A MIXED HYBRID FINITE VOLUME SCHEME FOR INCOMPRESSIBLE NAVIER-STOKES

M. Oriani, M.Sc.
(ESI Group, France & Queen Mary, University of London, UK)

Dr. G. Pierrot
(ESI Group, France)

Abstract

Mixed Virtual Elements (MVE) is an innovative class of discretization schemes allowing solution of PDEs on virtually any mesh; such schemes stem from the idea of building discrete operators mimicking certain key properties of their continuous counterparts. In our previous work [27] we implemented our own 1st-order MVE scheme for convection-diffusion. In the present work, a) we extend such scheme to formally 2nd-order accuracy, b) we deal with the subsequent stability issues, c) we derive a full formally 2nd-order MVE scheme for incompressible steady-state Navier-Stokes, d) we provide a first suggestion for a MVE N-S solution algorithm. Numerical results are reported for benchmark test cases.

1. Introduction

In the past decade, a new family of PDE discretization schemes has emerged as a promising alternative to classical methods such as Finite Volumes (FV). The aim is to allow for more freedom in the numerical model, most notably in mesh geometry (e.g. the possibility to use strongly non-orthogonal and/or non-convex elements) and anisotropy or discontinuity of material properties. Depending on the author and context, the methodology is known as Mimetic Finite Differences (MFD) or Hybrid Mimetic Mixed method (HMM), and it was recently recast under the umbrella term of Mixed Virtual Elements (MVE) [7].

MVE are fully consistent discretization schemes in which all discrete values “stay true” to their definition, i.e. the approximation is placed on operators rather than on variables. This allows fully controlling the order of accuracy of the method and ensuring its consistency, provided that

discrete operators are indeed constructed up to a given order of consistency.

One may think of MVE as a FV scheme where certain numerical artifacts have been replaced with some desirable features typical of Mixed Finite Volumes (MFV) and Finite Elements (FE). An example is the computation of gradients at mesh faces, which in FV is split into an orthogonal part and a non-orthogonal corrector (NOC): the NOC is typically treated explicitly, thus mixing together discretization scheme and solution algorithm as well as jeopardizing the convergence of the latter – unless a limiting procedure is employed, in which case it will be the gradient consistency to be affected. MVE, on the other hand, treats face-gradients as separate degrees of freedom of the problem, and therefore implicitly and consistently.

In its early developments MVE was aimed at discretizing the pure anisotropic diffusion equation [4,5,11,15,20,24]. Its potential in this case has been largely validated on a number of 2D polygonal and 3D polyhedral meshes, with stress on the fact that requirements on mesh regularity are minimal compared to traditional FV. Such freedom makes MVE appealing in a number of applications featuring complex geometries or physics, most notably the modeling of geological layers in reservoirs, magnetostatic fields or flow through porous media.

Recent attempts have also been made to extend MVE to convection-diffusion-type problems [8,13,28] and to the (1^{st}-order accurate) Navier-Stokes equations [12], thus making it a promising alternative CFD tool towards the solution of large industrial cases, where nowadays one has to respect the mesh regularity restrictions imposed by classical FV or else face inefficient (if not diverging) solution processes.

But the industrial relevance of MVE goes beyond that. In the fast-growing field of numerical optimization, the mesh-independent nature of MVE implies that, when running optimal control or shape optimization algorithms, one needs not worry about grid quality deterioration caused by mesh morphing algorithms (such aspect is highlighted by e.g. [1,2]). More importantly, the intrinsic stability and robustness of MVE may come into play when solving the *adjoint* flow equations, an increasingly popular technique to compute gradients in gradient-based optimization currently facing convergence/robustness issues in its continuous version, and reliability issues in its discrete version; this is in fact what motivated our research in the first place.

In our previous work [27] we derived our own MVE scheme, accurate to 2^{nd}-order for diffusion-type problems and 1^{st}-order for convection-diffusion; we hereby rename our scheme Mixed Hybrid Finite Volumes

(MHFV), which we find more appropriate considering the way we typically approach the issue – by parallelism with classical FV. In the present work, after recalling the key aspects of our method (Section 2), we introduce in Section 3 an extension to 2nd-order accuracy for convective terms and discuss in Section 4 various strategies aimed at dealing with the arising stability issues. Next (Section 5 and 7) we introduce for the first time a 2nd-order accurate MHFV scheme for the incompressible steady-state Navier-Stokes equations. We also derive in Section 6 a MHFV version of the SIMPLEC segregated algorithm for cases where a direct solution is unfeasible – which is expected to always be the case at an industrial level. Validation, numerical results, conclusions and prompts for future work are reported in Section 8 through 12.

2. Principles of Mixed Virtual Elements

We outline in this section, in a simplified fashion, the main theoretical findings leading to the construction of a generic MVE discretization scheme, with particular emphasis on those features that are distinctive of our own approach whilst leaving rigorous demonstrations to previous literature [5,11,20].

In this context, it is customary [11,15,20,22] to start by considering the pure anisotropic diffusion equation:

$$\nabla \cdot (-\mathbb{K}\nabla\varphi) = f \tag{1}$$

where φ is the (scalar) unknown, $\mathbb{K}$ the diffusivity tensor (hence the anisotropy) and f the source term; we omit boundary conditions for simplicity. We rewrite (1) as a system of two 1st-order equations (*mixed formulation*):

$$\begin{cases} \boldsymbol{V} = -\mathbb{K}\nabla\varphi \\ \nabla \cdot \boldsymbol{V} = f \end{cases} \tag{2}$$

where we introduced the vector variable $\boldsymbol{V}$, the (negative) gradient of φ scaled by the diffusivity tensor.

Let us now consider a FV-like discretized domain Ω_h containing n_C cells and n_F faces. We define the discrete space $\boldsymbol{Q_h}$ where we represent scalar variables as cell-averaged (*3-cochains*):

$$\varphi_C = \frac{1}{|C|}\int_C \varphi \, dV \qquad \forall \varphi_C \in \boldsymbol{Q_h} \tag{3}$$

(with $|C|$ being the cell volume), and space $\boldsymbol{X_h}$ where vector-valued variables are represented by their *fluxes* across faces (*2-cochains*):

$$\boldsymbol{V}_{F\leftarrow C} = \int_F \boldsymbol{V}\cdot\boldsymbol{n}_{FC}\, dS \qquad \forall \boldsymbol{V}_{F\leftarrow C} \in \boldsymbol{X_h} \tag{4}$$

where the *"$F \leftarrow C$"* subscript stands for "flux through face F outward w.r.t. cell C" and $\boldsymbol{n}_{FC}$ is the unit normal vector to F outward w.r.t. C. We also introduce the existence of face-averaged discrete scalars (*hybrid* variables):

$$\varphi_F = \frac{1}{|F|}\int_F \varphi\, dS \tag{5}$$

with $|F|$ being the face area.

We now want to define a consistent approximation to the constitutional equation (first equation in system (2)), i.e. some discrete *flux operator* mapping from discrete scalar variables in $\boldsymbol{Q_h}$ to discrete fluxes in $\boldsymbol{X_h}$.

In classical FE, one would typically rewrite the constitutional equation in weak formulation:

$$\int_C \mathbb{K}^{-1}\boldsymbol{V}\cdot\boldsymbol{W}dV = -\int_C \nabla\varphi\cdot\boldsymbol{W}dV \qquad \forall \boldsymbol{W}\in\boldsymbol{X} \tag{6}$$

and subsequently use *shape functions* L_C to reconstruct vector fields $\boldsymbol{V}$ and $\boldsymbol{W}$ inside cell C from their discrete face fluxes. The l.h.s. of (6) would then be discretized as:

$$\int_C \mathbb{K}^{-1} L_C(\boldsymbol{V}_F)_{\partial C}\cdot L_C(\boldsymbol{W}_F)_{\partial C} dV \qquad \forall \boldsymbol{W}_F \in \boldsymbol{X_h} \tag{7}$$

where $(\boldsymbol{V}_F)_{\partial C}$ is intended as the array holding all values $\boldsymbol{V}_{F\leftarrow C}$, i.e. fluxes across each face belonging to cell C.

The core idea of MVE is to construct, for each cell, a local (SPD) operator $\mathbb{M}_C$ such that the following holds:

$$\mathbb{M}_C(\boldsymbol{V}_F)_{\partial C}\cdot(\boldsymbol{W}_F)_{\partial C} = \int_C \mathbb{K}^{-1} L_C(\boldsymbol{V}_F)_{\partial C}\cdot L_C(\boldsymbol{W}_F)_{\partial C} dV \tag{8}$$

In other words, as explained in [6], we want to bypass the step of explicitly defining shape functions L_C, which can be difficult for complex geometries, and equip space $\boldsymbol{X_h}$ with a *discrete scalar product*, defined

on each cell by a matrix $\mathbb{M}_C$ which *implies* the existence of said shape functions (hence the *virtual* nature of the method). More precisely, since $\mathbb{M}_C$ includes for convenience the inverse of the diffusivity tensor $\mathbb{K}$, it is in fact a *material-dependent* scalar product. It is useful to introduce the following notation for such scalar product:

$$[\boldsymbol{V},\boldsymbol{W}]_C^{\boldsymbol{X_h},\mathbb{K}} = \mathbb{M}_C(\boldsymbol{V}_F)_{\partial C}\cdot(\boldsymbol{W}_F)_{\partial C} \tag{9}$$

It has been shown [9] that any $\mathbb{M}_C$ is indeed associated with a linearly consistent reconstruction provided that it satisfies the following two conditions:

1. Local consistency:

$$[\mathbb{K}_C\nabla\varphi^I,\boldsymbol{W}]_C^{\boldsymbol{X_h},\mathbb{K}} + \int_C \varphi^I DIV_C(\boldsymbol{W})dV = \sum_{F\in\partial C}\boldsymbol{W}_{F\leftarrow C}\frac{1}{|F|}\int_F \varphi^I dS \tag{10}$$

   i.e. the Green-Gauss formula is satisfied at a discrete level and must be exact for a linear function φ^I (here DIV_C is some discrete divergence operator which we shall define later, and $\mathbb{K}_C$ is the cell-averaged diffusivity tensor);

2. Stability:

$$s^*\sum_{F\in\partial C}|C|\boldsymbol{W}_{F\leftarrow C}^2 \le [\boldsymbol{W},\boldsymbol{W}]_C^{\boldsymbol{X_h},\mathbb{K}} \le S^*\sum_{F\in\partial C}|C|\boldsymbol{W}_{F\leftarrow C}^2 \tag{11}$$

   i.e. the scalar product shall not vanish nor become unbound.

Several ways have been suggested to compute a suitable $\mathbb{M}_C$ [5,11,20]; we present here our own approach, leading to our MHFV scheme. We start by defining an *average* operator applicable to discrete fluxes:

$$<\boldsymbol{V}>_C = \frac{1}{|C|}\sum_{F\in\partial C}\boldsymbol{V}_{F\leftarrow C}(\boldsymbol{x}_F-\boldsymbol{x}_C) \tag{12}$$

where $\boldsymbol{x}_F$ and $\boldsymbol{x}_C$ are the face and cell centroids, respectively. Then we build a first expression for the scalar product based on (12):

$$[\boldsymbol{V},\boldsymbol{W}]_C^{\boldsymbol{X_h},\mathbb{K},avg} = |C|\mathbb{K}_C^{-1}<\boldsymbol{V}>_C\cdot<\boldsymbol{W}>_C \tag{13}$$

Now, while (13) is evidently linearly consistent, in general it does not satisfy the stability condition; most notably it can give rise to zero-energy modes, as we showed in [27]. Thus we introduce the following *stabilization term*:

$$R_C(\boldsymbol{V},\boldsymbol{W}) = \sum_{F\in\partial C} \lambda_{FC}(\boldsymbol{V}_{F\leftarrow C} - |F| < \boldsymbol{V} >_C \cdot \boldsymbol{n}_{FC})\,(\boldsymbol{W}_{F\leftarrow C} - |F| < \boldsymbol{W} >_C \cdot \boldsymbol{n}_{FC}) \tag{14}$$

where λ_{FC} is some face weight which we shall discuss later. Hence the complete scalar product is expressed as:

$$[\boldsymbol{V},\boldsymbol{W}]_C^{\boldsymbol{X_h},\mathbb{K}} = [\boldsymbol{V},\boldsymbol{W}]_C^{\boldsymbol{X_h},\mathbb{K},avg} + R_C(\boldsymbol{V},\boldsymbol{W}) \tag{15}$$

It is easy to verify that (15) is stable whilst still satisfying the consistency condition, since $R_C(\boldsymbol{V},\boldsymbol{W})$ vanishes when either $\boldsymbol{V}$ or $\boldsymbol{W}$ is the gradient of a linear function.

Having defined a suitable $\mathbb{M}_C$, and introducing now a discrete divergence operator:

$$DIV_C(\boldsymbol{V}) = \frac{1}{|C|}\sum_{F\in\partial C} \boldsymbol{V}_{F\leftarrow C} \tag{16}$$

we can derive the discrete flux operator by injecting (16) and (15) in the discrete Green-Gauss (10) thus yielding, after some manipulation:

$$(\boldsymbol{V}_F)_{\partial C} = \mathbb{M}_C^{-1}(\varphi_C - \varphi_F)_{\partial C} \qquad \forall C \in \Omega_h \tag{17}$$

This is a local discrete form of the constitutional equation in system (2). We observe that, since (17) constitutes a mapping from *1-cochains* (gradients along lines $(\boldsymbol{x}_F - \boldsymbol{x}_C)$) to *2-cochains* (face fluxes), it falls within the definition of *Hodge star* operator.

The Poisson equation (second equation in (2)) is easily discretized via the divergence operator (16):

$$\sum_{F\in\partial C} \boldsymbol{V}_{F\leftarrow C} = |C| f_C \qquad \forall C \in \Omega_h \tag{18}$$

The system is then closed by imposing *flux conservation* across each face:

$$\boldsymbol{V}_{F\leftarrow C^+} + \boldsymbol{V}_{F\leftarrow C^-} = 0 \qquad \forall F \in \Omega_h \tag{19}$$

where C^+ and C^- denote the two cells connected by face F.

In order to assemble and solve the resulting discrete system; we typically favour the following approach:

1. replace flux operator (17) in the discrete Poisson equation (18) to obtain an expression of φ_C in function of $(\varphi_F)_{\partial C}$;

2. re-inject in (17) to eliminate φ_C;

3. impose flux conservation (19) for each face in Ω_h.

