1987) 133–166.
[13] K. Mahesh, G. Constantinescu, P. Moin, A numerical method for large-eddy simulation in complex geometries, J. Comput. Phys. 197 (2004) 215–240.
[14] T.A. Manteuffel, B. Andrew, J. White, The numerical solution of second-order boundary value problems on nonuniform meshes, Math. Comput. 47 (176) (1986) 511–535.
[15] R. Mittal, P. Moin, Suitability of upwind-biased finite difference schemes for large-eddy simulation of turbulent flows, AIAA J. 35 (1997) 1415–1417.
[16] Y. Morinishi, T.S. Lund, O.V. Vasilyev, P. Moin, Fully conservative higher order finite difference schemes for incompressible flow, J. Comput. Phys. 143 (1998) 90–124.
[17] R.D. Moser, J. Kim, N.N. Mansour, Direct numerical simulation of turbulent channel flow up to $Re_\tau = 590$, Phys. Fluids 11 (4) (1999) 943–945.
[18] B. Perot, Conservative properties of unstructured staggered mesh schemes, J. Comput. Phys. 159 (2000) 58–89.
[19] B. Perot, R. Nallapati, A moving unstructured staggered mesh method for the simulation of incompressible free-surface flows, J. Comput. Phys. 184 (2003) 192–214.
[20] C.M. Rhie, W.L. Chow, Numerical study of the turbulent flow past an airfoil with trailing edge separation, AIAA J. 21 (Nov.) (1983) 1525–1532.
[21] M. Shashkov, S. Steinberg, Support-operator finite-difference algorithms for general elliptic problems, J. Comput. Phys. 118 (1995) 131–151.
[22] R.W.C.P. Verstappen, A.E.P. Veldman, Symmetry-preserving discretization of turbulent flow, J. Comput. Phys. 187 (2003) 343–368.