# A locally refined cut-cell method with exact conservation for the incompressible Navier-Stokes equations

6th European Conference on Computational Mechanics (ECCM 6)
7th European Conference on Computational Fluid Dynamics (ECFD 7)
1115 June 2018, Glasgow, UK

# A LOCALLY REFINED CUT-CELL METHOD WITH EXACT CONSERVATION FOR THE INCOMPRESSIBLE NAVIER-STOKES EQUATIONS

**RENÉ BELTMAN[1], MARTIJN ANTHONISSEN[1], BARRY KOREN[1]**

[1] Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
r.beltman@tue.nl, m.j.h.anthonissen@tue.nl, b.koren@tue.nl

**Key words:** Navier-Stokes, Cut-cell method, Mimetic, Conservative, Cartesian mesh

**Abstract.** We present a mimetic discretization of the incompressible Navier-Stokes equations for general polygonal meshes. The discretization employs staggered velocity variables and results in discrete equations that exactly conserve mass, momentum and kinetic energy (in the inviscid limit) up to and including the boundaries where Dirichlet conditions apply. Moreover, the discrete equations give rise to a discrete global vorticity that is consistent with the Dirichlet boundary conditions.

As the method retains all its favorable properties on general meshes, it can be perfectly applied as a locally refined Cartesian mesh cut-cell method. We numerically verify the conservation properties for the lid-driven cavity flow and demonstrate the method for the unsteady flow around a circular cylinder.

## 1 INTRODUCTION

### 1.1 Low-cost mesh generation: Cartesian cut-cell mesh

In CFD a computationally expensive task may be the discretization of the flow domain into a mesh. In an industrial setting the flow domain is often very complicated and can seldom be discretized by a simple Cartesian mesh. To discretize the domain the engineer may then have to resort to unstructured or curvilinear meshes.

The meshing of a complicated flow domain using an unstructured or curvilinear mesh is computationally expensive and often a bottleneck in the computation. In a lot of applications the geometry of the flow domain is also time-dependent because the flow problem contains moving elements, like for example the turbine blades in a wind-farm simulation. In these problems the mesh may have to be updated after every time step to account for the change of domain geometry. The mesh generation will then take up an even larger part of the total computational time.

An alternative is to use an immersed boundary method. These methods employ just a single Cartesian mesh for the complete flow domain. The complicated boundaries of the domain are immersed in this Cartesian mesh and near such a boundary, in case of viscous flow, the Cartesian method is adjusted for the no-slip boundary condition.

In the class of immersed boundary methods roughly two approaches for modeling the no-slip boundary condition exist. In the first of these approaches the effect of the no-slip boundary condition is indirectly modeled by an extra force term in the Navier-Stokes equations. In the second approach the no-slip boundary condition is directly taken into account by adjusting the numerical method near the no-slip boundaries. Although the second approach is often more tedious, it does allow for a boundary discretization that is still locally conservative with respect to, for example, mass and momentum.

Cartesian cut-cell methods belong to this second approach. In cut-cell methods the flow domain is simply "cut" out of the covering Cartesian mesh. Immediately adjacent to the domain boundaries non-Cartesian cut cells will be created. In these cells the numerical scheme needs to take their non-Cartesian shape and the no-slip condition on one (or more) of its faces into account. The cut cells are polygons in 2D and polyhedrons in 3D, that can have quite a general shape. Especially in 3D there is a lot of variation in the resulting cut cells as they can have a varying number of faces that are again polygons with a varying number of edges.

### 1.2 The method of choice: The MAC method

On Cartesian meshes, one of the most popular numerical methods for incompressible viscous flows is the Marker and Cell method (MAC method) [1]. In the MAC method a staggered mesh is used. Hence, the velocity variables are the normal components of the velocity on the faces of the Cartesian cells and pressure variables are located in the centers of the cells. Using this positioning of the discretized variables and central difference approximations, a very compact discretization scheme for the incompressible Navier-Stokes equations may be derived. Moreover, the staggering of the variables avoids spurious pressure oscillations and exactly conserves the primary quantities momentum and mass, and also secondary quantities like vorticity and energy (in the inviscid limit).

The staggering of the velocity variables makes the treatment of cut-cells difficult. Therefore, most cut-cell methods use a colocated mesh in which the velocity and pressure variables are both located in the centers of the cells. However, methods that use a colocated mesh suffer, in contrast to staggered mesh methods, from pressure oscillations that have to be suppressed. As a consequence, colocated mesh methods also necessarily have a nonzero numerical dissipation. This prevents the discrete conservation of kinetic energy (in the inviscid limit).

Extensions of the MAC method to cut-cell meshes exist [2, 3]. In these approaches the different ways that the Cartesian cells can be cut off are considered and a corresponding numerical scheme is derived for all cases such that the resulting stencil is still a 5-point stencil and the resulting discretization conserves mass, momentum, and energy. However, achieving a fully conservative method using a 5-point stencil is impossible [3]. Moreover, an extension of the method to 3D is problematic due to the many possible cut-cell configurations.

We will here present an extension of the MAC method to polygonal meshes that retains all the favorable properties of the Cartesian MAC method. To formulate this extension we will use recent developments in mimetic discretization methods [4]. This generic approach

has two important advantages. Firstly, the method will be given by general discretization formulas that apply to general cells. As a result we do not have to consider the large variety of cut-cell configurations one-by-one. The resulting method is easy to implement in both 2D and 3D. Secondly, the method can deal with locally refined Cartesian meshes because these are just a special type of polygonal mesh. This allows us to use local refinement near the no-slip boundaries where extra resolution is needed to better resolve boundary layers.

In Section 2 we present the mimetic conservative extension of the MAC method to general polygonal meshes. We focus on a conservative discretization of Dirichlet boundary conditions which is crucial for the method to function as a conservative cut-cell method. In Section 3 we give some numerical evidence to show that the resulting method is indeed conservative also on more general meshes, like refined Cartesian meshes. Finally, in Section 4 we successfully apply the method to the benchmark test of flow around a circular cylinder in a channel [5].