This way we eliminate all but the hybrid variables φ_F and we are left to solve a SPD system which scales with n_F:

$$\boldsymbol{\mathcal{A}}_{\mathbb{K}}(\varphi_F)_{F\in\Omega_h} = \left(rhs_F^{f,\mathbb{K}}\right)_{F\in\Omega_h} \tag{20}$$

$\boldsymbol{\mathcal{A}}_{\mathbb{K}}$ thus represents the MHFV Laplacian operator, whereas the r.h.s. stems from the hybridization procedure described above (we omit here its exact expression). Notice that $\boldsymbol{\mathcal{A}}_{\mathbb{K}}$ is a $n_F \times n_F$ SPD sparse matrix whose sparsity pattern is given by the linking of each face F with all faces belonging to its neighbouring cells C^+ and C^-, as in the stencil shown in Figure 1.

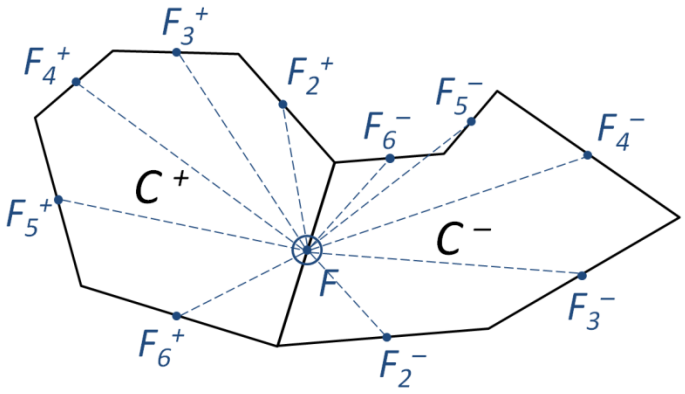


Figure 1: Example of stencil for a MHFV operator on a generic 2D mesh.

Before moving forward it is worth mentioning that, when the scalar product matrix $\mathbb{M}_C$ is derived following the procedure described above, the action of its inverse can be expressed as a combination of two linearly consistent approximate gradients of φ, $\nabla_C^{\mathcal{G}}\varphi$ and $\nabla_C^{\mathcal{L}}\varphi$, based on the Green-Gauss formula and the least-squares approach respectively:

$$\boldsymbol{V}_{F\leftarrow C} = -|F|\mathbb{K}_C\nabla_C^{\mathcal{G}}\varphi\cdot\boldsymbol{n}_{FC} - \frac{1}{\lambda_{FC}}\left\{\varphi_F - \varphi_C - \nabla_C^{\mathcal{L}}\varphi\cdot(\boldsymbol{x}_F - \boldsymbol{x}_C)\right\} \tag{21}$$

We provided a demonstration of such result in [27], also noticing that (21) is similar to the stabilized flux expression found in [28]. The weight $(\lambda_{FC})^{-1}$ comes from the stabilization term (14) and coincides with the

one used for the least-squares gradient reconstruction: $\nabla_C^{\mathcal{L}}\varphi$. Expression (21) also suggests 2nd-order accuracy of the scheme for variables φ_F and φ_C, a result theoretically proven by [4] which we also investigated via numerical experiments in [27].

One has a certain freedom of choice regarding the exact expression of λ_{FC}; considering (15) it is reasonable to demand for it to scale with $(h\,\mathbb{K}_C)^{-1}$, with h being some local characteristic length; on the other hand, as we showed in [27], a certain parallelism can be identified between (21) and the classical FV gradient approximation with Non-Orthogonal Correctors (NOCs, see e.g. [19]), thus suggesting a collection of choices for λ_{FC} based on various NOC expressions. We tested and compared a number of these in [27] and concluded that there was no distinct superiority of one choice over another. In the present work we shall employ a weight of type "over-relaxed" in the form:

$$\lambda_{FC} = \frac{|(\boldsymbol{x}_F - \boldsymbol{x}_C)\cdot \boldsymbol{n}_{FC}|}{|F|\,|\mathbb{K}_C \boldsymbol{n}_{FC}\cdot \boldsymbol{n}_{FC}|} \tag{22}$$

3. MHFV for Convection-Diffusion

It was shown in the previous section how our MHFV scheme is derived for the pure anisotropic diffusion equation. Much more relevant to CFD is the convection-diffusion equation:

$$\nabla\cdot(-\mathbb{K}\nabla\varphi + \boldsymbol{U}\varphi) = f \tag{23}$$

with $\boldsymbol{U}$ being the convective velocity field. Again, we first rewrite it in mixed formulation:

$$\begin{cases} \boldsymbol{V} = -\mathbb{K}\nabla\varphi + \boldsymbol{U}\varphi \\ \nabla\cdot\boldsymbol{V} = f \end{cases} \tag{24}$$

Notice that vector variable $\boldsymbol{V}$ is now a combination of both the (still anisotropic) diffusive flow and the convective flow. Since (17) already provides an approximate relationship between diffusive fluxes and (cell-averaged and hybrid) scalar values for a given cell, we simply need to add a convective term to said fluxes.

Given the definition of convective flux:

$$\boldsymbol{V}_F^{cnv} = \int_F \varphi\boldsymbol{U}\cdot\boldsymbol{n}_{FC}\,d\Sigma \tag{25}$$

it would seem natural to define its discrete counterpart as:

$$\boldsymbol{V}_{F\leftarrow C}^{cnv} = \varphi_F \boldsymbol{\Phi}_{FC} \tag{26}$$

with:

$$\boldsymbol{\Phi}_{FC} = \int_F \boldsymbol{U} \cdot \boldsymbol{n}_{FC} d\Sigma \tag{27}$$

i.e. $\boldsymbol{\Phi}_{FC}$ is the projection of the convective velocity onto space $\boldsymbol{X_h}$ and φ_F is the previously introduced hybrid variable, representing the face-averaged scalar; we shall refer to scheme (26) as *hybrid centered*.

This approach however, as noted by [10], is affected by the very same issues encountered in classical FV centered convective schemes: numerical instabilities manifest themselves when we operate at high Peclet numbers, unless the mesh is sufficiently refined – which is not always feasible, especially for large, industrial cases.

Therefore, much like in FV, in order to stabilize the scheme we can resort to upwinding techniques. The difference is that, as pointed out by [13], we can take the advantage of the existence of the hybrid variable φ_F as a natural unknown of the problem. This is expected to improve accuracy compared to classical FV, which typically deals with cell-averaged scalars only.

Following [10], in our previous work [27] we described a number of centered and 1st-order upwind schemes and implemented a unified framework to handle them. For the purpose of this work we shall only recall what we named the *hybrid 1st-order upwind* strategy, defining the discrete convective flux across face F as:

$$\begin{aligned} &\boldsymbol{V}_{F\leftarrow C}^{cnv} = \boldsymbol{\Lambda}_{FC}\varphi_F + (\boldsymbol{\Phi}_{FC} - \boldsymbol{\Lambda}_{FC})\varphi_C \\ &\quad where \quad \boldsymbol{\Lambda}_{FC} = \min(0, \boldsymbol{\Phi}_{FC}) \end{aligned} \tag{28}$$

That is: the scalar quantity convected across F is taken as the cell-averaged value of φ if C is upwind w.r.t. F, and its face-averaged value (i.e. the hybrid variable) otherwise.

We may now add together convective and diffusive fluxes and rewrite expression (17) for the total fluxes in the form:

$$\begin{aligned} &(\boldsymbol{V}_F)_{\partial C} = \mathbb{N}_C(\varphi_C - \varphi_F)_{\partial C} + (\boldsymbol{\Phi}_{FC})_{\partial C}\varphi_C \qquad \forall C \in \Omega_h \\ &\qquad where \quad \mathbb{N}_C = \mathbb{M}_C^{-1} - \mathrm{diag}(\boldsymbol{\Lambda}_{FC})_{\partial C} \end{aligned} \tag{29}$$

It is possible at this point to proceed as we did for the pure diffusion case, i.e. by eliminating cell-averaged scalars and flux variables via the discrete Poisson equation (18) and flux conservation (19) to assemble a linear system covering the whole mesh Ω_h:

$$\boldsymbol{\mathcal{F}}_{\mathbb{K},\Phi}(\varphi_F)_{F\in\Omega_h} = \left(rhs_F^{f,\mathbb{K},\Phi}\right)_{F\in\Omega_h} \tag{30}$$

Here $\boldsymbol{\mathcal{F}}_{\mathbb{K},\Phi}$ is the MHFV convection-diffusion operator; unlike the Laplacian operator $\boldsymbol{\mathcal{A}}_{\mathbb{K}}$, $\boldsymbol{\mathcal{F}}_{\mathbb{K},\Phi}$ is not symmetric, but it is still non-singular and fairly easy to deal with for a standard linear solver.

Now, as mentioned above, the pure diffusion scheme is 2nd-order accurate for scalar φ; whilst this still holds true for a hybrid centered convection strategy, unfortunately the introduction of 1st-order upwind significantly affects h-convergence properties of the scheme. Moreover, numerical results (see Section 9 and 10) show how solution precision is also degraded due to the phenomenon of *numerical diffusion*, a well-known issue in classical FV.

Hence we introduce here an extension of the MHFV convection-diffusion operator to formally 2nd-order accuracy, partially inspired by [15,28]. In classical FV, a 2nd-order convective scheme typically operates as follows: for a given face F interfacing cells C^+ and C^-, the quantity transported across F is taken as a reconstruction of the face value of φ based on an approximate gradient (e.g. least-squares) evaluated in whichever of the two nodes C^+ and C^- lies upwind w.r.t. F.

We can consider a similar approach in MHFV whilst still taking advantage of the extra degrees of freedom we have available, that is: if C is downwind w.r.t. F, then we identify the transported quantity with the hybrid variable φ_F as in the 1st-order scheme, whereas if we are upwind then we approximate it based on some gradient reconstruction in C. This leads to the following formulation for the total fluxes:

$$\begin{aligned}(\boldsymbol{V}_F)_{\partial C} = {} & \mathbb{N}_C(\varphi_C - \varphi_F)_{\partial C} + (\boldsymbol{\Phi}_{FC})_{\partial C}\varphi_C + \\ & \left\{\boldsymbol{\Theta}_{FC}\,\nabla_C^{\mathcal{L}}\varphi\cdot(\boldsymbol{x}_F - \boldsymbol{x}_C)\right\}_{\partial C} \qquad\qquad \forall C \in \Omega_h\end{aligned} \tag{31}$$

where we introduced for convenience the notation:

$$\begin{aligned}\boldsymbol{\Theta}_{FC} &= \boldsymbol{\Phi}_{FC} - \boldsymbol{\Lambda}_{FC} \\ &\quad i.e. \\ \boldsymbol{\Theta}_{FC} &= \max(0, \boldsymbol{\Phi}_{FC})\end{aligned} \tag{32}$$

The choice of a least-squares gradient in (31) is not arbitrary: the local operator $\mathbb{N}_C$ already contains a term stemming from a least-squares

approximation as part of the MHFV diffusive flux (see (21)); in practice, calculating the contribution of such term to the coefficients of $\mathbb{N}_C$ requires inverting (in 3D) a 3×3 matrix for each cell; hence it is computationally convenient to simply reuse the same approximate gradient, i.e. $\nabla_C^{\mathcal{L}}\varphi$ weighted via $(\lambda_{FC})^{-1}$, and lump together diffusive and convective contributions. To clarify, we rewrite the one-sided flux expression (21) with the addition of a 2nd-order upwind convective term:

$$\boldsymbol{V}_{F\leftarrow C} = -|F|\mathbb{K}_C\nabla_C^{\mathcal{G}}\varphi\cdot\boldsymbol{n}_{FC} + \boldsymbol{\Lambda}_{FC}\varphi_F + \boldsymbol{\Theta}_{FC}\varphi_C - \frac{\varphi_F-\varphi_C}{\lambda_{FC}} + \left(\boldsymbol{\Theta}_{FC}+\frac{1}{\lambda_{FC}}\right)\nabla_C^{\mathcal{L}}\varphi\cdot(\boldsymbol{x}_F-\boldsymbol{x}_C) \tag{33}$$

Compared to classical FV, the convection strategy outlined above brings about some considerable advantages: besides still maintaining the aforementioned freedom on grid geometry, it is also worth noticing that, in MHFV, going from 1st to 2nd-order upwind does not require any modification to local stencils (see Figure 1) as these already suffice to construct a 2nd-order accurate scheme. In classical FV such operation entails a modification to the mesh connectivity; it is common practice to maintain a 1st-order stencil in the system matrix and treat the approximate gradients explicitly, i.e. via *deferred correction* as done for NOCs [19], which in turns not only requires an iterative procedure (unnecessary in MHFV, since we maintain a fully implicit scheme) but may also contribute to an overall convergence slowdown when solving for nonlinear flow equations such as Navier-Stokes.

4. Flux Limiters and Stabilizing Techniques

The usage of approximate gradients in 2nd-order upwind schemes is known to cause certain issues when solving on coarser grids, namely the appearance of nonphysical oscillations in the numerical solution. They manifest themselves especially in proximity of steep gradients or near-discontinuities, due to the fact that reconstructed gradients involve values on both sides of said discontinuities; unfortunately the issue affects MHFV as much as any other method.

A number of strategies have been devised in the past to circumvent the problem. Arguably the most widely employed in FV is *flux limiting*, which operates by expressing the convective flux across face F as:

$$\boldsymbol{V}_{F\leftarrow C}^{cnv,lim} = \boldsymbol{\Lambda}_{FC}\varphi_F + \boldsymbol{\Theta}_{FC}\{\varphi_C + \theta_C\nabla_C^{\mathcal{L}}\varphi\cdot(\boldsymbol{x}_F-\boldsymbol{x}_C)\} \tag{34}$$

i.e. the reconstructed gradient value is limited by factor θ_C ($0 \leq \theta_C \leq 1$) computed such that it prevents the formation of new local extrema in the solution field. There is a vast choice regarding the expression of θ_C; we choose here to investigate the effects of the *Venkatakrishnan limiter* [31], which operates over cell C as follows:

1. find $\delta\varphi_C^{min}$ and $\delta\varphi_C^{max}$, the largest negative and positive values of $(\varphi_{C'} - \varphi_C)$, with C' being neighbour cells of C;

2. use the least-squares gradient to reconstruct face values: $\varphi_{F'} = \varphi_C + \nabla_C^{\mathcal{L}}\varphi \cdot (\boldsymbol{x}_{F'} - \boldsymbol{x}_C) \quad \forall F' \in \partial C$;

3. compute for each F':

$$\theta_{F'} = \begin{cases} \dfrac{\gamma_{F'}^2 + 2\gamma_{F'}}{\gamma_{F'}^2 + \gamma_{F'} + 2} & if\ \varphi_{F'} - \varphi_C \neq 0 \\ 1 & if\ \varphi_{F'} - \varphi_C = 0 \end{cases}$$

$$where:$$

$$\gamma_{F'} = \begin{cases} \dfrac{\delta\varphi_C^{max}}{\varphi_{F'} - \varphi_C} & if\ \varphi_{F'} - \varphi_C > 0 \\ \dfrac{\delta\varphi_C^{min}}{\varphi_{F'} - \varphi_C} & if\ \varphi_{F'} - \varphi_C < 0 \end{cases}$$

4. select $\theta_C = \min(\theta_{F'})_{\mathrm{F}\in\partial\mathrm{C}}$

The procedure is a differentiable version of the *Barth-Jespersen limiter* [3], which ensures that reconstructed face values of φ are bounded by cell values found in the neighbours of C; in other words, it ensures solution *monotonicity*, as explained in [26]. The method is easily translated to MHFV, with the addition that, thanks to the extra degrees of freedom available, we can also think of a second version of the limiter where we limit by face-averaged rather than cell-averaged values, i.e. by redefining $\delta\varphi_C^{min}$ and $\delta\varphi_C^{max}$ respectively as the $\min$ and $\max$ of $(\varphi_F - \varphi_C)_{F\in\partial C}$. To distinguish the two versions, in the sequel we shall refer to them as *cell-bounded* and *face-bounded*.

