Non-Graded Adaptive Grid Approaches to the Incompressible Navier-Stokes Equations

Frédéric Gibou 1, Chohong Min 2, Hector D. Ceniceros 3

Abstract: We describe two finite difference schemes for simulating incompressible flows on nonuniform meshes using quadtree/octree data structures. The first one uses a cell-centered Poisson solver that yields first-order accurate solutions, while producing symmetric linear systems. The second uses a node-based Poisson solver that produces second-order accurate solutions and second-order accurate gradients, while producing nonsymmetric linear systems as the basis for a second-order accurate Navier-Stokes solver. The grids considered can be non-graded, i.e. the difference of level between two adjacent cells can be arbitrary. In both cases semi-Lagrangian methods are used to update the intermediate fluid velocity in a standard projection framework. Numerical results are presented in two and three spatial dimensions.

Communicated by John Lowengrub (GuestEditor) and Mark Sussman

1 Introduction

Incompressible flows are at the center of countless applications in physical and biological sciences. Uniform Cartesian grids used in numerical simulations are limited in their ability to resolve small scale details and as a consequence nonuniform meshes are often desirable in practice. Since the work of Berger and Oliger (4) on compressible flows, adaptive mesh refinement techniques are quite common (see e.g. (13; 31; 7; 32; 33; 34; 2; 9) and the references therein), but implementations based on the optimal quadtree/octree data structure is less common.

In the case of a standard projection method (see e.g. (8; 6)), the most computationally expensive part comes from solving a Poisson equation for the pressure. This is also the limiting part in terms of accuracy, since high order accurate (and unconditionally stable) semi-Lagrangian methods exist for the convective part. In (26), Popinet proposed a second-order nonsymmetric numerical method to study the incompressible Navier-Stokes equations using an octree data structure. In (22), Losasso et al. proposed a symmetric solution of the Poisson equation for non-graded adaptive grids, i.e. grids for which the size’s ratio between adjacent cells is not constrained. This work relies on the observation that, in the case of the Poisson equation, first-order perturbations in the location of the solution yield consistent schemes (see Gibou et al. (12)). Losasso et al. then extended the work of Lipnikov et al. (19) to the case of arbitrary grids to propose a second-order accurate symmetric discretization of the Poisson equation (21). In (25), Min et al. proposed a second-order accurate scheme for the Poisson equation that also yields second-order accurate gradients. In this case the linear system is nonsymmetric, but diagonally dominant, i.e. for each row the diagonal element is greater or equal to the sum of the nondiagonal elements. This Poisson solver was used for solving the Navier-Stokes equations to second-order accuracy in Min and Gibou (24) using the projection methods described in Brown, Cortez and Minion (6). In this paper, we describe

1 Mechanical Engineering Department & Computer Science Department, University of California, Santa Barbara, CA 93106.
2 Mathematics Department, University of California, Santa Barbara, CA 93106.
3 Mathematics Department, University of California, Santa Barbara, CA 93106.
two finite difference schemes for simulating incompressible flows on nonuniform meshes using quadtree/octree data structures. The first one uses a cell-centered Poisson solver that yields first-order accurate solutions, while producing symmetric linear systems (see Losasso, Gibou and Fedkiw (22)). The second uses a node-based Poisson solver that produces second-order accurate solutions and second-order accurate gradients, while producing nonsymmetric linear systems (see Min, Gibou and Ceniceros (25) for a supra-convergent Poisson solver and Min and Gibou (24) for a second-order accurate Navier-Stokes solver). The grids considered can be non-graded, i.e. the difference of level between two adjacent cells can be arbitrary. In both cases semi-Lagrangian methods are used to update the intermediate fluid velocity in a standard projection framework. We present numerical results in two and three spatial dimensions to complement the analysis of (22; 25; 24).

2 The Navier-Stokes Equations

The motion of fluids is described by the incompressible Navier-Stokes equations for the conservation of momentum and mass:

\[
\begin{align*}
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} &= -\nabla p + \mathbf{f} + \mu \Delta \mathbf{u}, \\
\nabla \cdot \mathbf{u} &= 0,
\end{align*}
\]

where \( \mathbf{u} = (u, v, w) \) is the velocity field, \( \mathbf{f} \) accounts for the external forces such as gravity and where the spatially constant density of the fluid has been absorbed in the pressure \( p \). We assume the viscosity parameter \( \mu \) to be constant.