## 2 THE CONSERVATIVE MIMETIC DISCRETIZATION

The many conservation properties and absence of spurious modes in the MAC scheme are a result of the symmetry of the discrete equations and the exact way in which the involved differential operators are discretized. These symmetries are not so apparent from the original presentation of the MAC scheme in [1]. We will therefore represent the method in a way that reflects its symmetry. From this representation it also becomes clear how to extend the method to more general meshes. We focus here on 2D. For 3D see [6].

### 2.1 The MAC method as a dual mesh method

In the reformulation of the MAC method the symmetry of the method is emphasized by its underlying primal-dual mesh structure together with an exact discretization of the differential operators on both the primal mesh and the dual mesh. Before introducing the dual mesh we focus on the discretization of the differential operators on the primal mesh.

We consider the incompressible Navier-Stokes equations in the following form:

$$\partial_t \underline{u} + \nabla \cdot (\underline{u} \otimes \underline{u}) + \nu \nabla \times \underline{\omega} + \nabla p = 0, \tag{1a}$$

$$\nabla \times \underline{u} - \underline{\omega} = 0, \tag{1b}$$

$$\nabla \cdot \underline{u} = 0. \tag{1c}$$

For these equations we discretize the separate differential operators involved, i.e., the gradient, curl and divergence. These discretizations then dictate the position of the discrete variables and the resulting discrete equations.

Let us explain how these operators can be discretized on a mesh without introducing a discretization error yet. The computational mesh consists of vertices, edges and cells. We denote the sets of vertices, edges and cells, respectively, by $\mathcal{C}_{(0)}$, $\mathcal{C}_{(1)}$ and $\mathcal{C}_{(2)}$. Here the subscripts refer to their dimensions. Together these sets constitute our mesh $\mathcal{G} := \{\mathcal{C}_{(0)}, \mathcal{C}_{(1)}, \mathcal{C}_{(2)}\}$, which covers the flow domain $\Omega$.

First consider the incompressibility constraint (1c). To discretize the divergence we integrate (1c) over a cell $\sigma_{(2)} \in \mathcal{C}_{(2)}$ and apply the divergence theorem to obtain

$$0 = \sum_{\sigma_{(1)} \in \partial\sigma_{(2)}} o_{\sigma_{(2)}\sigma_{(1)}} \int_{\sigma_{(1)}} \underline{u} \cdot \underline{n} \, \mathrm{d}L, \tag{2}$$

where the sum is over the edges $\sigma_{(1)}$ that make up the boundary of $\sigma_{(2)}$, $o_{\sigma_{(2)}\sigma_{(1)}}$ is 1 if the orientations of $\sigma_{(1)}$ and $\sigma_{(2)}$ agree and $-1$ otherwise, and $\underline{n}$ is the unit normal vector that points in the direction given by the orientation of $\sigma_{(1)}$. If we take the fluxes through the edges of the mesh as discrete variables, (2) is an exact discretization of the divergence of the velocity field in each cell. We introduce a vector $\boldsymbol{u}^{(1)} = (u^{(1)}_{\sigma_{(1)}})$, with $u^{(1)}_{\sigma_{(1)}} := \int_{\sigma_{(1)}} \underline{u} \cdot \underline{n} \, \mathrm{d}L$, collecting all these fluxes. Let us denote the finite linear space corresponding to unknowns on the edges in $\mathcal{C}_{(1)}$ by $\mathcal{C}^{(1)}$ (thus $\boldsymbol{u}^{(1)} \in \mathcal{C}^{(1)}$), and let us analogously define $\mathcal{C}^{(0)}$ (vertices) and $\mathcal{C}^{(2)}$ (cells). When we introduce the discrete divergence operator $\mathbb{D}^{(2,1)} : \mathcal{C}^{(1)} \to \mathcal{C}^{(2)}$ defined by $\mathbb{D}^{(2,1)}_{\sigma_{(2)}\sigma_{(1)}} := o_{\sigma_{(2)}\sigma_{(1)}}$ (assuming $o_{\sigma_{(2)}\sigma_{(1)}} = 0$ if $\sigma_{(1)} \notin \partial\sigma_{(2)}$), we can write (2) at once for all cells in $\mathcal{C}_{(2)}$ as $\mathbf{0}^{(2)} = \mathbb{D}^{(2,1)}\boldsymbol{u}^{(1)}$ with $\mathbf{0}^{(2)}$ the zero vector in $\mathcal{C}^{(2)}$.

As a second example we discretize the curl of the vorticity in (1a). Recall that in 2D this term is given by $\underline{\mathrm{rot}}\ \omega := (\partial_y\omega, -\partial_x\omega)$, where $\omega$ is a scalar field. We discretize the vorticity at the vertices of the mesh. We collect these values in the vector $\boldsymbol{\omega}^{(0)} = (\omega^{(0)}_{\sigma_{(0)}}) \in \mathcal{C}^{(0)}$. Integrating $\underline{\mathrm{rot}}\ \omega$ over the edges of the mesh and applying the fundamental theorem for line integrals gives $\int_{\sigma_{(1)}} \underline{\mathrm{rot}}\ \omega \cdot \underline{n} \, \mathrm{d}L = \sum_{\sigma_{(0)} \in \partial\sigma_{(1)}} o_{\sigma_{(1)}\sigma_{(0)}} \omega^{(0)}_{\sigma_{(0)}}$, where $o_{\sigma_{(1)}\sigma_{(0)}}$ equals 1 if the orientations of $\sigma_{(1)}$ and $\sigma_{(0)}$ agree and $-1$ otherwise. Defining $\mathbb{D}^{(1,0)} : \mathcal{C}^{(0)} \to \mathcal{C}^{(1)}$ analogously to $\mathbb{D}^{(2,1)}$ we see that

$$\int_{\sigma_{(1)}} \underline{\mathrm{rot}}\ \omega \cdot \underline{n} \, \mathrm{d}L = [\mathbb{D}^{(1,0)}\boldsymbol{\omega}^{(0)}]_{\sigma_{(1)}}. \tag{3}$$

The discrete differential operators $\mathbb{D}^{(1,0)}$ and $\mathbb{D}^{(2,1)}$, also known as incidence matrices, are not sufficient to discretize (1). To finish the reformulation of the MAC method we need a dual mesh. The dual mesh also covers the flow domain $\Omega$ and is geometrically dual to the primal mesh in the following sense. For each $k$-dimensional cell $\sigma_{(k)} \in \mathcal{C}^{(k)}$ there is a $(2-k)$-cell $\tilde{\sigma}_{(k)} \in \tilde{\mathcal{C}}_{(k)}$, with $\tilde{\mathcal{C}}_{(k)}$ the set of $(2-k)$-dimensional dual cells. Note that $\tilde{\sigma}_{(k)}$ has dimension $2-k$. As an example the dual mesh is shown for a square Cartesian mesh in Figure 1.