Limiters effectively eliminate all oscillations in the solution field, enforcing monotonicity, but they are known to affect the order of accuracy. An interesting alternative to flux limiting is the family of so-called *Essentially Non-Oscillatory* and *Weighted Essentially Non-Oscillatory* schemes (ENO and WENO) [18,25]; the core idea of ENO is to compute the approximate gradient based on a local stencil selected such that it does not cross discontinuities; similarly, WENO schemes reconstruct gradients over several stencils and perform a weighted

combination of these, with weights favouring those stencils lying on smoother areas. It has been shown how ENO and WENO schemes manage to restrict the amplitude of nonphysical oscillations in the solution.

A similar concept, although easier to implement on unstructured grids, gives rise to the *Weighted Least-Squares* (WLSQR) approach [16]: rather than tampering with local stencils, WLSQR operates by making use, for least-squares gradient reconstruction, of weights designed to strongly favour those nodes that would constitute the corresponding ENO stencil. Such weights, defined in previous literature for classical FV, are adapted to MHFV as follows:

$$\omega_{FC} = \sqrt{\frac{h^a}{\left|\frac{\varphi_F - \varphi_C}{h}\right|^b + h^c}} \qquad where \qquad h = |\boldsymbol{x}_F - \boldsymbol{x}_C| \tag{35}$$

with exponents $a = -1$, $b = 4$ and $c = -2$ determined empirically based on numerical results (see Section 9). Subsequently the weights λ_{FC} used in (33) are replaced by:

$$\lambda_{FC}^{WLSQR} = \frac{\lambda_{FC}\,(1+\lambda_{FC}\omega_{FC})_{\max}}{1+\lambda_{FC}\omega_{FC}} \tag{36}$$

Notice that multiplying by $(1 + \lambda_{FC}\omega_{FC})_{\max} = 1 + \max(\lambda_{FC}\omega_{FC})_{\partial C}$ entails a normalization procedure. While this is not necessary in classical FV, it is paramount in our case because λ_{FC}^{WLSQR} is not only the (inverse of the) weight used for least-squares gradient reconstruction, but also the scaling factor for the stabilization term (14) defined as having a certain dimensionality that must be maintained. Our choice of normalizing by the maximum value of ω_{FC} implies that the value of $(\lambda_{FC})^{-1}$ will be left unmodified where the solution field is smooth, and reduced for faces placed across steep gradients/discontinuities.

WLSQR fits well within a classical FV framework; similar ideas have also been developed for FE schemes. An example is the *Streamline-Upwind Petrov-Galerkin* (SUPG) strategy [8]. For the purpose of this paper we provide the following simplified interpretation of SUPG:

1. 1st-order upwind schemes can be thought of as centered schemes with added artificial diffusivity, which yields stability;

2. artificial diffusivity in FV is however isotropic, i.e. it acts equally in all directions thus producing excessively diffusive results in the crosswind direction, thus degrading accuracy in convection-dominated problems;

3. rather than upwinding, we can therefore consider a centered scheme with an artificially increased diffusivity acting exclusively in the convective streamline direction.

Our MHFV framework swiftly lends itself to SUPG implementation, as it already encompasses anisotropic diffusivity. More specifically, it requires modifying the cell-averaged diffusivity tensor as follows:

$$\mathbb{K}_C^{SUPG} = \mathbb{K}_C + \tau_C(\boldsymbol{U}_C \otimes \boldsymbol{U}_C) \tag{37}$$

with $\boldsymbol{U}_C$ being the cell-averaged convective velocity and τ_C some SUPG stabilization parameter. The exact expression of τ_C, at least for generic meshes, is rather vague and heuristic in nature. We provide here a definition of our own:

$$\tau_C = \frac{\sum_{F\in\partial C}(\boldsymbol{\Theta}_{FC}|\boldsymbol{x}_F - \boldsymbol{x}_C|^2)}{|C|\,|\boldsymbol{U}_C|^2} \tag{38}$$

Our formulation is derived via the following argument: we want to introduce an amount of streamline dissipation roughly equivalent to that caused by 1st-order upwinding. We first consider the energy associated to the convective term:

$$E = \sum_{C\in\Omega_h}\sum_{F\in\partial C} \boldsymbol{V}^{cnv}_{F\leftarrow C}\,\varphi_C \tag{39}$$

which we can rewrite as:

$$E \approx \sum_{C\in\Omega_h}\sum_{F\in\partial C} \boldsymbol{V}^{cnv}_{F\leftarrow C}\,(\varphi_C - \varphi_F) \tag{40}$$

(the jump from (39) to (40) is fairly legitimate since all φ_F will cancel each other out, apart from boundary values which we neglect since we are only interested in a dimensional analysis). Assuming a 1st-order upwind scheme (28), we have:

$$\begin{aligned} E &\approx \sum_{C\in\Omega_h}\sum_{F\in\partial C}[(\boldsymbol{\Theta}_{FC}\varphi_C + \boldsymbol{\Lambda}_{FC}\varphi_F)(\varphi_C - \varphi_F)] \\ &\approx \sum_{C\in\Omega_h}\sum_{F\in\partial C}[(\boldsymbol{\Theta}_{FC}(\varphi_F + \varphi_C - \varphi_F) + \boldsymbol{\Lambda}_{FC}\varphi_F)(\varphi_C - \varphi_F)] \\ &\approx \sum_{C\in\Omega_h}\sum_{F\in\partial C}[\boldsymbol{\Phi}_{FC}\varphi_F(\varphi_C - \varphi_F) + \boldsymbol{\Theta}_{FC}(\varphi_C - \varphi_F)^2] \end{aligned} \tag{41}$$

Getting rid of $\sum_{F\in\partial C}[\boldsymbol{\Phi}_{FC}\varphi_F(\varphi_C-\varphi_F)]$, which corresponds to a centered (i.e. non-dissipative) approximation to $\int_{\partial C}\varphi\nabla\cdot(\boldsymbol{U}\varphi)$, we can express the added dissipation as:

$$E_{diss}=\sum_{C\in\Omega_h}\sum_{F\in\partial C}[\boldsymbol{\Theta}_{FC}(\varphi_C-\varphi_F)^2] \tag{42}$$

Noticing that $(\varphi_C-\varphi_F)$ is 1st-order equivalent to $\nabla_C\varphi\cdot(\boldsymbol{x}_F-\boldsymbol{x}_C)$, we may manipulate (42) it into the form:

$$E_{diss}=\sum_{C\in\Omega_h}\mathbb{Q}_C\nabla_C\varphi\cdot\nabla_C\varphi \tag{43}$$

with $\mathbb{Q}_C$ being (in 3D) a 3×3 matrix. Since (43) is the diffusion-like expression we want our τ_C to scale with, we take as an indicator of its magnitude the trace of $\mathbb{Q}_C$ divided by the number of spatial dimensions, which can be shown to be:

$$\frac{\mathrm{trace}(\mathbb{Q}_C)}{3}=\sum_{F\in\partial C}(\boldsymbol{\Theta}_{FC}|\boldsymbol{x}_F-\boldsymbol{x}_C|^2) \tag{44}$$

and lastly we divide by $(|C|\,|\boldsymbol{U}_C|^2)$ in order to be consistent dimension-wise with (37), thus yielding (38).

Inspired by the key concepts of both WLSQR and SUPG, we now introduce a novel stabilization technique that we shall name *Upwind Least-Squares* (ULSQR). This new method is based on the same basic idea as SUPG, in the sense that it stabilizes in the streamline direction; however, rather than using a centered scheme with modified diffusivity, we maintain a 2nd-order upwind approximation and we act on the least-squares weights, as WLSQR does, in an attempt to recreate an equivalent of the stabilizing effects of SUPG. More specifically, in the one-sided flux expression (33) we replace weights λ_{FC} with:

$$\lambda_{FC}^{ULSQR}=\frac{\lambda_{FC}}{1+\lambda_{FC}|\boldsymbol{\Lambda}_{FC}|} \tag{45}$$

Notice that, since $\boldsymbol{\Lambda}_{FC}=0$ if F is downwind w.r.t. C, the modification only affects the upwind faces, hence the method's name. In other words, besides using an upwind scheme, we also compute an upwind-biased gradient reconstruction. On an upwind face, such bias is proportional to the dimensionless quantity:

$$Pe_{FC}:=\lambda_{FC}|\boldsymbol{\Lambda}_{FC}| \tag{46}$$

which is evidently a *local Peclet number*. Therefore, since the weight employed in least-squares reconstruction is in fact $\left(\lambda_{FC}^{ULSQR}\right)^{-1}$, it follows that the more the problem is convection-dominated, the stronger the upwind bias will be. We argue that, compared to WLSQR, ULSQR may be just as heuristic but it fits more elegantly within the MHFV scheme, namely because:

1. since Pe_{FC} is dimensionless, expression (45) does not affect the weights' dimension, meaning that no further weight normalization is required;

2. ULSQR is perhaps less empirical, as it is based on quantities directly related to the problem's physics; conversely, our version of WLSQR relies on empirical determination of exponents in (35) based on numerical results that may be case-dependent;

3. like SUPG, ULSQR is also solution-independent, hence it does not affect the fully implicit nature of the MHFV scheme.

5. MHFV for Navier-Stokes

The schemes implemented in the previous sections provide all tools necessary to discretize the steady-state, incompressible Navier-Stokes equations (where we denote by $\boldsymbol{g}$ the vector of body forces):

$$\begin{cases} \boldsymbol{U} \cdot \nabla \boldsymbol{U} - \nu \nabla \cdot \nabla \boldsymbol{U} + \nabla p = \boldsymbol{g} \\ \nabla \cdot \boldsymbol{U} = 0 \end{cases} \tag{47}$$

Let us start with the momentum equations. In each direction x, y, z, they can be seen as a convection-diffusion equation for their respective velocity components u, v, w, with respective components of ∇p acting as source term. In the following we shall take for instance the x-momentum equation, which reads:

$$\nabla \cdot (-\nu \nabla u + \boldsymbol{U} u) = -\nabla^x p + g^x \tag{48}$$

This is effectively equivalent to (23), except for a) the anisotropic diffusivity tensor $\mathbb{K}$ is replaced by the scalar kinematic viscosity of the fluid, ν, and b) the convective flow field is no longer problem data, but rather the unknown $\boldsymbol{U}$ itself.

The l.h.s. of (48) is thus discretized via the previously discussed MHFV convection-diffusion operator $\boldsymbol{\mathcal{F}}_{\nu,\Phi}$ – with the difference that, here, the

convective flow is $\boldsymbol{U}$ itself, implying that $\boldsymbol{\mathcal{F}}_{\nu,\Phi}$ is a nonlinear operator. Away from boundary faces the operator is the same in all three spatial directions, hence the complete momentum operator acting on $\boldsymbol{U}$ can be formally expressed as:

$$\boldsymbol{\mathcal{F}}_{\boldsymbol{U},\nu,\Phi} = \begin{pmatrix} \boldsymbol{\mathcal{F}}_{\nu,\Phi} & 0 & 0 \\ 0 & \boldsymbol{\mathcal{F}}_{\nu,\Phi} & 0 \\ 0 & 0 & \boldsymbol{\mathcal{F}}_{\nu,\Phi} \end{pmatrix} \tag{49}$$

As for the pressure gradient, one could think of approximating it over a cell via classical FV methods (e.g. Green-Gauss) and then treat it as a cell-averaged source term. In this work however we favour an approach more consistent with the methodology outlined in the previous sections, as also suggested by [12]. We assume the discrete pressure to live in space $\boldsymbol{Q_h}$ (cell-averaged scalars), and we write the momentum equation in each direction in mixed formulation as follows:

$$\begin{cases} \boldsymbol{\Psi}_x = -\nu\nabla u + \boldsymbol{U}u + p\boldsymbol{e}^x \\ \nabla \cdot \boldsymbol{\Psi}_x = g^x \end{cases} \tag{50}$$

where $\boldsymbol{e}^x$ is the Cartesian basis vector for the x-direction. Notice that, in integral form, (50) is equivalent to (48) since we have:

$$\int_C \nabla \cdot (p\boldsymbol{e}^x) dV = \int_{\partial C} p\boldsymbol{e}^x \cdot \boldsymbol{n}\, dS = \int_C \nabla^x p\, dV \tag{51}$$

We can therefore define for the discrete momentum equation a "complete flux" (in analogy with (31)) which includes the pressure term:

$$\begin{aligned} \left(\boldsymbol{\Psi}_{x,F}\right)_{\partial C} = {} & \mathbb{N}_C(u_C - u_F)_{\partial C} + (\boldsymbol{\Phi}_{FC})_{\partial C} u_C + \\ & \left\{\boldsymbol{\Theta}_{FC}\, \nabla_C^{\mathcal{L}} u \cdot (\boldsymbol{x}_F - \boldsymbol{x}_C)\right\}_{\partial C} + p_C(|F| n_{FC}^x)_{\partial C} \quad \forall C \in \Omega_h \end{aligned} \tag{52}$$

where $n_{FC}^x = \boldsymbol{e}^x \cdot \boldsymbol{n}_{FC}$ is the x component of $\boldsymbol{n}_{FC}$. Then we apply the divergence operator to discretize the Poisson equation as earlier:

$$\sum_{F \in \partial C} \boldsymbol{\Psi}_{x,F \leftarrow C} = |C| g_C^x \qquad \forall C \in \Omega_h \tag{53}$$

and finally we impose flux conservation across faces:

$$\boldsymbol{\Psi}_{x,F \leftarrow C^+} + \boldsymbol{\Psi}_{x,F \leftarrow C^-} = 0 \qquad \forall F \in \Omega_h \tag{54}$$

(and similarly for the y and z-momentum equations). Now, since the discrete continuity equation reads:

$$\sum_{F\in\partial C} \boldsymbol{\Phi}_{FC} = 0 \qquad\qquad \forall C \in \Omega_h \tag{55}$$

where:

$$\boldsymbol{\Phi}_{FC} = |F|\left(u_F n_{FC}^x + v_F n_{FC}^y + w_F n_{FC}^z\right) \tag{56}$$

it is easily verified how the procedure above gives rise to a system in the form:

$$\begin{pmatrix} \boldsymbol{\mathcal{F}}_{\boldsymbol{U},\nu,\Phi} & \boldsymbol{\mathcal{G}} \\ \boldsymbol{\mathcal{G}}^{\mathrm{T}} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{U}_F \\ p_c \end{pmatrix} = \begin{pmatrix} \boldsymbol{rhs}_F^{\boldsymbol{g},\nu,\Phi} \\ 0 \end{pmatrix} \tag{57}$$

so that the gradient operator acting on the pressure space is the adjoint (transpose) of the divergence operator acting on the velocity space.