3 First-Order Accurate Symmetric Navier-Stokes Solver on Octrees

3.1 Cell-Centered Arrangement

In (22), Losasso et al. proposed a solver for the incompressible Euler equations on non-graded adaptive grids. The domain is tiled with cells as depicted in figure 1 and the mesh is refined automatically in order to capture the local details critical to realistic simulations and coarsened elsewhere. An octree data structure is used (see (27)) for efficient processing and the different variables are stored as depicted in figure 1: The velocity components \( u, v \) and \( w \) are stored at the cell faces while the pressure is stored at the center of the cell. This is the standard MAC grid arrangement used in previous works (see e.g. (14)). However, in the case of nonuniform meshes it is more convenient to store the other scalar quantities such as the density \( \rho \) at the nodes of each cell. This stems from the fact that interpolations are more difficult with cell-centered data as discussed in (30).

![Figure 1: Left: the domain is tiled with cells of sizes varying according to the refinement criterion. Right: Zoom of one computational cell. The velocity components \( u, v \) and \( w \) are defined on the cell faces while the pressure \( p \) is defined at the center of the cell. The other scalar quantities are stored at the nodes.](image)

3.2 Projection Method

A standard first-order accurate projection method (8) (see also (6)) is used to solve equations (1) and (2): First an intermediate velocity \( \mathbf{u}^* \) is computed over a time step \( \Delta t \), ignoring the pressure term

\[
\frac{\mathbf{u}^* - \mathbf{u}}{\Delta t} + \mathbf{u} \cdot \nabla \mathbf{u} = \mathbf{f}.
\]

This step, accounting for the convection and the external forces, is followed by a projection step to account for incompressibility and boils down to solving the Poisson equation

\[
\nabla^2 p = \frac{1}{\Delta t} \nabla \cdot \mathbf{u}^*.
\]

Finally, the new velocity field \( \mathbf{u} \) is defined as:

\[
\mathbf{u} = \mathbf{u}^* - \Delta t \nabla p.
\]
Non-Graded Adaptive Grid Approaches to the Incompressible Navier-Stokes Equations

The reader is referred to (22) and the references therein for the application of this scheme to the simulations of free surface flows.

3.2.1 Computing the Intermediate Velocity

The intermediate velocity \( \mathbf{u}^* \) is found by solving equation (3) using a first-order accurate semi-Lagrangian method. In the case of nonuniform grids, the standard high order accurate upwind methods (see e.g. (15; 28; 20)) traditionally used in the case of uniform grids are not well suited due to their stringent time step restrictions and the complexity of their implementations. On the other hand, semi-Lagrangian methods (see e.g. (29)) are unconditionally stable and are straightforward to implement.

The pressure values \( p_1, p_2, p_6, \) and \( p_{10} \) are defined at the center of the cells. \( p_a \) represents a weighted average pressure value. \( p_y \) defines the \( y \) component of the pressure gradient between Cell 1 and Cell 10 defined by standard central differencing. \( \hat{p}_x \) represents the discretization of the \( x \) component of the pressure gradient between Cell 1 and Cell 2, whereas \( p_x \) is a \( O(\Delta x) \) perturbation of \( \hat{p}_x \).

Figure 2: Discretization of the pressure gradient. The pressure values \( p_1, p_2, p_6, \) and \( p_{10} \) are defined at the center of the cells. \( p_a \) represents a weighted average pressure value. \( p_y \) defines the \( y \) component of the pressure gradient between Cell 1 and Cell 10 defined by standard central differencing. \( \hat{p}_x \) represents the discretization of the \( x \) component of the pressure gradient between Cell 1 and Cell 2, whereas \( p_x \) is a \( O(\Delta x) \) perturbation of \( \hat{p}_x \).

3.2.2 The Intermediate Velocity Divergence

Equation (14) is solved by first evaluating the right hand side at every grid point in the domain. Then, a linear system for the pressure is constructed and inverted. Consider the discretization of equation (14) for a large cell with dimensions \( \Delta x, \Delta y \) and \( \Delta z \) neighboring small cells as depicted in figure 1 (left). Since the discretization is closely related to the second vector form of Green’s theorem that relates a volume integral to a surface integral, we first rescale equation (14) by the volume of the large cell to obtain

\[
V_{\text{cell}} \Delta t \nabla^2 p = V_{\text{cell}} \nabla \cdot \mathbf{u}^*.
\]  