To show how the dual mesh can be used to complete the discretization we focus again on the diffusive term in the momentum equation (1a). We already saw in (3) how $\underline{\mathrm{rot}}\ \omega$ can be discretized on the edges of the primal mesh in terms of vorticity variables located in the vertices of the primal mesh. Moreover, it was shown how the velocity fluxes through the primal edges can be used to exactly discretize the continuity equation (1c) in every primal mesh cell. The dual mesh can be used to determine the vorticity variables $\boldsymbol{\omega}^{(0)}$ from the velocity variables $\boldsymbol{u}^{(1)}$.

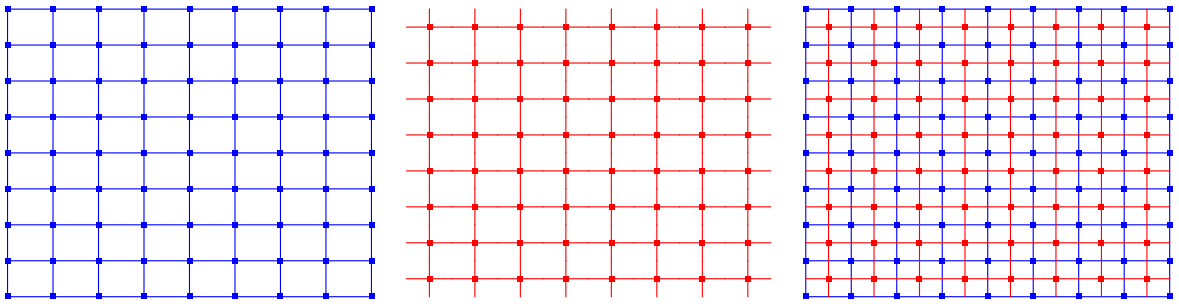

**Figure 1**: On the left the primal mesh is depicted for a square domain, in the middle the corresponding dual mesh is depicted and on the right they are superimposed.

The first step is to interpolate the velocity variables from the edges of the primal mesh to the edges of the dual mesh. We introduce an operator $\mathbb{H}^{(1)} : \mathcal{C}^{(1)} \to \tilde{\mathcal{C}}^{(1)}$ for this. Let us denote $\tilde{\boldsymbol{u}}^{(1)} := \mathbb{H}^{(1)}\boldsymbol{u}^{(1)}$. The entries of $\tilde{\boldsymbol{u}}^{(1)} = (\tilde{u}^{(1)}_{\tilde{\sigma}_{(1)}})$ are an approximation of the line integral of the velocity field over the dual edges, i.e.,

$$\tilde{u}^{(1)}_{\tilde{\sigma}_{(1)}} = \int_{\tilde{\sigma}_{(1)}} \underline{u} \cdot \mathrm{d}\underline{L} + \mathcal{O}(h^l). \tag{4}$$

The order of accuracy $l$ results from the choice of $\mathbb{H}^{(1)}$. In the Cartesian case we can take $\mathbb{H}^{(1)}$ to be the simple diagonal matrix with $\mathbb{H}^{(1)}_{\tilde{\sigma}_{(1)}\sigma_{(1)}} = |\tilde{\sigma}_{(1)}|/|\sigma_{(1)}|$, where $|\tilde{\sigma}_{(1)}|$ and $|\sigma_{(1)}|$ are the lengths of $\tilde{\sigma}_{(1)}$ and $\sigma_{(1)}$, respectively. For uniform Cartesian meshes this interpolation is second order accurate, so $l = 2$ in (4). We can define similar interpolation operators $\mathbb{H}^{(k)} : \mathcal{C}^{(k)} \to \tilde{\mathcal{C}}^{(k)}$ for $k = 0$ and $k = 2$. These can also be chosen diagonal and second order accurate in the Cartesian case. $\mathbb{H}^{(0)}$ interpolates the discretized vorticity field $\boldsymbol{\omega}^{(0)}$ to the dual mesh according to $\tilde{\boldsymbol{\omega}}^{(0)} = \mathbb{H}^{(0)}\boldsymbol{\omega}^{(0)}$. The entries of $\tilde{\boldsymbol{\omega}}^{(0)}$ are approximations of the vorticity field integrated over the 2D dual cells. The interpolation operators $\mathbb{H}^{(k)}$ are known as discrete Hodge operators [7].