Linearization of (57) by freezing convective fluxes $\boldsymbol{\Phi}_{FC}$ in the momentum equations yields the MHFV version of the so-called discrete *Oseen* problem. One may choose an iterative procedure, entailing solution of the whole block-coupled Oseen system (57) followed by an update of $\boldsymbol{\Phi}_{FC}$ via (56). For most industrial cases, however, the saddle-point nature of the block-coupled system – together with its prohibitive size – makes it difficult for a standard linear solver to deal with.

Besides, for average industrial applications, the size of the system matrix in (57) grows to a point where in practice it becomes difficult to stock/handle it efficiently. We discuss in the following section a possible solution strategy.

6. SIMPLEC for MHFV

We begin by reformulating certain concepts typical of classical FV in a way that is useful to our purposes. We consider a generic discrete Oseen problem (here we use the generic subscript h to indicate that we are dealing with discrete variables existing in some finite space):

$$\begin{pmatrix} \mathbb{F}_{\Phi} & \mathbb{G} \\ \mathbb{D} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{U}_h \\ p_h \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{58}$$

Here, and in the sequel, we simplified things a little by assuming that there are no body forces involved, i.e. no source term for the momentum equations; extension to a more generic scenario is however quite straight-forward.

As mentioned above, in general, solving the discrete Navier-Stokes equations requires an iterative procedure where, at each iteration, we first solve system (58) and then update the convection-diffusion operator $\mathbb{F}_{\Phi}$ with new convective flow values (*Picard iteration*). We mentioned how direct solution to the block-coupled system (58) is typically unfeasible, therefore calling for an inner iterative procedure to solve the Oseen problem itself. However, since there is no interest in obtaining the exact Oseen solution at each Picard iteration, the two are typically performed at the same time in a "one-shot" fashion.

To achieve that, classical FV often make use of the ever-popular SIMPLE-like solution algorithms (see e.g. [30]). Despite traditionally being presented as "*segregated algorithms*", highlighting the fact that they solve separately for velocity and pressure, SIMPLE-like strategies can in fact be seen as a way of preconditioning the discrete Oseen problem.

Notice that the system matrix in (58) can be factorized as:

$$\begin{pmatrix} \mathbb{F}_{\Phi} & \mathbb{G} \\ \mathbb{D} & 0 \end{pmatrix} = \begin{pmatrix} \mathbb{F}_{\Phi} & 0 \\ \mathbb{D} & -\mathbb{S} \end{pmatrix} \begin{pmatrix} Id & \mathbb{F}_{\Phi}^{-1}\mathbb{G} \\ 0 & Id \end{pmatrix} \tag{59}$$

where we introduced the so-called *Schur complement*:

$$\mathbb{S} = \mathbb{D}\mathbb{F}_{\Phi}^{-1}\mathbb{G} \tag{60}$$

This suggests an efficient way of preconditioning the Oseen system; in fact, an exact Schur complement would provide an exact preconditioner, i.e. it would allow us to solve (58) in one iteration only. However this would require inverting operator $\mathbb{F}_{\Phi}$, which given its size would be computationally extremely expensive in real-life engineering applications. Hence it is common practice to compute an approximate Schur complement instead:

$$\widehat{\mathbb{S}} = \mathbb{D}\widehat{\mathbb{F}_{\Phi}^{-1}}\mathbb{G} \tag{61}$$

and solve iteratively (relaxing if necessary) as follows:

1. solve $\mathbb{F}_{\Phi^n}\boldsymbol{U}_h^* = -\mathbb{G}p_h^n$ (*predictor step* for velocity)
2. solve $\widehat{\mathbb{S}}\delta p_h = \mathbb{D}\boldsymbol{U}_h^*$ (*pseudo-Laplacian* for pressure increment)
3. update pressure: $p_h^{n+1} = p_h^n + \delta p_h$
4. update velocity: $\boldsymbol{U}_h^{n+1} = \boldsymbol{U}_h^* - \widehat{\mathbb{F}_{\Phi^n}^{-1}}\mathbb{G}\delta p_h$ (*corrector step*)

5. update convective fluxes and assemble new operator $\mathbb{F}_{\Phi^{n+1}}$ (Picard step)

In its most basic implementation, SIMPLE operates by approximating the inverse of $\mathbb{F}_\Phi$ with the inverse of its diagonal:

$$\mathbb{F}_\Phi^{-1} \approx \widehat{\mathbb{F}_\Phi^{-1}} = \left(\mathrm{DIAG}(\mathbb{F}_\Phi)\right)^{-1} \tag{62}$$

Let us rephrase that in terms that the reader may be more familiar with. Assuming a collocated FV scheme is employed, the discrete x-momentum equation for node P can be written in the form:

$$A_{PP}u_P^* + \nabla_P^x p = \sum_{\partial P} A_{PP'}u_{P'}^* \tag{63}$$

(and similar for y and z), where the subscript P' represents any node neighbouring with P. In (63), SIMPLE introduces the approximation:

$$A_{PP}u_P^* - \sum_{\partial P} A_{PP'}u_{P'}^* \approx A_{PP}u_P^* \tag{64}$$

meaning that, as anticipated, when it comes to inverting the convection-diffusion operator we consider central coefficients A_{PP} only, i.e. the diagonal. Lastly, SIMPLE assumes that expression (64) can be "shifted" from cells to faces, typically via averaging procedures on coefficients; this is necessary as face values of u are required in the discrete continuity equation. In matrix form, such action is included in the divergence operator $\mathbb{D}$ in (58).

The popularity of SIMPLE is largely due to the fact that it is easy to implement and, for FV, it works reasonably well in most cases; on the downside, SIMPLE often requires heavily relaxing both momentum equation and pressure correction, making it a rather inefficient algorithm.

A significant improvement upon SIMPLE is SIMPLEC. Adding to (63) some form of implicit relaxation by factor α yields:

$$\alpha A_{PP}(u_P^* - u_P^n) + A_{PP}u_P^* + \nabla_P^x p^n = \sum_{\partial P} A_{PP'}u_{P'}^* \tag{65}$$

Now, if we introduce the approximation: $u_P^* \approx u_{P'}^*\ \forall P'$ (which is reasonable for a smooth enough solution field), and taking into account that any consistent discretization for steady-state convection-diffusion gives $A_{PP} = \sum_{\partial P} A_{PP'}$, we obtain:

$$\alpha A_{PP}(u_P^* - u_P^n) + A_{PP}u_P^* - \sum_{\partial P} A_{PP'}u_{P'}^* \approx \alpha A_{PP}(u_P^* - u_P^n) \tag{66}$$

which suggests the approximation:

$$\mathbb{F}_{\Phi,\alpha}^{-1} \approx \widehat{\mathbb{F}_{\Phi,\alpha}^{-1}} = \frac{1}{\alpha}\left(\mathrm{DIAG}(\mathbb{F}_\Phi)\right)^{-1} \tag{67}$$

In other words, rather than dropping extra-diagonal information, SIMPLEC lumps it all in the central coefficient and, in the steady-state case, yields a pressure equation aimed at correcting the velocity increment $(u_P^* - u_P^n)$ rather than the velocity itself. Barring the addition of implicit relaxation to the predictor step, a SIMPLEC iteration is identical to the one outlined above for SIMPLE; to be correct, one ought to replace $\mathbb{D}\boldsymbol{U}_h^*$ on the r.h.s. of the pressure equation with $\mathbb{D}(\boldsymbol{U}_h^* - \boldsymbol{U}_h^n)$, but the two are equivalent since $\mathbb{D}(\boldsymbol{U}_h^n) = 0$ by definition.

Since SIMPLEC acts on relaxed velocity increments, it does not require relaxing the pressure correction step, and it is found to be 20-30% faster than SIMPLE in many applications.

Now, in order to translate SIMPLE-like algorithms in the context of MHFV, some further considerations are in order. In particular, thanks to the existence of hybrid variables u_F, v_F and w_F, our MHFV divergence operator $\boldsymbol{\mathcal{G}}^{\mathrm{T}}$ can act directly on them and there is no need to "shift" the momentum equation from cell-centered to face-centered. All we have to do is to rewrite our operator $\boldsymbol{\mathcal{F}}_{\boldsymbol{U},\nu,\Phi}$ in a form comparable with (63) when written for a face, i.e. with the hybrid variable u_F taking the role of u_P, and u_{C^+} and u_{C^-}, as well as all hybrid values $u_{F'}$ belonging to the stencil of F (Figure 1), taking the role of neighbours $u_{P'}$.

Let us start by taking the one-sided flux expression (33) for a given velocity component (once again we use u to exemplify and we take a 2nd-order upwind scheme for convection):

$$\begin{aligned}\boldsymbol{\Psi}_{x,F\leftarrow C} = &-|F|\nu_C\nabla_C^{\mathcal{G}}u\cdot\boldsymbol{n}_{FC} + \boldsymbol{\Lambda}_{FC}u_F + \boldsymbol{\Theta}_{FC}u_C - \frac{u_F - u_C}{\lambda_{FC}} \\ &+\left(\boldsymbol{\Theta}_{FC} + \tfrac{1}{\lambda_{FC}}\right)\nabla_C^{\mathcal{L}}u\cdot(\boldsymbol{x}_F - \boldsymbol{x}_C) + |F|p_c n_{FC}^x\end{aligned} \tag{68}$$

We can subsequently write a detailed expression for flux conservation across F:

$$\begin{aligned}
&\Big\{-|F|\nu_{C^+}\nabla^{\mathcal{G}}_{C^+}u\cdot\boldsymbol{n}_{FC^+}+\boldsymbol{\Lambda}_{FC^+}u_F+\boldsymbol{\Theta}_{FC^+}u_{C^+}-\frac{u_F-u_{C^+}}{\lambda_{FC^+}}\\
&\qquad+\left(\boldsymbol{\Theta}_{FC^+}+\frac{1}{\lambda_{FC^+}}\right)\nabla^{\mathcal{L}}_{C^+}u\cdot(\boldsymbol{x}_F-\boldsymbol{x}_{C^+})+|F|p_{c^+}n^x_{FC^+}\Big\}\\
&+\Big\{-|F|\nu_{C^-}\nabla^{\mathcal{G}}_{C^-}u\cdot\boldsymbol{n}_{FC^-}+\boldsymbol{\Lambda}_{FC^-}u_F+\boldsymbol{\Theta}_{FC^-}u_{C^-}-\frac{u_F-u_{C^-}}{\lambda_{FC^-}}\\
&\qquad+\left(\boldsymbol{\Theta}_{FC^-}+\frac{1}{\lambda_{FC^-}}\right)\nabla^{\mathcal{L}}_{C^-}u\cdot(\boldsymbol{x}_F-\boldsymbol{x}_{C^-})+|F|p_{c^-}n^x_{FC^-}\Big\}\\
&\qquad=0
\end{aligned}\tag{69}$$

Now, introducing the following face weight definition:

$$\mu_F:=\frac{2\lambda_{FC^+}\lambda_{FC^-}}{\lambda_{FC^+}+\lambda_{FC^-}}\tag{70}$$

and a *local Reynolds number*:

$$\begin{aligned}
Re_F&:=\frac{\mu_F|\boldsymbol{\Phi}_F|}{2}\\
&where\\
|\boldsymbol{\Phi}_F|&=|\boldsymbol{\Phi}_{FC^+}|=|\boldsymbol{\Phi}_{FC^-}|
\end{aligned}\tag{71}$$

we obtain from (69), after some manipulation:

$$\begin{aligned}
&\frac{2(1+Re_F)}{\mu_F}u_F-|F|n^x_{FC^+}(p_{C^+}-p_{C^-})=\\
&\left(\boldsymbol{\Theta}_{FC^+}+\frac{1}{\lambda_{FC^+}}\right)\left[u_{C^+}+\ \nabla^{\mathcal{L}}_{C^+}u\cdot(\boldsymbol{x}_F-\boldsymbol{x}_{C^+})\right]+\\
&\left(\boldsymbol{\Theta}_{FC^-}+\frac{1}{\lambda_{FC^-}}\right)\left[u_{C^-}+\ \nabla^{\mathcal{L}}_{C^-}u\cdot(\boldsymbol{x}_F-\boldsymbol{x}_{C^-})\right]-\\
&\qquad|F|\left(\nu_{C^+}\nabla^{\mathcal{G}}_{C^+}u-\nu_{C^-}\nabla^{\mathcal{G}}_{C^-}u\right)\cdot\boldsymbol{n}_{FC^+}
\end{aligned}\tag{72}$$

The parallelism between (72) and (63) is now evident; in particular we can easily check that, as expected for steady-state, the relationship $A_{PP}=\sum_{\partial P}A_{PP'}$ holds (the reader may verify that).

Therefore (72) suggests a possible implementation of a MHFV-adapted SIMPLE iteration. However, since SIMPLEC is expected to perform better, we choose to proceed towards a SIMPLEC-like formulation. Bearing in mind the reasoning we just went through, it makes sense to apply the following scaling to the MHFV convection-diffusion operator:

$$\widetilde{\boldsymbol{\mathcal{F}}}_{\nu,\Phi,\alpha}=\boldsymbol{\mathcal{F}}_{\nu,\Phi}+\alpha\,\mathrm{diag}\left(\frac{2(1+Re_F)}{\mu_F}\right)_{F\in\Omega_h}\tag{73}$$

with α being the previously introduced momentum relaxation factor, typically ranging between 0.1 and 0.3. Notice that (73) is effectively a form of *inertial relaxation*, its strength being proportional to the local Re_F.