The right hand side of equation (6) now represents the quantity of mass flowing in and out of the large cell within a time step \( \Delta t \) in \( m^3 s^{-1} \). This can be further rewritten as

\[
V_{\text{cell}} \nabla \cdot (\mathbf{u}^* - \Delta t \nabla p) = 0.
\]  

This equation implies that the term \( \nabla p \) is most naturally evaluated at the same location as \( \mathbf{u}^* \), namely at the cell faces, and that there is a direct correspondence between the components of the vectors \( \nabla p \) and \( \mathbf{u}^* \). That is, there is a direct correspondence between \( p_x \) and \( u_x \), \( p_y \) and \( v_y \), \( p_z \) and \( w_z \) which live on the right and left faces, top and bottom faces, front and back faces, respectively. Moreover, substituting equation (15) into equation (16) implies \( V_{\text{cell}} \nabla \cdot \mathbf{u} = 0 \) or \( \nabla \cdot \mathbf{u} = 0 \) as desired.

Invoking the second vector form of Green’s theorem, one can write

\[
V_{\text{cell}} \nabla \cdot \mathbf{u}^* = \sum_{\text{faces}} (\mathbf{u}_{\text{face}}^* \cdot \mathbf{n}) A_{\text{face}},
\]  

where \( \mathbf{n} \) is the outward unit normal of the large cell and where \( A_{\text{face}} \) represents the area of a cell face. In the case of figure 1 (left), the discretization of the \( x \)-derivative of the \( x \)-component \( u_x^* \) of the velocity field \( \mathbf{u}^* \) reads

\[
\Delta x \Delta y \Delta z \frac{\partial u_x^*}{\partial x} = u_x^* A_2 + u_x^* A_3 + u_x^* A_4 + u_x^* A_5 - u_x^* A_1,
\]  

where the minus sign in front of \( u_x^* A_1 \) accounts for the fact that the unit normal points to the left. In this example, the discretization of \( \partial u_x^*/\partial x \) amounts to

\[
\frac{\partial u_x^*}{\partial x} = \frac{1}{\Delta x} \left( \frac{u_x^* + u_x^* + u_x^* + u_x^*}{4} - u_x^* \right). 
\]  

The \( y \)- and \( z \)-directions are treated similarly.
3.2.3 Defining the Pressure Derivative to Obtain a Symmetric Linear System

Once, the divergence is computed at the grid nodes, equation (14) is used to construct a linear system of equations for the pressure. Invoking again the second vector form of Green’s theorem, one can write

$$V \text{cell} \cdot (\Delta t \nabla p) = \sum_{\text{faces}} ((\Delta t \nabla p)_{\text{face}} \cdot \mathbf{n}) A_{\text{face}}. \quad (11)$$

Therefore, once the pressure gradient is computed at every face, we can carry out the computation in a similar manner as for the divergence of the velocity described above.

In Gibou et al. (12), we showed that $O(\Delta x)$ perturbations in the location of the solution sampling still yield consistent approximations. This was then exploited in (22) to define $\nabla p$ in order to construct a symmetric linear system. We simply define

$$p_x = \frac{p_2 - p_1}{\Delta},$$

where $\Delta$ can be defined as $\Delta = \Delta x$, which is the size of the large cell or $\Delta = 1/2 \Delta x$, which is the size of the small cell, or as the Euclidean distance between the locations of $p_1$ and $p_2$ or as the distance along the $x$ direction between the locations of $p_1$ and $p_2$. We have used the distance along the $x$ direction between the locations of $p_1$ and $p_2$.

4 Second-Order Accurate Navier-Stokes Solver on Octrees

4.1 Projection Method

In this case, we choose to store all the variable at the grid nodes in order to develop a simple supra-convergent scheme for the Poisson equation as well as a second-order accurate Navier-Stokes solver. Backward differentiation formulas offer a convenient choice to obtain second-order accuracy in time. In this case, the discretization of the momentum equation is written as:

$$\frac{1}{\Delta t} \left( \frac{3}{2} \mathbf{u}^{n+1} - 2 \mathbf{u}_d^n + \frac{1}{2} \mathbf{u}_d^{n-1} \right) + \nabla \mathbf{p}^{n+1} = \mu \Delta \mathbf{u}^{n+1} + g^{n+1}, \quad (12)$$

where $\mathbf{u}_d$ is the velocity at the "departure" point found by tracing back the characteristic curves and interpolated using quadratic interpolation procedures. Equation (12) can be solved using the pressure-free three-step projection method approach of Brown (6): In this method, the intermediate velocity $\mathbf{u}^*$ is first computed by ignoring the pressure component:

$$\frac{1}{\Delta t} \left( \frac{3}{2} \mathbf{u}^* - 2 \mathbf{u}_d^n + \frac{1}{2} \mathbf{u}_d^{n-1} \right) = \mu \Delta \mathbf{u}^* + f^{n+1}. \quad (13)$$