Let us consider (1b). By applying the Kelvin-Stokes theorem it follows that $\int_{\tilde{\sigma}_{(0)}} \omega \, \mathrm{d}V = \int_{\tilde{\sigma}_{(0)}} \mathrm{curl}\ \underline{u} \, \mathrm{d}V = \sum_{\tilde{\sigma}_{(1)} \in \partial\tilde{\sigma}_{(0)}} o_{\tilde{\sigma}_{(0)}\tilde{\sigma}_{(1)}} \int_{\tilde{\sigma}_{(1)}} \underline{u} \cdot \mathrm{d}\underline{L}$, where $o_{\tilde{\sigma}_{(0)}\tilde{\sigma}_{(1)}}$ is defined analogously to $o_{\sigma_{(1)}\sigma_{(0)}}$. This shows that when the dual mesh variables $\tilde{\boldsymbol{\omega}}^{(0)}$ and $\tilde{\boldsymbol{u}}^{(1)}$ would not contain an interpolation error, we would have $\tilde{\boldsymbol{\omega}}^{(0)} = \tilde{\mathbb{D}}^{(0,1)}\tilde{\boldsymbol{u}}^{(1)}$, where $\tilde{\mathbb{D}}^{(0,1)}$ is the incidence matrix on the dual mesh containing $o_{\tilde{\sigma}_{(0)}\tilde{\sigma}_{(1)}}$ as entries. Moreover, if we choose the orientation of the dual cells based on the orientation of their corresponding primal cell we have $o_{\tilde{\sigma}_{(0)}\tilde{\sigma}_{(1)}} = o_{\sigma_{(1)}\sigma_{(0)}}$ and it holds that $\tilde{\mathbb{D}}^{(0,1)} = (\mathbb{D}^{(1,0)})^T$.

From the previous discussion it follows that we can discretize the diffusion term as $\nu\mathbb{D}^{(1,0)}\boldsymbol{\omega}^{(0)}$, where $\boldsymbol{\omega}^{(0)}$ is the solution of $\mathbb{H}^{(0)}\boldsymbol{\omega}^{(0)} = \tilde{\mathbb{D}}^{(0,1)}\mathbb{H}^{(1)}\boldsymbol{u}^{(1)}$. On a Cartesian mesh with diagonal discrete Hodge operators the inverse is known and the diffusive term is given by $\nu\mathbb{D}^{(1,0)}(\mathbb{H}^{(0)})^{-1}\tilde{\mathbb{D}}^{(0,1)}\mathbb{H}^{(1)}\boldsymbol{u}^{(1)}$. It can be verified that this is the discretization of the diffusion term from the MAC scheme [1].

In terms of incidence matrices and discrete Hodge operators we discretize (1) as

$$\begin{aligned} \mathbb{H}^{(1)}\partial_t \boldsymbol{u}^{(1)} + \mathbb{C}[\boldsymbol{u}^{(1)}]\boldsymbol{u}^{(1)} + \nu\mathbb{H}^{(1)}\mathbb{D}^{(1,0)}(\mathbb{H}^{(0)})^{-1}\tilde{\mathbb{D}}^{(0,1)}\mathbb{H}^{(1)}\boldsymbol{u}^{(1)} + \tilde{\mathbb{D}}^{(1,2)}\tilde{\boldsymbol{p}}^{(2)} &= 0, \\ \mathbb{D}^{(2,1)}\boldsymbol{u}^{(1)} &= 0, \end{aligned} \tag{5}$$

where $\mathbb{C}[\boldsymbol{u}^{(1)}]\boldsymbol{u}^{(1)}$ is the discretization of the nonlinear convection term and where we use pressure variables located in the vertices of the dual mesh. On a Cartesian mesh, if we use the diagonal Hodge operators and the convection discretization of the MAC scheme, we find that (5) is just a reformulation of the MAC scheme. This reformulation is the perfect starting point to generalize the scheme to general polygonal meshes.

### 2.2 The generalization of the MAC method to polygonal meshes

To generalize the MAC scheme to polygonal meshes the most significant change is in the definition of the discrete Hodge operators. The incidence matrices naturally extend to any type of mesh, because they only depend on the topology of the mesh, i.e., the connectivity between different mesh elements. It is intuitively clear that by assigning orientations to the different elements of the polygonal mesh just like we did in the Cartesian case, we can derive the incidence matrices on the polygonal mesh. For the polygonal mesh we use as a dual mesh the barycentric dual mesh, which is constructed by connecting the barycenters of the cells with barycenters of the cell edges. The construction of the barycentric dual mesh for a randomly locally refined Cartesian mesh is exemplified in Figure 2 on the left.

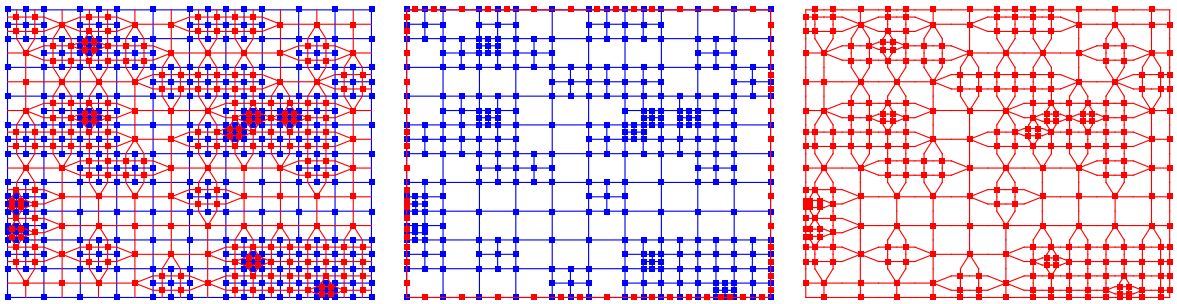

**Figure 2**: On the left the primal mesh (blue) and the barycentric dual mesh (red) are depicted and in the middle the primal mesh and boundary dual mesh are shown. On the right one sees the dual cell-complex which is the union of the interior and boundary dual mesh.

For the polygonal mesh we use as discrete Hodge matrices the mimetic inner product matrices [4, 8]. The Hodge matrices are built cell-wise and are no longer diagonal. They are by construction symmetric positive definite, which is crucial for the stability of the resulting scheme. The Hodge matrices we use have the property that they are diagonal on a Cartesian mesh. As a result, the generalized MAC scheme reduces to the traditional MAC scheme on Cartesian parts of the mesh.

For the convection term we use the discretization proposed in [9]. This is strictly speaking not a generalization of the MAC scheme's convection discretization. Although it has many of the desirable conservation properties of the MAC scheme discretization, it has a wider stencil.

### 2.3 Discretization at the boundary: Extension of dual mesh to cell-complex

To make the method conservative up to and including the boundary of the domain special attention is required. We assume that Dirichlet boundary conditions hold on the boundary. This covers the important cases of inflow and no-slip boundaries.