We now have all we need to outline the steps of an iteration of a MHFV SIMPLEC solution strategy:

1. solve (relaxed) momentum:

$$\widetilde{\boldsymbol{\mathcal{F}}}_{\boldsymbol{U},\nu,\Phi^n,\alpha}\boldsymbol{U}_F^* = -\boldsymbol{\mathcal{G}}p_C^n + \alpha\,\mathrm{diag}\left(\frac{2\left(1+Re_F^n\right)}{\mu_F}\right)_{F\in\Omega_h}\boldsymbol{U}_F^n$$

2. approx. inverse: $\widehat{\boldsymbol{\mathcal{F}}^{-1}_{\nu,\Phi^n,\alpha}} = \mathrm{diag}\left(\frac{\mu_F}{2\alpha\left(1+Re_F^n\right)}\right)_{F\in\Omega_h}$

3. solve pseudo-Laplacian: $\boldsymbol{\mathcal{G}}^{\mathrm{T}}\widehat{\boldsymbol{\mathcal{F}}^{-1}_{\boldsymbol{U},\nu,\Phi^n,\alpha}}\boldsymbol{\mathcal{G}}\,\delta p_C = \boldsymbol{\mathcal{G}}^{\mathrm{T}}\boldsymbol{U}_F^*$

4. correct pressure: $p_C^{n+1} = p_C^n + \delta p_C$

5. correct velocity: $\boldsymbol{U}_F^{n+1} = \boldsymbol{U}_F^* - \widehat{\boldsymbol{\mathcal{F}}^{-1}_{\boldsymbol{U},\nu,\Phi^n,\alpha}}\boldsymbol{\mathcal{G}}\,\delta p_C$

6. update conv. fluxes and assemble new operator $\widetilde{\boldsymbol{\mathcal{F}}}_{\nu,\Phi^{n+1},\alpha}$

It should also be mentioned that, depending on the type of boundary conditions imposed, the iteration may also include a pressure normalization step.

7. 2nd-Order Scheme for Pressure

We previously made the remark that, when freezing convective fluxes, the discrete momentum equations can be interpreted as convection-diffusion equations, one for each spatial dimension, where the unknown scalar is the corresponding velocity component and the pressure gradient acts as source term. More specifically, by looking for instance at the complete flux expression for u (68) one can notice how the cell-averaged pressure can be thought of as a piecewise-constant vector source term: $p_C\boldsymbol{e}^x$ which, when projected onto space $\boldsymbol{X_h}$, becomes a source term for the one-sided flux: $|F|p_C n_{FC}^x$.

Such term stems from approximating $\int_C \nabla\cdot(p\boldsymbol{e}^x) \approx p_C\sum_{F\in\partial C}|F|n_{FC}^x$; it involves only cell-averaged values p_C, and it is only 1st-order accurate. We are now interested in increasing the order of accuracy on the

pressure space, namely by replacing the source term on the flux with a better approximated value. This is simply done by approximating the pressure as piecewise-linear rather than piecewise-constant, i.e. by reconstructing its values at faces via some approximate gradient (here we choose a least-squares approach). Thus the discrete pressure term becomes:

$$\int_C \nabla \cdot (p\boldsymbol{e}^x) \approx \sum_{F \in \partial C} \left[p_C + \nabla_C^{\mathcal{L}} p \cdot (\boldsymbol{x}_F - \boldsymbol{x}_C)\right] |F| n_{FC}^x \tag{74}$$

Injecting (74) in the one-sided complete flux expression for the x-momentum equation gives:

$$\begin{aligned} \boldsymbol{\Psi}_{x,F\leftarrow C} = & -|F|\nu_C \nabla_C^{\mathcal{G}} u \cdot \boldsymbol{n}_{FC} + \boldsymbol{\Lambda}_{FC} u_F + \boldsymbol{\Theta}_{FC} u_C \; - \\ & \frac{u_F - u_C}{\lambda_{FC}} + \left(\boldsymbol{\Theta}_{FC} + \frac{1}{\lambda_{FC}}\right) \nabla_C^{\mathcal{L}} u \cdot (\boldsymbol{x}_F - \boldsymbol{x}_C) + \\ & |F|\left[p_C + \nabla_C^{\mathcal{L}} p \cdot (\boldsymbol{x}_F - \boldsymbol{x}_C)\right] n_{FC}^x \end{aligned} \tag{75}$$

The idea is fairly straight-forward, but some remarks are in order: firstly, the addition of $\nabla_C^{\mathcal{L}} p$ to the flux requires a modification to the stencil locally used for pressure. Whilst in the 1st-order case the gradient operator on the pressure space ($\boldsymbol{\mathcal{G}}$ in (57)) for a given face F only involved values p_{C^+} and p_{C^-}, it now requires an extended connectivity which also includes all neighbouring cells of C^+ and C^-, much like a typical FV 2nd-order stencil; the resulting operator is no longer adjoint to the divergence operator acting on velocity.

Extending connectivity, however, is in practice not an issue when using SIMPLE-like algorithms, since pressure is treated explicitly in the momentum equations anyway. The pressure correction equation, which only serves as an update in the algorithm, can be left unmodified without any significant loss in performance.

Secondly, when using SIMPLE or SIMPLEC coupled with a 2nd-order pressure scheme, one needs to take special care with the assembly of the MHFV convection-diffusion operator: since we use the discrete Poisson equation (53) to obtain u_C in function of u_F, and since pressure is treated explicitly (i.e. moved to the r.h.s. of (69)), we end up with an extra term on the r.h.s.: $\sum_{F\in\partial C} |F| \nabla_C^{\mathcal{L}} p \cdot (\boldsymbol{x}_F - \boldsymbol{x}_C) n_{FC}^x$ which must be accounted for when eliminating u_C; from a purely computational viewpoint, it can be treated as a quantity added to the cell-integrated source term in the momentum equation. This was not an issue in the 1st-order pressure scheme because $\sum_{F\in\partial C} p_C |F| n_{FC}^x$ is null, hence in that case p_C did not appear at all in the discrete Poisson equation.

It is also worth mentioning that, when rewriting (72) with the addition of a 2nd-order pressure term, the resulting equation strongly resembles what is known in FV as *Rhie-Chow interpolation*. This is traditionally presented as the addition of pressure smoothing terms when shifting momentum equations from cells to faces, with the purpose of avoiding checkerboard modes. We interpret the fact that Rhie-Chow arises naturally in the MHFV framework as a further indicator of the method's intrinsic robustness and soundness.

8. Validation of 2nd-Order Convective Scheme

MVE schemes for anisotropic diffusion and 1st-order upwind convection have been largely validated in previous literature [5,10,13,24], including our own MHFV implementation [27]. For the purpose of this paper we move on to validating the 2nd-order MHFV convective scheme, a more peculiar aspect of the present work.

Validation is performed via a h-convergence study: we solve, over the 2D square domain $\Omega = [0,1]^2$, the convection-diffusion equation (23) with a source term f calculated such that the exact solution is:

$$\varphi_{ex}(x,y) = 2x^2 + \cos(2\pi x y^2)$$

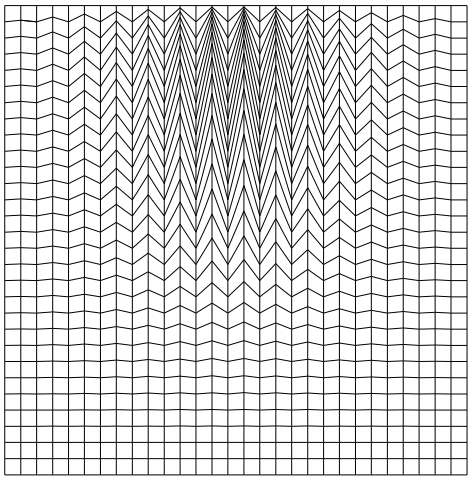


Figure 2: Type of mesh used for validating the 2nd-order accurate MHFV convection-diffusion scheme.

We apply Dirichlet boundary conditions (BCs) everywhere. We assume isotropic diffusivity $\nu = 10^{-5}$ and a conservative convective velocity field:

$$\boldsymbol{U}(x,y) = \begin{pmatrix} 10x + 2 \\ 3x - 10y \end{pmatrix}$$

giving a domain-averaged Peclet number $Pe \approx 8.5 \times 10^5$. As for the mesh, we use grids featuring a number of purposely distorted, non-orthogonal cells (see Figure 2) and we take as a measure of mesh refinement h the maximum face area found in the mesh.

Errors are measured in $\boldsymbol{L}^2$-norm on cell-averaged variables:

$$E = \sqrt{\frac{\sum_{C\in\Omega_h} |C| \left(\varphi_C - \varphi_{ex}(\boldsymbol{x}_C)\right)^2}{\sum_{C\in\Omega_h} |C| \left(\varphi_{ex}(\boldsymbol{x}_C)\right)^2}}$$

We report in Figure 3 results for both the 1st and 2nd-order upwind schemes, compared with their respective expected slope. Results clearly confirm that the scheme is indeed 2nd-order accurate and that it brings about a significant improvement in terms of accuracy compared to 1st-order upwinding. We also show how the independence of MHFV from "mesh quality" parameters such as skewness and orthogonality, which has been vastly verified in MVE literature for purely diffusive and 1st-order upwind convective schemes, is not affected and still holds when increasing the order of the scheme.

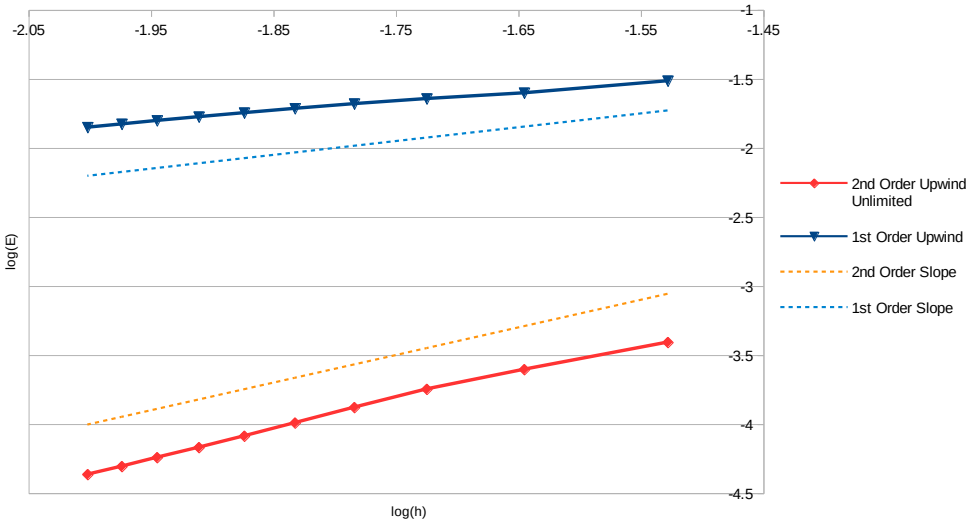


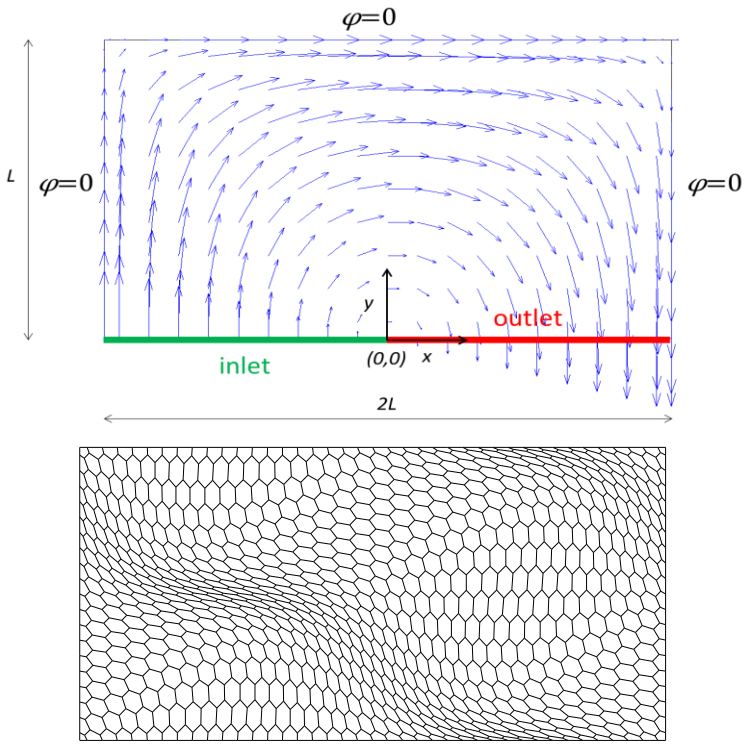
Figure 3: Convection-diffusion: h-convergence for 1st and 2nd-order upwind MHFV schemes.

Interestingly enough, the 2nd-order scheme converges on all grids even though we did not apply any form of flux limiting or stabilization; we

attribute this fortunate result to the relative smoothness of the solution to this test case.

9. Comparison of stabilization strategies

We now want to analyse how the various stabilization techniques described in Section 4 compare against each other. To this purpose we selected a specifically designed convection-diffusion test case: the Smith-Hutton problem [29].



(a)

(b)

Figure 4: Smith-Hutton problem: BCs setup and convective flow field (a) and mesh type (b).

The problem is solved over a 2D rectangular domain of height $L = 1$ and length $2L$, with BCs and convective flow $\boldsymbol{U}$ set as shown in Figure 4(a) (see [29] for the analytical expression of $\boldsymbol{U}$). At the inlet, the distribution of the unknown φ is:

$$\varphi_{in}(x) = 1 + \tanh[\alpha(1 + 2x)]$$

which, for high values of α (here we take $\alpha = 100$), defines a profile containing a rather steep jump from 0 to 2, as shown in Figure 5:

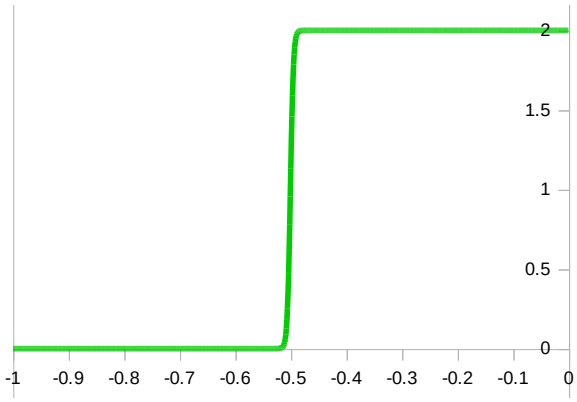


Figure 5: Distribution of φ_{in} imposed at the inlet ($\alpha = 100$).

For a purely convective problem an exact mirror image of such profile would be produced at the outlet; we can expect a similar result by setting low diffusivity, $\nu = 10^{-6}$, meaning we have a high Peclet number: $Pe \approx 10^{6}$. The problem is specifically designed to cause severe instabilities with high-order schemes via the presence of a near-discontinuity in the solution field.