Second, a potential function $\phi^{n+1}$ satisfying the Poisson equation:

$$\Delta_h \phi^{n+1} = \frac{1}{\Delta_t} (\nabla_h \cdot \mathbf{u}^*). \quad (14)$$

is computed to project $\mathbf{u}^*$ onto the divergence free field:

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta_t \alpha \nabla_h \phi^{n+1}, \quad (15)$$

where

$$\alpha = \frac{\mathbf{u}^* \cdot \nabla_h \phi^{n+1}}{\nabla_h \phi^{n+1} \cdot \nabla_h \phi^{n+1}}$$

guarantees that the projection step is an Hodge decomposition at the discrete level (24). The inner product of two functions $f$ and $g$ is computed cell-wise by multiplying the average value for $f \times g$ using the cell’s nodes with the volume of the cell.

Taking the divergence of equation (15) and using the relation given by equation (14) yields a velocity field $\mathbf{u}^{n+1}$ that is indeed divergence free (up to the accuracy of the scheme). The relation between $\phi^{n+1}$ and the $p^{n+1}$ is given by:

$$\nabla p^{n+1} = \frac{3}{2} \nabla \phi^{n+1} - \Delta t \mu \Delta \nabla \phi^{n+1}. \quad (24)$$

The boundary conditions on $\mathbf{u}^*$:

$$\begin{align*}
\mathbf{n} \cdot \mathbf{u}^* |_{\partial \Omega} &= \mathbf{n} \cdot \mathbf{u}^{n+1} |_{\partial \Omega}, \\
\mathbf{t} \cdot \nabla \phi^* |_{\partial \Omega} &= \mathbf{t} \cdot \mathbf{u}^{n+1} \nabla \phi^n + \Delta t \cdot \mathbf{t} \cdot \nabla \phi^n, \\
\nabla \phi \cdot \mathbf{n} |_{\partial \Omega} &= 0,
\end{align*}$$

where $\mathbf{n}$ and $\mathbf{t}$ denote the normal and tangent vectors at the boundary, respectively, are sufficient to ensure second-order accuracy for the velocity field (see (6; 17)).
Figure 3: Example of refinement in two spatial dimensions. The total number of cells increases quadratically whereas the number of locally nonuniform cells (shaded) increases linearly. The contribution of nonuniform cells decreases relatively to that of uniform cells.

4.2 Supra-Convergent Poisson Solver

The Poisson solver presented in section 3.2.3 is globally first-order accurate (consistent), even though the discretization at nonuniform mesh points is inconsistent. In fact, the different approximations of the pressure gradients in (22) result in consistent schemes, regardless of how the distance between the two adjacent cells involved in the discretization of the pressure gradients is accounted for. In this case, the scheme is therefore locally inconsistent on nonuniform meshes but still leads consistent solutions. This was explained by the fact that first-order perturbations in the location produce a consistent method as demonstrated in (11; 12). This can be related to the work of Johansen and Colella (16) who provided a heuristic argument based on potential theory as to why schemes that are only first-order accurate at locally nonuniform grid nodes can be globally second-order accurate (see also the related work by Manteuffel et al. (23) as well as Kreiss et al. (18)). One of the basic reasons is that the set of locally non-uniform cells is one-dimension lower than the set of locally uniform cells (see figure 3). Here, we say that a cell is locally nonuniform if its size is different from the size of at least one of its neighbors whereas a cell is locally uniform if its size is equal to that of all of its neighbors. In turn, the influence of nonuniform cells is absorbed by that of the uniform one through the inversion of the elliptic solver, yielding a second-order accurate scheme. Based on this argument and on numerical evidence, Min et al. hypothesized in (25) that a strategy for deriving $p^{th}$ order accurate finite difference schemes in the $L^\infty$ norm, is to focus on designing schemes that are $(p-1)^{th}$ order accurate at locally nonuniform cells, which reduce to at least $p^{th}$ order accurate schemes at locally uniform cells. In particular, in order to derive second-order accurate schemes, it is enough to focus on finding a consistent approximation at non-uniform cells.