In the discretization above we used a primal and dual mesh. The primal mesh is a so-called cell-complex, because every $k$-dimensional cell $\sigma_{(k)} \in \mathcal{C}_{(k)}$ has a boundary $\partial\sigma_{(k)}$ that is built-up from lower dimensional cells that are also part of the mesh. Contrastingly, the dual mesh is not a cell-complex, as can be seen, for example, in Figure 1.

To derive a discretization that is conservative up to and including the boundary we extend the dual mesh to a cell-complex. This can be done as follows. The primal mesh is restricted to the boundary of the domain $\partial\Omega$ and the dual mesh to this restriction is calculated within $\partial\Omega$. Subsequently, this boundary dual mesh is joined with the interior dual mesh. The resulting union is a cell-complex. The construction of the dual cell-complex is illustrated for the randomly locally refined Cartesian mesh in Figure 2. To naturally extend the discretization up to the boundary with Dirichlet conditions we extend the incidence matrices on the dual mesh to the complete dual cell-complex. We denote these extensions by replacing the tilde by a bar, i.e., $\bar{\mathbb{D}}^{(k-1,k)}$, with $k = 0, 1$. This extension also leads to extra pressure variables on the boundary of the mesh as we will show in the next paragraph.

### 2.4 The discrete equations

Using the extension of the dual mesh and incidence matrices to the dual cell-complex, we are able to write down the semi-discrete equations for a flow domain with Dirichlet boundary conditions. We first state the equations and then comment on them:

$$\begin{bmatrix} \mathbb{H}^{(1)}\partial_t + \mathbb{C}[\boldsymbol{u}^{(1)}] & \mathbb{H}^{(1)}\mathbb{D}^{(1,0)} & \bar{\mathbb{D}}_i^{(1,2)} \\ \tilde{\mathbb{D}}^{(0,1)}\mathbb{H}^{(1)} & -\nu^{-1}\mathbb{H}^{(0)} & 0 \\ (\bar{\mathbb{D}}_i^{(1,2)})^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^{(1)} \\ \boldsymbol{\omega}^{(0)} \\ \bar{\boldsymbol{p}}^{(2)} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{0}}^{(1)} \\ -\tilde{\mathbb{I}}_b^{(0,0)}\tilde{\boldsymbol{v}}_b^{(0)} \\ \mathbb{I}^{(1,1)}\boldsymbol{v}_b^{(1)} \end{bmatrix}. \tag{6}$$

For each edge of the dual mesh we have a discrete momentum equation in the top line of (6). The subscript on $\bar{\mathbb{D}}_i^{(1,2)}$ indicates the contribution of $\bar{\mathbb{D}}^{(1,2)}$ restricted to the interior edges of the dual mesh. The pressure variables $\bar{\boldsymbol{p}}^{(2)}$ are located in the vertices of the dual cell-complex.

The tangential component of the Dirichlet boundary condition for the velocity is incorporated in the equation defining the vorticity. The vector $\tilde{\boldsymbol{v}}_b^{(0)} \in \tilde{\mathcal{C}}_b^{(0)}$ contains the integrals of the tangential velocity over the boundary dual edges. The boundary term $-\tilde{\mathbb{I}}_b^{(0,0)}\tilde{\boldsymbol{v}}_b^{(0)}$ gives the contribution of boundary edges in the dual cell-complex to the vorticity integral over the dual cells. The entries of $\boldsymbol{\omega}^{(0)}$ are actually the vorticity variables multiplied with the viscosity $\nu$. This definition of the variables increases the symmetry of the system.

The normal-component of the Dirichlet boundary conditions are incorporated in the last line. The matrix $(\bar{\mathbb{D}}_i^{(1,2)})^T$ can be written as $(\bar{\mathbb{D}}_i^{(1,2)})^T = [(\tilde{\mathbb{D}}^{(1,2)})^T\ \tilde{\mathbb{I}}_b^{(1,2)}]^T$, where the first part is the usual incidence matrix on the dual mesh and $\tilde{\mathbb{I}}_b^{(1,2)}$ gives the boundary

contribution of the dual cell-complex. The matrix $\mathbb{I}^{(1,1)}$ is defined such that $(\bar{\mathbb{D}}_i^{(1,2)})^T \boldsymbol{u}^{(1)} = \mathbb{I}^{(1,1)} \boldsymbol{v}_b^{(1)}$ results in $\mathbb{D}^{(2,1)} \boldsymbol{u}^{(1)} = \boldsymbol{0}^{(2)}$ and $(\tilde{\mathbb{I}}_b^{(1,2)})^T \boldsymbol{u}^{(1)} = (\tilde{\mathbb{I}}_b^{(1,2)})^T \boldsymbol{v}_b^{(1)}$, i.e., the discrete incompressibility constraint and the definition of the Dirichlet boundary conditions, imposing that the fluxes for the boundary edges are equal to the prescribed values $\boldsymbol{v}_b^{(1)} \in \mathcal{C}_b^{(1)}$.

More information on these discrete equations and their 3D version can be found in [6].

## 3 EXACT CONSERVATION

The generalization of the MAC method as presented in Section 2 can be used on general polygonal meshes and therefore also on locally refined Cartesian meshes. In this section we will numerically demonstrate that the exact conservation properties of the MAC method also apply for the generalization. To show this we simulate the standard lid-driven cavity flow for Reynolds number $\mathrm{Re} = 1000$. We use this flow because Dirichlet boundary conditions apply on all sides. The viscosity is set equal to $\nu = 10^{-3}$. We use the mesh in Figure 2 with the velocity in the cavity everywhere equal to zero at $t = 0$. We run the simulation till $t = 50$, when a steady state has been reached. The final steady solution is shown in Figure 3 on the left. The purpose of this test is to demonstrate the exact conservation, not to accurately determine the lid-driven cavity flow because for this the mesh is much too coarse.