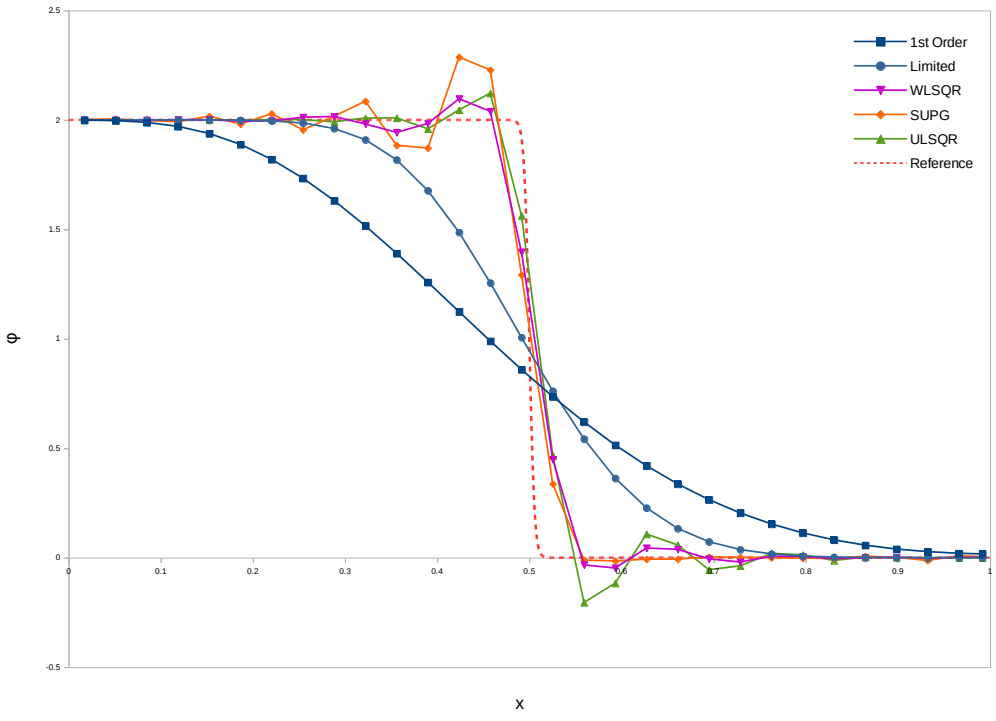


Figure 6: Distribution of φ at the outlet, comparison among stabilization schemes; $Pe \approx 10^{6}$, $h \approx 3.7 \times 10^{-2}$.

We solve over a distorted polygonal mesh (rather coarse, $h \approx 3.7 \times 10^{-2}$), similar to the one in Figure 4(b). After verifying that neither a

centered scheme nor an unlimited 2nd-order upwind scheme can deal with the problem (either the linear solver fails or it produces meaningless results), we test a selection of the previously mentioned stabilization strategies, including our own ULSQR. We plot in Figure 6 the distribution of φ_F at the outlet as computed by each scheme, which helps us perform a qualitative analysis of results; we use as a reference solution a mirror image of the inlet distribution.

As expected, 1st-order upwinding is non-oscillatory but definitely too diffusive (notice how heavily it smoothens the near-discontinuity). Compared to that, the traditional 2nd-order scheme with (face-bounded) Venkatakrishnan flux limiter is closer to the reference solution whilst still being monotone, although a considerable amount of numerical diffusion is still visible.

Both the WLSQR and ULSQR schemes perform comparably: overshoots are present on either side of the discontinuity, but we can expect them to be bounded regardless of Pe or mesh parameters (for WLSQR, stability is theoretically proven in [16] for special discontinuous data; we have not yet conducted a rigorous mathematical analysis of ULSQR, which remains for now empirical in nature). Judging by the amplitude of oscillations, in particular on the lower side, WLSQR would appear to be slightly more favourable; such result however may be case-dependent, especially considering that we tuned the exponents in weight formulation (35) based on the present results. On the other hand, the reader is reminded that ULSQR possesses the attractive feature of not being data-dependent, meaning that it only requires one linear solve and no iterative processes.

As for SUPG we notice that oscillations are completely dumped on the lower side of the step, but there is a significant overshoot at the top. Further test results (not published here) reveal that, on more regular meshes, such issue is not as pronounced; this might indicate that our choice of SUPG stabilization parameter (38), whilst guaranteeing convergence, does not adapt well enough to generic meshes, thus exhibiting excessive nonphysical oscillations. Further investigations on the matter are left to our future work.

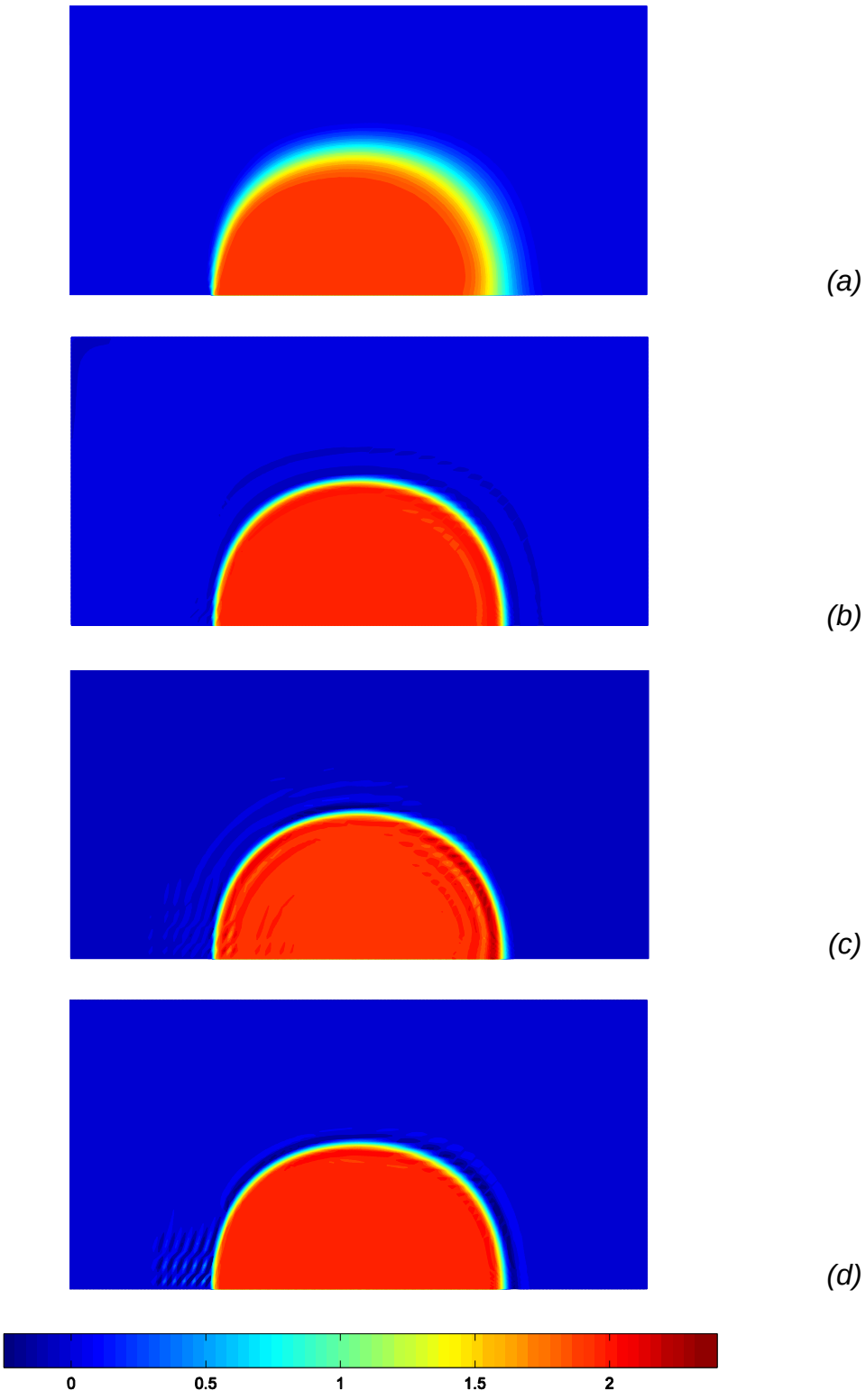


Figure 7: Solution field φ computed via schemes: flux limiting (a), WLSQR (b), SUPG (c), ULSQR (d); $Pe \approx 10^6$.

For further analysis, we also plot in Figure 7 the entire solution field for each strategy. Besides confirming the conclusions we drew from outlet profile analysis (most notably the excessive diffusiveness caused by flux limiting) we also observe that, at least for this specific case, WLSQR seems indeed to be slightly superior: the solution appears to be more uniform and well smoothed in areas away from the overshoot and undershoot near the step, while SUPG and ULSQR both present oscillations propagating from the discontinuity to a considerable extent.

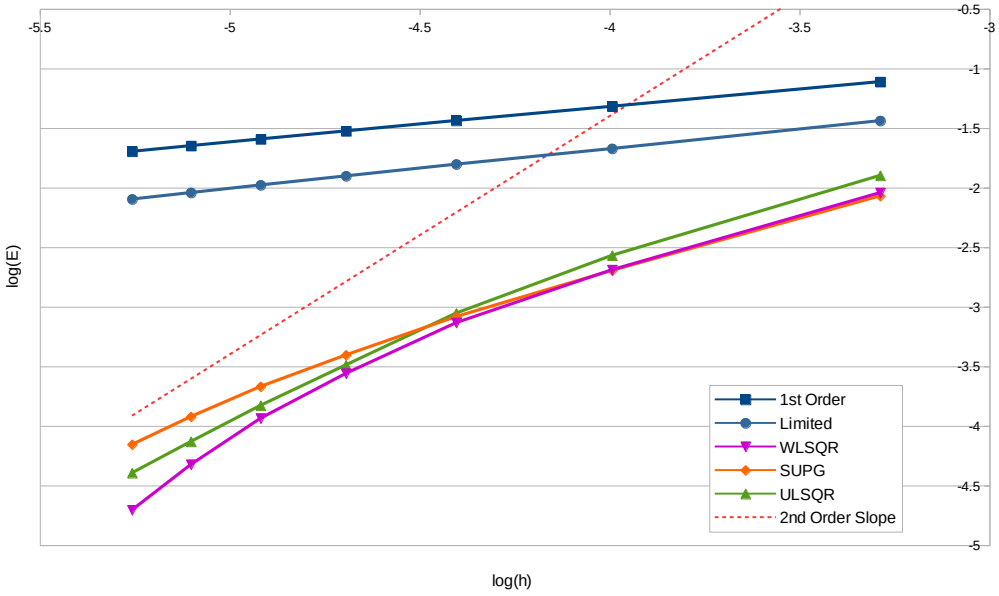


Figure 8: Smith-Hutton: h-convergence for different convection schemes ($Pe \approx 10^6$).

We conclude this section with a h-convergence analysis, carried out via the same methodology as described in Section 8. We use the solution to the purely convective problem as reference; results are reported in Figure 8. It is noticeable how the flux limiting procedure severely degrades the order of h-convergence of the method: for this test case, flux limiters actually push it back to 1st-order, although the error is consistently smaller than with 1st-order upwind.

On the other hand, WLSQR, SUPG and ULSQR all perform comparably: one can see a pre-asymptotic behaviour in the sequence, which steepens as the mesh is refined until reaching an exact (or close to) 2nd-order slope in the last few entries. Absolute values of error norms do not display any significant differences amongst them either apart from, again, a slight superiority displayed by WLSQR.

10. Validation of the Navier-Stokes Scheme: Poiseuille Flow

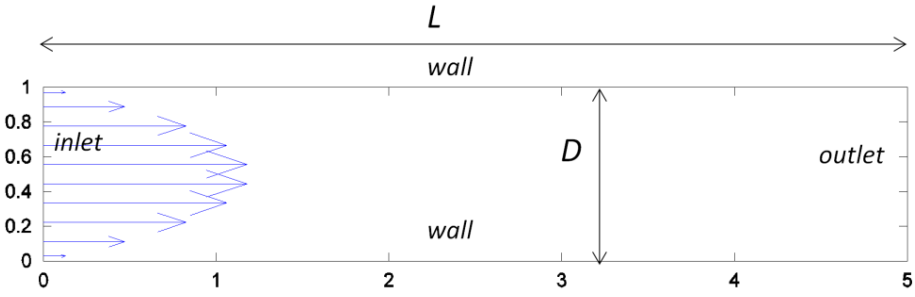
In order to perform a first h-convergence analysis of our full Navier-Stokes scheme we choose a basic 2D test case: the laminar plane Poiseuille flow. We solve over a channel of width $D = 1$ and length $L = 5$ with BCs set as in Figure 9(a). Setting $\nu = 10^{-2}$, imposing a total pressure drop $\Delta p = (p_{in} - p_{out}) = 2$ and assuming laminar flow, we have the following analytical solution for the x-component of $\boldsymbol{U}$:

$$u(y) = \frac{\Delta p}{2\nu}(y - y^2)$$

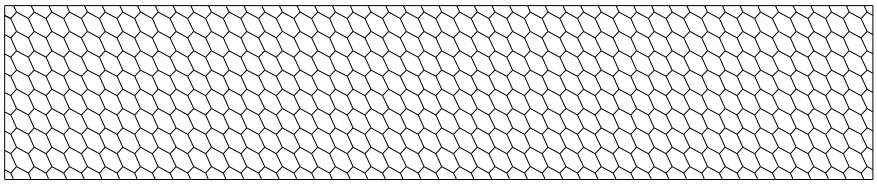
which we impose at the inlet, whereas $v = 0$ everywhere. Pressure is linearly distributed:

$$p(x) = \Delta p(L - x)$$

and defined up to a constant (here we chose to define it such that $p(L) = 0$, i.e. zero pressure at the outlet). The Reynolds number, based on the channel's half-width and maximum velocity, is $Re = \frac{D\, u_{max}}{2\nu} = 1250$, which is high enough but still well below the critical value $Re_{crt} = 5314$ above which, according to literature [21], the laminar assumption no longer holds.



(a)



(b)

Figure 9: Poiseuille flow: BCs setup (a) and mesh type (b).

We solve over fairly regular, although non-orthogonal, polygonal meshes – a sort of skewed honeycomb structure, Figure 9(b). The

scheme is set to 2nd-order accuracy for both $\boldsymbol{U}$ and p; we use our ULSQR strategy to stabilise convective terms.