Figure 4: Local structure around a node $v_0$ in a quadtree mesh: At most one node in the two Cartesian directions might not exist. In this case, we define a ghost node (here $v_4$) to be used in the discretizations.

Consider a Cartesian domain $\Omega \in \mathbb{R}^n$ with boundary $\partial \Omega$ and the variable Poisson equation $\nabla \cdot (\rho \nabla u) = f$ on $\Omega$ with Dirichlet boundary condition $u|_{\partial \Omega} = g$. We assume that the variable coeffi-
cient ρ is bounded from below by a positive constant. In one spatial dimension, standard central differencing formulae read:

\[
\left( \frac{u_{i-1} - u_i}{s_{i-\frac{1}{2}}} \cdot \frac{\rho_{i-1} + \rho_i}{2} + \frac{u_{i+1} - u_i}{s_{i+\frac{1}{2}}} \cdot \frac{\rho_{i+1} + \rho_i}{2} \right) \cdot \frac{2}{s_{i-\frac{1}{2}} + s_{i+\frac{1}{2}}} = f_i,
\]

where \( s_{i-\frac{1}{2}} \) is the distance between nodes \( i - 1 \) and \( i \). This discretization is second-order accurate and can be applied in a dimension by dimension framework. However, special care needs to be taken when vertices are no longer aligned (see, e.g. figure 4). In this case, (25) proposed to use the truncation error in linear interpolation in the transverse direction as part of the stencil for the derivative in the other direction, leading to a more compact stencil, and an M-matrix. For example, referring to figure 4 the discretizations for \( (\rho u_x)_x \) and \( (\rho u_y)_y \), along with their Taylor analysis are given respectively by

\[
\left( \frac{u_1 - u_0}{s_1} \cdot \frac{\rho_1 + \rho_0}{2} + \frac{s_6 D_5 + s_4 D_6}{s_5 + s_6} \right) \cdot \frac{2}{s_1 + s_4} = (\rho u_x)_x + \frac{s_5 s_6}{(s_1 + s_4)s_4} (\rho u_y)_y + O(h),
\]

and

\[
\left( \frac{u_2 - u_0}{s_2} \cdot \frac{\rho_2 + \rho_0}{2} + \frac{u_3 - u_0}{s_3} \cdot \frac{\rho_3 + \rho_0}{2} \right) \cdot \frac{2}{s_2 + s_3} = (\rho u_y)_y + O(h),
\]

with

\[
D_5 = \frac{u_5 - u_0}{s_4}, \quad D_6 = \frac{u_6 - u_0}{s_4}.
\]

The spurious term \( \frac{s_5 s_6}{(s_1 + s_4)s_4} (\rho u_y)_y \) is cancelled by weighting appropriately equations (16) and (17) as

\[
\left( \frac{u_1 - u_0}{s_1} \cdot \frac{\rho_1 + \rho_0}{2} + \frac{s_6 a_5 + s_5 a_6}{s_5 + s_6} \right) \cdot \frac{2}{s_1 + s_4} + \left( \frac{u_2 - u_0}{s_2} \cdot \frac{\rho_2 + \rho_0}{2} + \frac{u_3 - u_0}{s_3} \cdot \frac{\rho_3 + \rho_0}{2} \right) \cdot \frac{2}{s_2 + s_3} \left( 1 - \frac{s_5 s_6}{(s_1 + s_4)s_4} \right) = f_0 + O(h).
\]

The discretization obtained is now first-order accurate at locally nonuniform points and second-order accurate at locally uniform points, hence yields a globally second-order accurate scheme in the maximum norm.

5 Numerical Results

We report numerical evidences that confirm the schemes described in section 4 yield second-order accuracy in the \( L^1 \) and \( L^\infty \) norms, on highly irregular grids. In particular the difference of level between cells can be greater that one, illustrating that the method preserves its second-order accuracy on non-graded adaptive meshes. In the case of the Poisson scheme derived in section 4.2, we demonstrate that both the solution and its gradients are second-order accurate. The linear systems of equations are solved using a bi-conjugate gradient method with an incomplete Cholesky preconditioner.