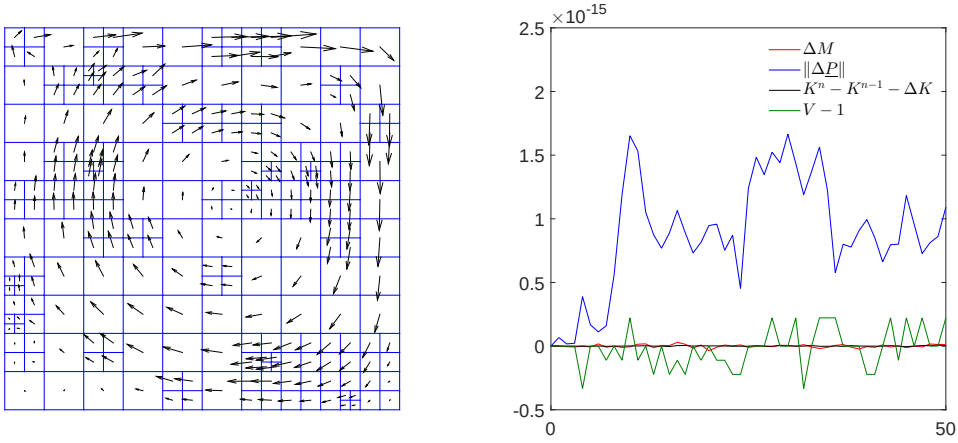


**Figure 3**: On the left: the converged steady velocity field superimposed on the primal mesh. On the right: the conservation of mass, momentum, energy and vorticity numerically verified.

The exactly conserved discrete quantities are mass, momentum and energy. The fluid is incompressible and therefore conservation of mass is expressed by the incompressibility constraint, which is discretized in (6) as $\mathbb{D}^{(2,1)} \boldsymbol{u}^{(1)} = 0$. This relation is exact. The net change of the amount of fluid is proportional to $\Delta M := (\boldsymbol{e}^{(2)})^T \mathbb{D}^{(2,1)} \boldsymbol{u}^{(1)}$, with $\boldsymbol{e}^{(2)} \in \mathcal{C}^{(2)}$ such that $e^{(2)}_{\sigma_{(2)}} = 1$ for all $\sigma_{(2)} \in \mathcal{C}_{(2)}$. Thus $\Delta M$ is the sum of all entries in $\mathbb{D}^{(2,1)} \boldsymbol{u}^{(1)}$. In Figure 3 on the right $\Delta M$ is shown to stay zero up to machine precision.

The global discrete momentum is defined by $P_i := (\boldsymbol{e}_i^{(1)})^T \mathbb{H}^{(1)} \boldsymbol{u}^{(1)}$ for $i = 1, 2$, where $\boldsymbol{e}_i^{(1)}$ is the discretization of the unit vector $\underline{e}_i$ on the primal edges of the mesh, i.e.,

$(e_i^{(1)})_{\sigma_{(1)}} := \int_{\sigma_{(1)}} \underline{e}_i \cdot \underline{n} \, \mathrm{d}L$, where $\underline{n}$ is the normal to $\sigma_{(1)}$. The discrete momentum is conserved and can only change by fluxes over the domain boundary. In the lid-driven cavity the global momentum remains zero over time, because the fluid as a whole remains in the cavity. This also implies that the convective momentum flux over the boundary of the cavity is zero, which is also what we see numerically. The diffusive and pressure terms do give a nonzero boundary flux but their total contributions cancel and the resulting global momentum change $\Delta \underline{P}$ is zero. This is verified in Figure 3.

Similarly, the conserved discrete kinetic energy is defined by $K := \frac{1}{2}(\boldsymbol{u}^{(1)})^T \mathbb{H}^{(1)} \boldsymbol{u}^{(1)}$. This is an, in general first-order, approximation of the global kinetic energy of the flow. In the lid-driven cavity the total energy changes by two mechanisms: a diffusive energy flux over the boundary of the domain and the dissipation in the interior of the cavity. There are no boundary fluxes due to the convective and pressure terms, because on the boundary the normal velocity is zero. This is indeed what we observe numerically. To analyze the energy conservation we use the two-stage energy-conserving Gauss method for time integration. In Figure 3 we see that the change in total energy in every time step is exactly accounted for by the diffusive flux and the dissipation. The difference is again zero up to machine precision.

Finally, we consider the discrete global vorticity, which is given by $V := (\boldsymbol{e}^{(0)})^T \mathbb{H}^{(0)} \boldsymbol{\omega}^{(0)}$, where $\boldsymbol{e}^{(0)} \in \mathcal{C}^{(0)}$ such that $e^{(0)}_{\sigma_{(0)}} = 1$ for all $\sigma_{(0)} \in \mathcal{C}^{(0)}$. In the continuous setting it follows from the definition of the vorticity, the Kelvin-Stokes theorem and the Dirichlet boundary conditions, that the global vorticity equals $\int_\Omega \omega \, \mathrm{d}A = \int_\Omega \nabla \times \underline{u} \, \mathrm{d}A = \int_{\partial\Omega} \underline{u} \cdot \mathrm{d}\underline{L} = 1$. This is also what we find in the discrete setting. In Figure 3 it is shown that the discrete global vorticity $V$ equals 1 up to machine precision.

In [6] proofs of these discrete conservation properties can be found. Further numerical verifications on different polytopal meshes in 2D and 3D are given there as well.

## 4 NUMERICAL TEST: FLOW AROUND A CIRCULAR CYLINDER

To apply the method described above as a cut-cell method we first need to calculate the cut-cell mesh. We illustrate here how this is done in the case of a uniform Cartesian mesh for a square domain containing a cylinder. Subsequently, we apply the method to the benchmark test of an unsteady flow around a cylinder in a channel from [5].

### 4.1 Construction of the cut-cell mesh

We start by discretizing the cylinder. We give the cylinder its own discretization independent of the Cartesian mesh, because in a future extension of the method to time-dependent geometries this is needed for conservation of mass. The boundary of the cylinder is discretized by taking equidistantly distributed points on it and connecting these by straight edges. The discretized cylinder is then immersed in the Cartesian mesh and extra vertices are added where the edges of the cylinder cross the edges of the Cartesian mesh. Finally, the cylinder is cut out of the Cartesian mesh.