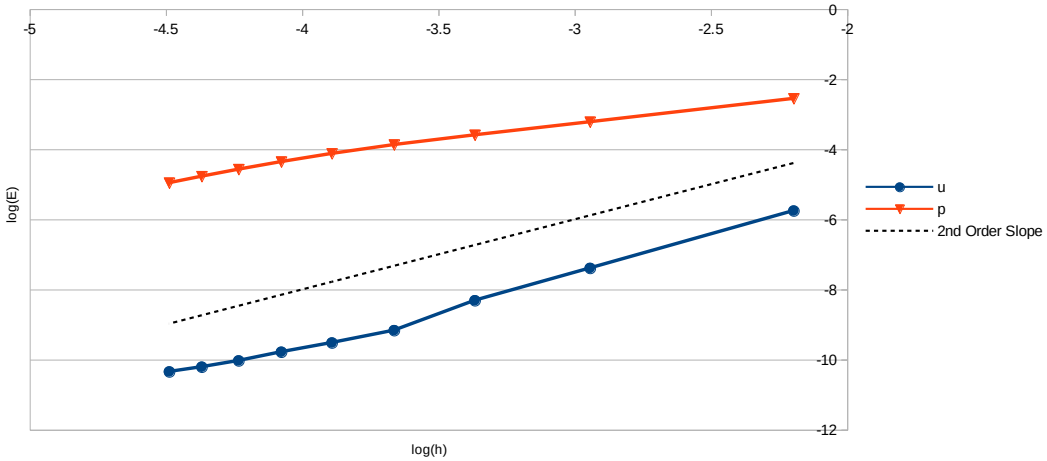


Figure 10: Poiseuille flow, h-convergence for variables u_C and p_C (2nd-order scheme).

Figure 10 reports h-convergence results for variables u_C and p_C (v_C would be somewhat more difficult to analyse since the exact solution is 0 everywhere in the channel, making it problematic to compute the error in $\boldsymbol{L}^2$-norm). Results for u are clearly very positive, confirming a slope fairly close to the theoretical 2nd-order trend (despite the usage of decentered gradients in the ULSQR scheme) and only minor oscillations near the inlet and outlet for coarser meshes – small enough to not be visible in the solution field, see Figure 11(b). Comparison with a 1st-order accurate solution field (Figure 11(a)) also shows the definite superiority of the higher order scheme due to the significant numerical diffusion displayed by the former.

Unfortunately, results for p are in this case less satisfactory: despite the formally 2nd-order scheme, we still observe accuracy closer to 1st-order, although the slope in Figure 10 appears to steepen as we refine the mesh. One could explain the phenomenon by assuming a pre-asymptotic behaviour, but this would seem excessively stretched since the finest meshes in the sequence are rather well refined ($h \approx 5.2 \times 10^{-3}$).

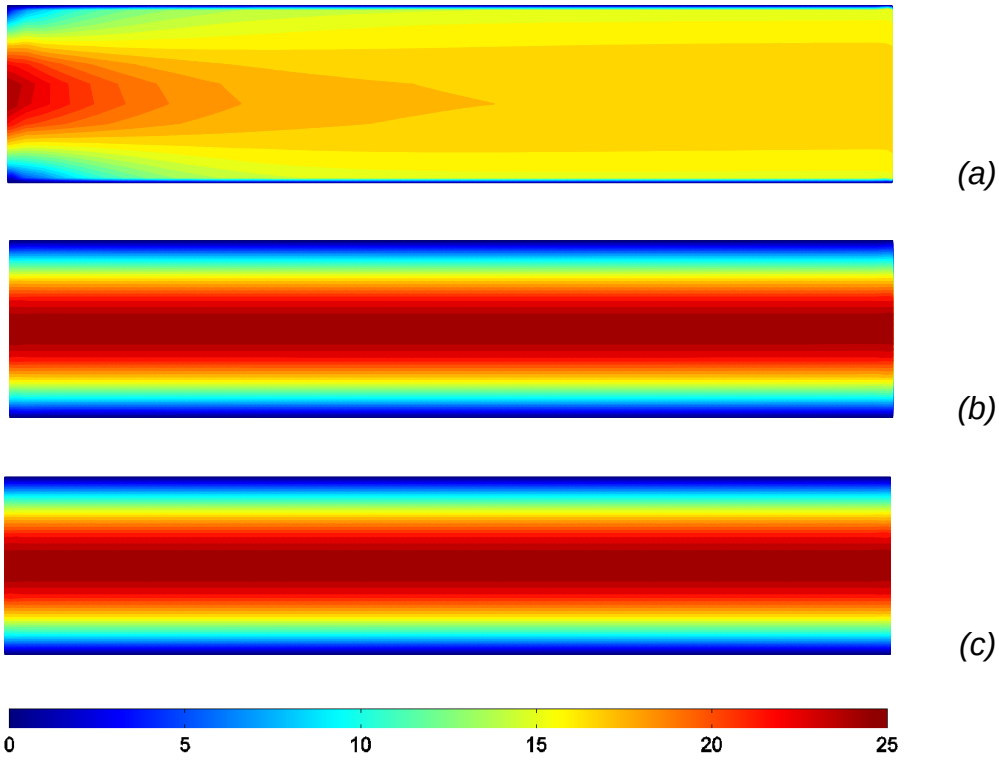


Figure 11: Velocity magnitude in 1st-order (a) and 2nd-order (b) accuracy, compared to the analytical solution (c); $h \approx 0.11$.

An analysis of the solution field p_C (Figure 12) suggests another possible explanation. Clearly, much of the error is concentrated in a small area at the inlet, where some local peaks are visible, while in the rest of the channel we observe a trend fairly close to the expected linear distribution; this might be due to the fact that the analytical distribution of u imposed at the inlet is incompatible with the discrete solution u_F satisfying the discrete Navier-Stokes system, and the solver responds to that by creating a small transitional trait with altered pressure values resolving the inconsistency, which would in turns degrade accuracy on p.

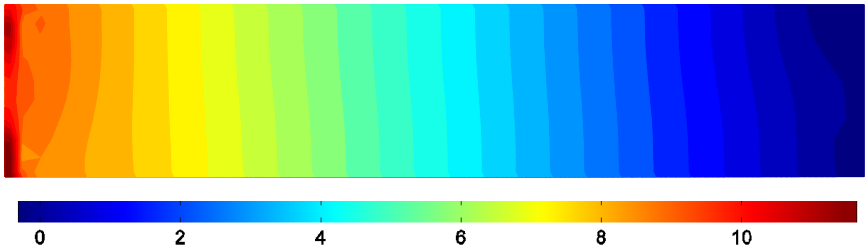


Figure 12: Pressure field in 2nd-order accuracy; $h \approx 0.11$.

In that case the issue would be problem-specific and not related to the scheme itself; thus, a change in BCs definition (e.g. periodic instead of Dirichlet for $\boldsymbol{U}$) would solve the problem. If that were to fail, then it would be necessary to investigate the discrete pressure scheme in more detail and look for possible sources of instability. We shall leave such considerations to our future work, deeming the present results acceptable for the purpose of this paper.

11. Lid-Driven Cavity Test Case

We further validate our MHFV Navier-Stokes solver via the well-known 2D lid-driven cavity test case, which is set up as in Figure 13(a) over a square domain of side $L = 1$. We set $u_{lid} = 1$ and we let viscosity ν vary in order to match $Re = 10^2$, 10^3 and 10^4 which will allow us to compare with previous literature.

More specifically, we compare against benchmark results reported by Ghia *et al.* [17], which are computed over a uniform Cartesian grid of size 129×129. Our simulations are run on a slightly coarser quadrilateral mesh (120×120) which we also distort to introduce strongly non-orthogonal elements, as in Figure 13(b); notice that our distortion pattern also causes further mesh coarsening in certain (arbitrarily located) areas w.r.t. the original uniform grid.

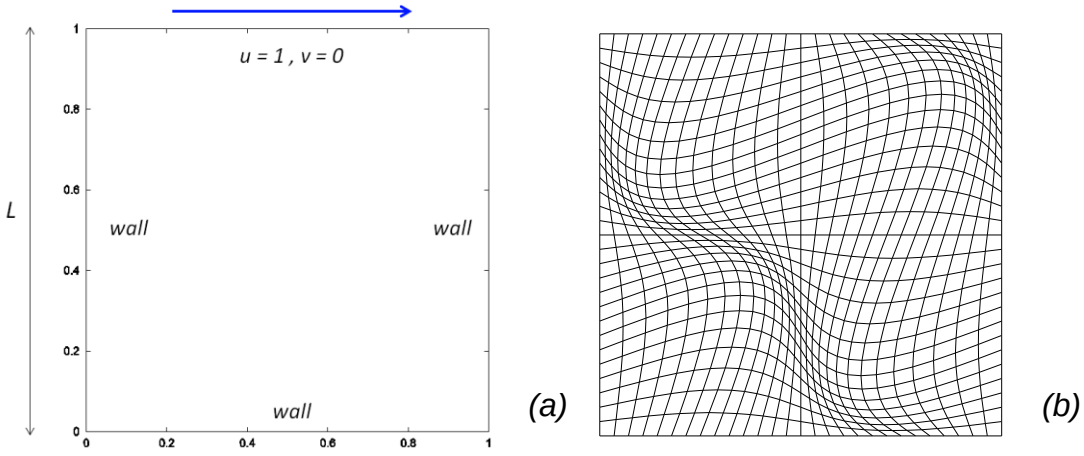


Figure 13: Lid-driven cavity: BCs setup (a) and mesh type (b).

We extract from the MHFV solution field two sets of values: u along a vertical line and v along a horizontal line passing through the geometric center of the cavity, and we plot them together with those found in literature (Figure 14). We observe excellent agreement for all three values of Re, despite the coarser and distorted mesh we used.

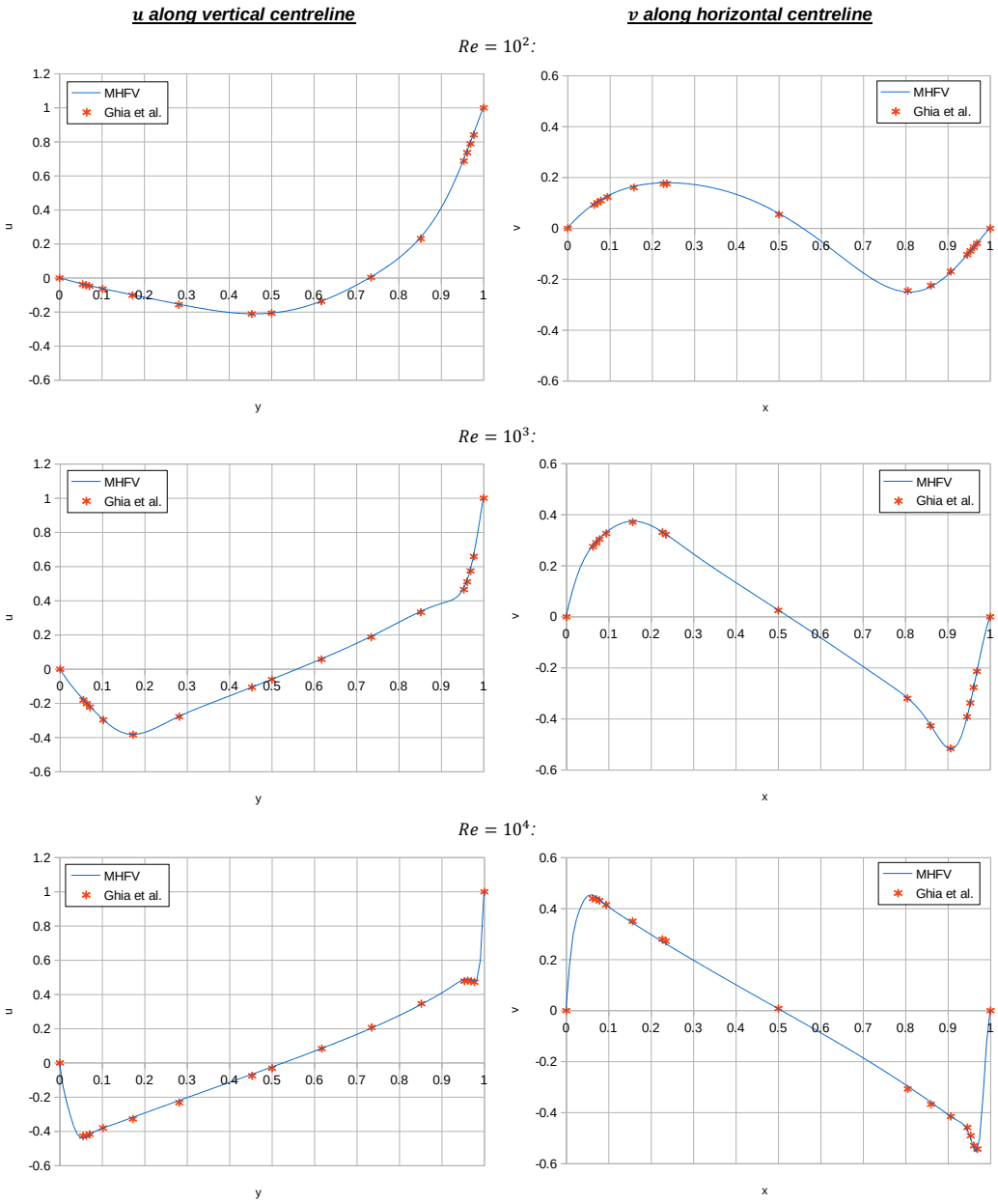


Figure 14: Profiles of u along vertical centerline and v along vertical centerline; comparison with benchmark results.

We also report in Figure 15 the velocity magnitude field produced by our MHFV solver compared to the product of a commercial CFD FV solver, the first computed on a 120×120 grid distorted as in Figure 13(b), the second on a similarly-sized Cartesian grid; both solvers are set to 2nd-order accuracy, with ULSQR stabilization for the MHFV scheme and classical flux limiting for the FV one.