5.1 Accuracy for the Supra-Convergent Poisson Solver

Consider a domain \( \Omega = [0, 2] \times [0, 1] \) and a grid depicted in figure 5 and \( \Delta u = f \) with an exact solution of \( u(x,y) = \sin(x)\cos(y) \). Dirichlet boundary conditions are imposed on the boundary. Tables 1 and 2 demonstrate second-order accuracy in the \( L^1 \) and \( L^\infty \) norms for the solution and its gradient, respectively.
Non-Graded Adaptive Grid Approaches to the Incompressible Navier-Stokes Equations

### 5.2 Accuracy for the Navier-Stokes Equation

#### 5.2.1 Unconditional Stability

Unconditional schemes are not bound to respect the CFL condition $\Delta t < \Delta x_s$, which can lead to very severe time step restriction. Here, we demonstrate that our solver allows unconstrained time steps. Consider a domain $\Omega = \left[ -\pi/2, \pi/2 \right] \times \left[ -\pi/2, \pi/2 \right]$ and a grid depicted in figure 6. We consider the Navier-Stokes equations with an exact solution of:

$$
\begin{align*}
    u(x,y,t) &= -\cos(t) \ast \cos(x) \ast \sin(y), \\
    v(x,y,t) &= \cos(t) \ast \sin(x) \ast \cos(y), \\
    p(x,y,t) &= \sin(x) \ast \sin(y) \ast (\sin(t) - 2 \ast \cos(t)).
\end{align*}
$$

The viscosity is set to $\mu = 1$ and Dirichlet boundary conditions are imposed on the boundary. Tables 3 and 4 demonstrate second-order accuracy in the $L^1$ and $L^\infty$ norms for the solution when the time step is given by $\Delta t = \Delta x_s$ and $\Delta t = 3\Delta x_s$, respectively, where $\Delta x_s$ refers to the size of the most refined grid cell.

#### 5.2.2 Lid-Driven Cavity

We test our Navier-Stokes solver on the well-known lid-driven cavity problem studied extensively by Ghia et al. (10): Consider a domain $\Omega = [0,1]^2$, with the top wall moving with unit velocity. We impose no-slip boundary conditions on the four walls. In this example, we take a Reynolds number $Re = 1000$, i.e. a viscosity coefficient $\mu = 1/1000$. For the refinement criteria, there exist various choices including a posteriori error control based on Richardson extrapolation (3), or a simple criteria based on the magnitude of local vorticity (26; 24). The criterion for mesh refinement we use is that proposed in (5), i.e. a cell

<table>
<thead>
<tr>
<th>Finest Resolution</th>
<th>$L^\infty$ error on $u$</th>
<th>rate</th>
<th>$L^1$ error on $u$</th>
<th>rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$64^2$</td>
<td>$2.73 \times 10^{-3}$</td>
<td></td>
<td>$3.75 \times 10^{-2}$</td>
<td></td>
</tr>
<tr>
<td>$128^2$</td>
<td>$6.99 \times 10^{-4}$</td>
<td>1.91</td>
<td>$4.18 \times 10^{-4}$</td>
<td>2.33</td>
</tr>
<tr>
<td>$256^2$</td>
<td>$1.76 \times 10^{-4}$</td>
<td>1.97</td>
<td>$2.75 \times 10^{-5}$</td>
<td>1.89</td>
</tr>
<tr>
<td>$512^2$</td>
<td>$4.38 \times 10^{-5}$</td>
<td>1.98</td>
<td>$6.87 \times 10^{-6}$</td>
<td>2.05</td>
</tr>
<tr>
<td>$1024^2$</td>
<td>$1.09 \times 10^{-5}$</td>
<td>2.00</td>
<td>$1.71 \times 10^{-6}$</td>
<td>2.00</td>
</tr>
</tbody>
</table>

Table 1: Convergence rate of $u$ for example 5.1.

<table>
<thead>
<tr>
<th>Finest Resolution</th>
<th>$L^\infty$ error on $\nabla u$</th>
<th>rate</th>
<th>$L^1$ error on $\nabla u$</th>
<th>rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$64^2$</td>
<td>$2.28 \times 10^{-1}$</td>
<td></td>
<td>$5.04 \times 10^{-2}$</td>
<td></td>
</tr>
<tr>
<td>$128^2$</td>
<td>$7.63 \times 10^{-2}$</td>
<td>1.58</td>
<td>$1.07 \times 10^{-2}$</td>
<td>1.24</td>
</tr>
<tr>
<td>$256^2$</td>
<td>$2.05 \times 10^{-2}$</td>
<td>1.89</td>
<td>$2.71 \times 10^{-3}$</td>
<td>1.99</td>
</tr>
<tr>
<td>$512^2$</td>
<td>$5.21 \times 10^{-3}$</td>
<td>1.98</td>
<td>$6.06 \times 10^{-4}$</td>
<td>2.16</td>
</tr>
<tr>
<td>$1024^2$</td>
<td>$1.31 \times 10^{-3}$</td>
<td>1.99</td>
<td>$1.46 \times 10^{-4}$</td>
<td>2.04</td>
</tr>
</tbody>
</table>

Table 2: Convergence rate of $\nabla u$ for example 5.1.