Once the primal mesh has been calculated the cut-cell primal-dual cell complex is determined just like in Figure 2. However, the boundary of the mesh contains both the

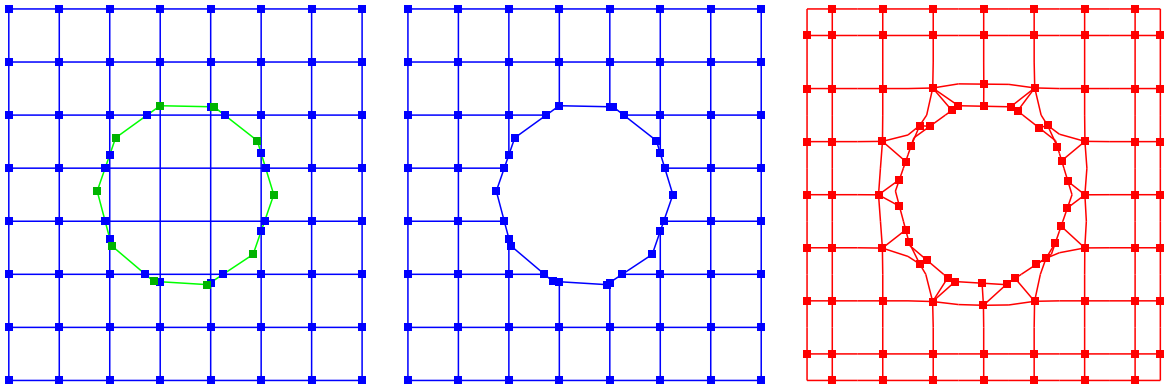

**Figure 4**: On the left: the Cartesian mesh (blue) with the discretized cylinder (green) and the extra added vertices on the cylinder (blue). In the middle: the resulting primal cell-complex. On the right: the resulting dual cell-complex.

boundary of the square and the boundary of the cylinder. In Figure 4 we depict the construction of the dual cell-complex for the cut-cell mesh.

### 4.2 Unsteady flow around cylinder

We test the mimetic cut-cell method on the benchmark problem of flow around an asymmetrically placed circular cylinder in a channel, taken from [5]. At the top and bottom of the channel no-slip boundary conditions apply. We prescribe a parabolic inflow profile and use a no-stress boundary condition for the outflow. To concentrate mesh cells close to the cylinder we use four different refinement levels. An example mesh is given in Figure 5. For time-integration we use the explicit midpoint method. We take the time step small enough for the temporal discretization error to be negligible compared to the spatial discretization error.

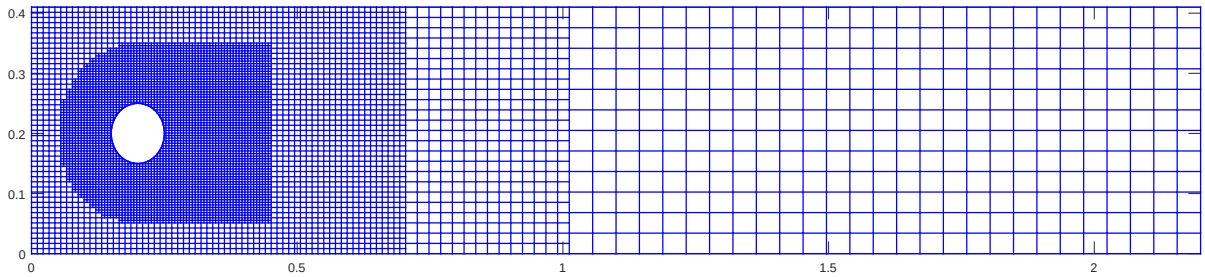


**Figure 5**: This mesh is constructed by starting with a uniform mesh of $N_x \times N_y$ cells and successively refining (by dividing cells in four) in regions increasingly closer to the cylinder. The regions of different refinement levels are kept fixed. The mesh shown here corresponds to $N_x = 50$ and $N_y = 12$.

The inflow velocity corresponds to $\mathrm{Re} = 100$ and after time an unsteady periodic flow develops. We compute for one period the maximum drag coefficient $c_{\mathrm{Dmax}}$, the maximum lift coefficient $c_{\mathrm{Lmax}}$, the Strouhal number $St$ and the pressure difference between front and back of the cylinder after half a period, where we take the start of the period to

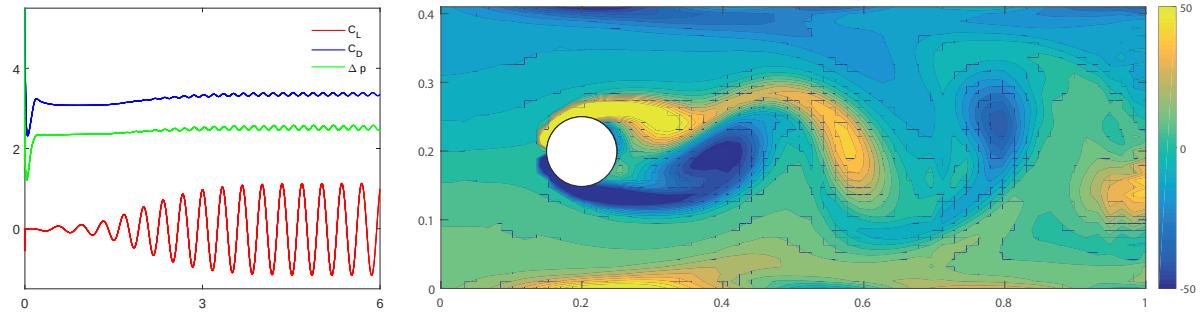


**Figure 6**: On the left $C_{\mathrm{L}}$ (in red), $C_{\mathrm{D}}$ (in blue) and $\Delta p$ are shown as a function of time for the mesh with $N_x \times N_y = 80 \times 20$. On the right the corresponding vorticity field at $t = 6$ is shown.