Figure 6: Original mesh used in example 5.2.
$C$ is refined whenever

$$\min(\triangle x, \triangle y)^2 \times \max_{x \in C} (|u_{x1}|, |u_{y1}|, |v_{x1}|, |v_{y1}|) > \tau,$$

(18)

where $\tau$ is an empirically chosen threshold taken to be .01. More precisely, consider a grid structure $G^n$ at time $t^n$ on which the velocity field is updated from $u^n$ to $u^{n+1}$. The grid $G^{n+1}$ at $t^{n+1}$ is constructed in the following way: First, we compute the second-order derivatives at every nodes of $G^n$. Second, starting from the root of $G^{n+1}$ split the cell if (18) is satisfied. Finally, $u^{n+1}$ is defined on the new grid $G^{n+1}$ from the values of $u^{n+1}$ on $G^n$ using the quadratic interpolation.

Figure 7 depicts the evolution of the streamlines and the evolution of the adaptive grid until steady state, while figure 8 demonstrates the convergence of the velocity at steady state to the benchmark solution of (10). We note that these simulation results are comparable to the results of (24) that

---

**Table 3:** Convergence rate of the $x$-component $u$ of the velocity field $u$ for example 5.2.1 in the case of a time step $\Delta t = \Delta x_i$.

<table>
<thead>
<tr>
<th>Finest Resolution</th>
<th>$L^\infty$ error on $u$</th>
<th>rate</th>
<th>$L^1$ error on $u$</th>
<th>rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$64^2$</td>
<td>$1.49 \times 10^{-1}$</td>
<td></td>
<td>$2.71 \times 10^{-2}$</td>
<td></td>
</tr>
<tr>
<td>$128^2$</td>
<td>$1.91 \times 10^{-2}$</td>
<td>2.96</td>
<td>$3.57 \times 10^{-3}$</td>
<td>2.92</td>
</tr>
<tr>
<td>$256^2$</td>
<td>$3.86 \times 10^{-3}$</td>
<td>2.31</td>
<td>$7.58 \times 10^{-4}$</td>
<td>2.23</td>
</tr>
<tr>
<td>$512^2$</td>
<td>$2.81 \times 10^{-4}$</td>
<td>3.78</td>
<td>$7.92 \times 10^{-5}$</td>
<td>3.25</td>
</tr>
</tbody>
</table>

**Table 4:** Convergence rate of the $x$-component $u$ of the velocity field $u$ for example 5.2.1 in the case of a time step $\Delta t = 3\Delta x_i$.

<table>
<thead>
<tr>
<th>Finest Resolution</th>
<th>$L^\infty$ error on $u$</th>
<th>rate</th>
<th>$L^1$ error on $u$</th>
<th>rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$64^2$</td>
<td>$7.10 \times 10^{-2}$</td>
<td></td>
<td>$1.89 \times 10^{-2}$</td>
<td></td>
</tr>
<tr>
<td>$128^2$</td>
<td>$1.30 \times 10^{-2}$</td>
<td>2.44</td>
<td>$2.33 \times 10^{-3}$</td>
<td>3.02</td>
</tr>
<tr>
<td>$256^2$</td>
<td>$4.98 \times 10^{-3}$</td>
<td>1.38</td>
<td>$6.72 \times 10^{-4}$</td>
<td>1.79</td>
</tr>
<tr>
<td>$512^2$</td>
<td>$1.29 \times 10^{-3}$</td>
<td>1.98</td>
<td>$1.94 \times 10^{-4}$</td>
<td>2.26</td>
</tr>
</tbody>
</table>

**Table 5:** Accuracy of the velocity field in the $L^1$ and $L^\infty$ norms for example 5.2.3.

| Size of the Finest Grid | $||U − U_h||_1$ | Order | $||U − U_h||_1$ | order |
|-------------------------|-----------------|-------|-----------------|-------|
| $32^2$                  | $6.92E – 2$     | 1     | $2.60E – 2$     | 1     |
| $64^2$                  | $2.64E – 2$     | 1.38  | $9.73E – 3$     | 1.41  |
| $128^2$                 | $6.28E – 3$     | 2.07  | $2.49E – 3$     | 1.96  |
| $256^2$                 | $1.07E – 3$     | 2.54  | $4.98E – 4$     | 2.32  |
| $512^2$                 | $2.23E – 4$     | 2.26  | $3.94E – 5$     | 2.90  |

**Table 6:** Accuracy of the divergence free condition in the $L^1$ and $L^\infty$ norms for example 5.2.3.

| Size of the Finest Grid | $||\nabla \cdot U_h||_1$ | Order | $||\nabla \cdot U_h||_1$ | order |
|-------------------------|--------------------------|-------|--------------------------|-------|
| $32^2$                  | $3.56E – 1$              | 1     | $4.82E – 2$              | 1     |
| $64^2$                  | $1.36E – 1$              | 1.38  | $1.59E – 2$              | 1.60  |
| $128^2$                 | $3.74E – 2$              | 1.87  | $2.30E – 3$              | 2.78  |
| $256^2$                 | $9.55E – 3$              | 1.96  | $2.94E – 4$              | 2.96  |
| $512^2$                 | $2.56E – 3$              | 1.90  | $3.94E – 5$              | 2.90  |
utilized a different refinement criteria, a different CFL number and a different level difference between coarsest and finest cells.

Figure 7: Adaptive grids and streamlines for the driven cavity example 5.2.2. From top to bottom and left to right: $t = 3.12, 7.50, 13.75$ and $37.50$. The coarsest grid has level 6, and the finest has level 8.

5.2.3 Three Spatial Dimensions

In three spatial dimensions, we consider a domain $\Omega = \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]^3$ and a flow with viscosity $\mu = 1$ and with an exact solution defined by:

$$u(x, y, z, t) = -2\cos(t)\cos(x)\sin(y)\sin(z)$$

$$v(x, y, z, t) = \cos(t)\sin(x)\cos(y)\sin(z)$$

$$w(x, y, z, t) = \cos(t)\sin(x)\sin(y)\cos(z)$$

$$p(x, y, z, t) = \frac{1}{4}\cos^2(t)\left(2\cos(2x) + \cos(2y) + \cos(2z)\right)$$

The time step is chosen as $\Delta t = 5 \times \Delta x_s$, where $\Delta x_s$ is the size of the finest grid cell and we run the simulation up to a final time of $t = \pi$. Table 5 demonstrates the second-order accuracy of the velocity field in the $L^1$ and $L^\infty$ norms while table 6 demonstrates the second-order accuracy for the divergence free condition in the $L^1$ and $L^\infty$ norms.

Figure 8: $x$- and $y$- components of the velocity field in the driven cavity example 5.2.2. The domain is $[0, 1]$, $Re = 1000$ and the time step is $\Delta t = 5\Delta x_s$, where $\Delta x_s$ is the size of the smallest grid cell. The symbols are the experimental results of (10), the dotted line depicts the numerical results obtained with an adaptive quadtree with levels ranging from 6 to 8, whereas the solid line depicts the numerical results obtained with an adaptive quadtree with levels ranging from 7 to 9.

6 Conclusion

We have described two finite difference schemes for simulating incompressible flows on nonuniform meshes using quadtree/octree data structures. The first one uses a cell-centered Poisson solver that yields first-order accurate solutions, while producing symmetric linear systems (see Losasso, Gibou and Fedkiw (22)). The second
Figure 9: From top to bottom and from left to right: Arbitrarily generated three dimensional grid used in example 5.2.3, its front view, side view and top view. In particular, note that the difference of level between adjacent grid cells can exceed one.

uses a node-based Poisson solver that produces second-order accurate solutions and second-order accurate gradients, while producing nonsymmetric linear systems (see Min, Gibou and Ceniceros (25) for a supra-convergent Poisson solver and Min and Gibou (24) for a second-order accurate Navier-Stokes solver). The grids considered can be non-graded, i.e. the difference of level between two adjacent cells can be arbitrary, which facilitates grid generations. In both cases semi-Lagrangian methods were used to update the intermediate fluid velocity in a standard projection framework. Numerical results were reported in two and three spatial dimensions to demonstrate the accuracy of the methods.

Acknowledgement: The research of F. Gibou was supported in part by the Alfred P. Sloan Foundation through a research fellowship in Mathematics. The research of H. D. Ceniceros was supported in part by NSF DMS-0311911.

References


Non-Graded Adaptive Grid Approaches to the Incompressible Navier-Stokes Equations