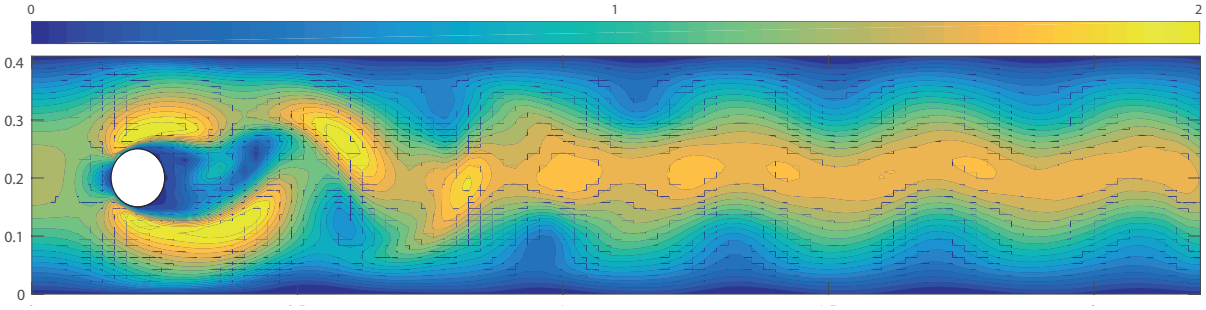


**Figure 7**: The magnitude of the velocity at $t = 6$ on the mesh with $N_x \times N_y = 80 \times 20$.

coincide with $t$ such that $c_{\mathrm{L}}(t) = c_{\mathrm{Lmax}}$.

For the mesh with $N_x \times N_y = 80 \times 20$, the lift and drag coefficients and the pressure difference are shown as a function of time in Figure 6 on the left. We see that from approximately $t = 5$ the flow becomes periodic. On the right in Figure 6 the vorticity field in the first half of the channel is shown. A von Kármán vortex street is formed by the vortex shedding of the cylinder. In Figure 7 the magnitude of the velocity field is plotted for the same mesh and time.

| $N_x \times N_y$ | $\#\boldsymbol{u}^{(1)}$ | $\#\boldsymbol{\omega}^{(1)}$ | $\#\tilde{\boldsymbol{p}}^{(2)}$ | $c_{\mathrm{Dmax}}$ | $c_{\mathrm{Lmax}}$ | $St$ | $\Delta p$ |
|---|---|---|---|---|---|---|---|
| $50 \times 12$ | 14297 | 814 | 7415 | 3.504 | 1.227 | 0.2950 | 2.599 |
| $60 \times 15$ | 21159 | 981 | 10890 | 3.475 | 1.174 | 0.2950 | 2.578 |
| $70 \times 18$ | 29490 | 1127 | 15099 | 3.419 | 1.171 | 0.2963 | 2.576 |
| $80 \times 20$ | 37171 | 1252 | 18979 | 3.393 | 1.129 | 0.2967 | 2.550 |
| $90 \times 23$ | 48619 | 1418 | 24745 | 3.355 | 1.103 | 0.2985 | 2.540 |
| $100 \times 25$ | 58401 | 1534 | 29676 | 3.266 | 1.074 | 0.2994 | 2.517 |
| [5] (lower bound) | - | - | - | 3.22 | 0.99 | 0.295 | 2.46 |
| [5] (upper bound) | - | - | - | 3.24 | 1.01 | 0.305 | 2.50 |

**Table 1**: Results for the benchmark test. The last two lines give a lower and upper bound for the exact values. These bounds come from [5] and are based on the results for many methods collected there.

The values found for $c_{\mathrm{Dmax}}$, $c_{\mathrm{Lmax}}$, $St$ and $\Delta p$ are shown in Table 1. We simulated the flow for 6 different meshes. It can be seen that all four values seem to converge to the range where the exact values lie. Even the finest mesh we used here (with $N_x \times N_y = 100 \times 25$) is still quite coarse. However, the results obtained are in good agreement with the most accurate values found in [5].

## ACKNOWLEDGMENTS

This research is part of the EUROS program, which is supported by NWO domain Applied and Engineering Sciences and partly funded by the Ministry of Economic Affairs.

## REFERENCES

[1] F.H. Harlow, and J.E. Welch, *Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface*, Physics of Fluids **8** (1965), 2182–2189.

[2] M. Dröge, and R. Verstappen, *A new symmetry-preserving Cartesian-grid method for computing flow past arbitrarily shaped objects*, International Journal for Numerical Methods in Fluids **47** (2005), 979–985.

[3] Y. Cheny, and O. Botella, *The LS-STAG method: A new immersed boundary/level-set method for the computation of incompressible viscous flows in complex moving geometries with good conservation properties*, Journal of Computational Physics **229** (2010), 1043–1076.

[4] K. Lipnikov, G. Manzini, and M. Shaskov, *Mimetic finite difference method*, Journal of Computational Physics **257** (2014), 1163–1227.

[5] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher, *Benchmark computations of laminar flow around a cylinder*, Flow Simulation with High-Performance Computers II, Vieweg+ Teubner Verlag (1996), 547–566.

[6] R. Beltman, M.J.H. Anthonissen, and B. Koren, *Conservative polytopal mimetic discretization of the incompressible Navier-Stokes equations*, Journal of Computational and Applied Mathematics, in press https://doi.org/10.1016/j.cam.2018.02.007.

[7] P.B. Bochev, and J.M. Hyman, *Principles of mimetic discretizations of differential operators*, in Compatible Spatial Discretizations, Vol. 142 of The IMA Volumes in Mathematics and its Applications, Springer 2006, pp. 89–119.

[8] J. Bonelle, *Compatible discrete operator schemes on polyhedral meshes for elliptic and Stokes equations*, Dissertation Université Paris-Est, 2014.

[9] B. Perot, *Conservation properties of unstructured staggered mesh schemes*, Journal of Computational Physics **159** (2000), 58–89.