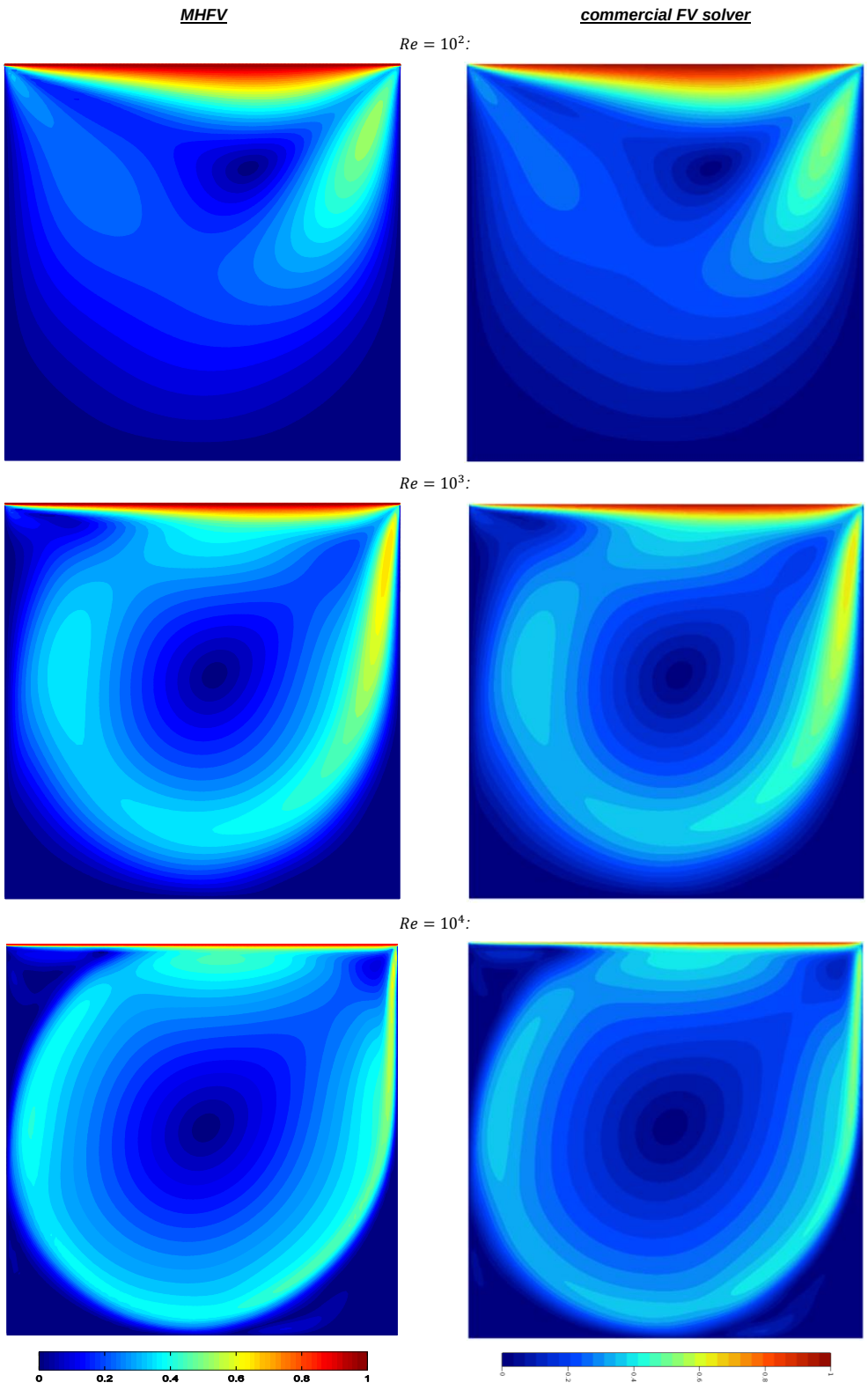


Figure 15: Velocity magnitude computed via MHFV on a distorted mesh (left) and a commercial FV solver on a Cartesian mesh (right) at different Re.

Again, our results are very satisfactory for all Re values – except for perhaps some slight overestimations in certain areas, but then again we have no reason to assume that the reference FV results are more reliable.We want to emphasize here how our results appear to be completely unaffected by the underlying distorted grid pattern shown in Figure 13(b), and stay true to the problem's physics. It is also worth mentioning that, when we attempted to solve over the same distorted grid with the commercial software, it failed to converge (most likely due to excessive non-orthogonality and/or skewness).

We conclude this section with a preliminary analysis of our SIMPLEC strategy, outlined in Section 6. We run the lid-driven cavity test case on two mesh types: Type A - the same distorted quadrilateral grid, pictured in Figure 13(b), and Type B - a skewed honeycomb grid like the one we used in Section 10, Figure 9(b). We test for two Re values: 10^2 and 10^3 for different degrees of mesh refinement (we avoid higher Re values due to steady-state convergence issues on coarse grids); we take the number of faces in each mesh as a measure of the problem size.

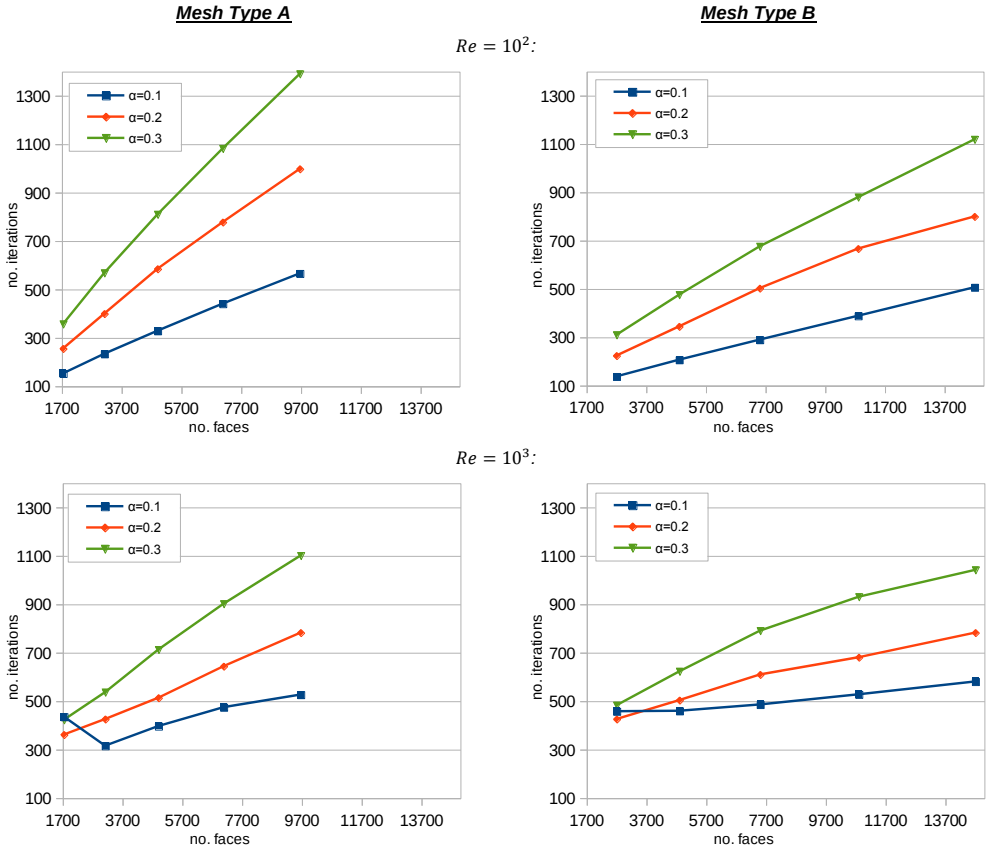


Figure 16: MHFV SIMPLEC: iteration count to steady-state for different mesh types/sizes, Reynolds numbers and relaxation factors α.

The goal is to investigate how the choice of inertial relaxation factor α in (73) affects the performance of SIMPLEC, and whether or not the optimal choice depends on physical and/or numerical parameters. Hence we run each case with values $\alpha = 0.1, 0.2$ and 0.3.

The graphs in Figure 16 show the number of iterations required to converge to steady-state, given a tolerance of 10^{-4}, in function of all of the above mentioned parameters. It appears that: for both mesh types, both Re and all levels of grid size, setting $\alpha = 0.1$ gives the best performance in terms of both iteration count and algorithm scaling vs. problem size. The only exception is the coarsest mesh at $Re = 10^3$ for both mesh types, for which the optimal α is likely to be in between 0.1 and 0.2 (we did verify that decreasing α below 0.1 increased the iteration count).

Results are encouraging, as they suggest that the optimal relaxation factor does not depend on the mesh type, and it is only slightly influenced by grid coarseness and problem physics. However, further testing involving other mesh types, a wider range of Re and other problem definitions is in order before drawing any definite conclusions.

On the other hand, these results also highlight a severe limitation of SIMPLEC itself: it heavily depends on numerical parameters, and in particular on mesh type. Notice how the iteration count is in general much higher, and grows more rapidly with the problem size, for mesh Type A (left column in Figure 16). This can be attributed to the irregularity of Type A, which features several strongly distorted cells as well as a wide range of cell volumes/face areas, whereas Type B is only slightly non-orthogonal, and very regular in terms of element size; this might be causing SIMPLEC to underperform on Type A compared to Type B, despite the considerably smaller problem size. We take such preliminary analysis as a motivation to look into more efficient ways of solving the discrete Navier-Stokes in the future.

12. Conclusions

We believe that our results, although based on relatively simple test cases, are a clear sign of the potential of MVE methods in the CFD industry. Particularly relevant is the fact that our Navier-Stokes scheme proved itself to be in some way superior to a commercial FV solver, the latter having failed to solve the lid-driven cavity problem over the same distorted mesh. In general we can argue that MHFV, once finalized, will be able to tackle industrial-sized problems with much more ease compared to the currently existing methods: not just because of the

extra freedom on grid geometry, but also, and more importantly, thanks to its intrinsic stability and robustness. The only drawback we can think of at this stage is the increased size of linear systems to be solved – our hybrid operators scale with the number of mesh faces, rather than cells. Efforts should be made in this sense towards improving CPU efficiency of linear solvers, a matter we have not dealt with in this paper.

In the immediate future we plan to tackle some open questions raised by the present work: further research into stabilization techniques for 2^{nd}-order convection schemes (SUPG in particular, being very heuristic in nature, can be expressed via several different formulations besides the one we derived in Section 4); thorough analysis of stability properties of our 2^{nd}-order pressure discretization, see Section 7; investigation of alternative, more efficient preconditioners to the Navier-Stokes system, other than SIMPLE-like strategies – a promising idea may be the Pressure Convection-Diffusion (PCD) preconditioner, see e.g. [14].

Efforts will also be made towards the implementation of features that will allow us to test on actual industrial cases, namely: a turbulence modeling module – not too challenging, since many existing models simply require solving additional transport equations, which we can discretize via the scheme in Section 3; extension to unsteady flow capabilities – either via well-established methods or by investigating new possibilities; implementation of a MHFV adjoint flow solver, in order to assess performance in the context of design optimization – which, as mentioned in Section 1, is what ultimately drives our work.

Acknowledgements: This work was made possible by the technical support of ESI Group France and involvement of its scientific research team, and funded by the EU through the FP7-PEOPLE-2012-ITN “AboutFlow” grant agreement number 317006.

13. References

1 Antonietti P., Da Veiga L., Bigoni N., Verani M. (2014). *Mimetic Finite Differences for non-linear and control problems*. Mathematical Models and Methods in Applied Sciences, 24 (8). 1457-1493.

2 Antonietti P., Bigoni N., Verani M. (2013). *Mimetic Finite Difference method for shape optimization problems*.

Proceedings of the 10th European Conference on Numerical Mathematics and Advanced Applications; Springer Verlag Italia.

3 Barth T., Jespersen D. (1989). *The design and application of upwind schemes on unstructured meshes*. AIAA paper 89-0366.

4 Brezzi F., Lipnikov K., Shashkov M. (2005). *Convergence of mimetic finite difference method for diffusion problems on polyhedral meshes*. SIAM Journal of Numerical Analysis, 43 (5). 1872-1896.

5 Brezzi F., Lipnikov K., Simoncini V. (2005). *A family of mimetic finite difference methods on polygonal and polyhedral meshes*. Mathematical Models and Methods in Applied Sciences, 15 (10). 1533-1551.

6 Brezzi F. (2008). *Cochain approximations of differential forms*. Foundations of Computational Mechanics, Hong Kong, June 16-26.

7 Brezzi F., Falk R.S., Marini L.D. (2014). *Basic principles of mixed virtual element method*. Mathematical Modeling and Numerical Analysis, 48 (4). 1227-1240.

8 Brooks A., Hughes T. (1982). *Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations*. Computational Methods in Applied Mechanics and Engineering, 32. 199–259.

9 Cangiani A., Manzini G. (2008). *Flux reconstruction and solution post-processing in mimetic finite difference methods*. Computational Methods in Applied Mechanics and Engineering, 197. 933-945.

10 Da Veiga L.B., Droniou J., Manzini G. (2011). *A unified approach to handle convection terms in Finite Volumes and Mimetic Discretization Methods for elliptic problems*. IMA Journal of Numerical Analysis, 31 (4). 1357-1401.

11 Droniou J., Eymard R. (2006). *A mixed finite volume scheme for anisotropic diffusion problems on any grid*. Numerische Mathematik, 105 (1). 35-71.

12 Droniou J., Eymard R. (2008). *Study of the mixed finite volume method for Stokes and Navier-Stokes equations*. Numerical Methods for Partial Differential Equations, 25 (1). 137-171.

13 Droniou J. (2010). *Remarks on discretizations of convection terms in hybrid mimetic mixed methods*. Networks and Heterogeneous Media, 5 (3). 545-563.

14 Elman H., Howle V., Shadid J., Shuttleworth R., Tuminaro R. (2006). *Block preconditioners based on approximate commutators*. SIAM Journal on Scientific Computing, 27 (5). 1651-1668.

15 Eymard R., Gallouet T., Herbin R. (2010). *Discretization of heterogeneous and anisotropic diffusion problems on general nonconforming meshes SUSHI: a scheme using stabilization and hybrid interfaces*. IMA Journal of Numerical Analysis, 30 (4). 1009-1043.

16 Furst J. (2005). *A Finite Volume scheme with Weighted Least Square reconstruction*. Finite Volumes for Complex Applications IV, Hermes Science. 345-354.

17 Ghia U., Ghia N., Shin T. (1982). *High-Re solutions for incompressible flow using the Navier-Stokes equations and a Multigrid Method*. Journal of Computational Physics, 48. 387-411.

18 Harten A., Engquist B., Osher S., Chakravarthy S. (1987). *Uniformly high order accurate essentially non-oscillatory schemes III*. Journal of Computational Physics, 71. 35-61.

19 Jasak H. (1996). *Error analysis and estimation for the Finite Volume Method with applications to fluid flows*. Ph.D. Thesis, Imperial College, London (UK).

20 Kuznetsov Y., Lipnikov K., Shshkov M. (2004). *The mimetic finite difference method on polygonal meshes for diffusion-type problems*. Computational Geosciences, 8 (4). 301-324.

21 Lin C. (1955). *The theory of hydrodynamic stability*. Cambridge University Press (UK).

22 Lipnikov K., Shashkov M., Svyatskiy D. (2005). *The mimetic finite difference discretization of diffusion problem on unstructured polyhedral meshes*. Journal of Computational Physics, 211. 473-491.

23 Lipnikov K. (2010). *Mimetic finite difference method for solving PDEs on polygonal and polyhedral meshes*. International Workshop Non-Standard Numerical Methods for PDEs, University of Pavia (Italy). June 29-July 2.

24 Lipnikov K., Manzini G., Brezzi F., Buffa A. (2011). *The mimetic finite difference method for the 2D magnetostatic field problems on polyhedral meshes*. Journal of Computational Physics, 230 (2). 305-328.

25 Liu X., Osher S., Chan T. (1994). *Weighted essentially non-oscillatory schemes*. Journal of Computational Physics, 115. 200-212.

26 Michalak K., Ollivier-Gooch C. (2008). *Limiters for unstructured higher-order accurate solutions of the Euler equations*: AIAA paper 08-0776.

27 Oriani M., Pierrot G. (2014). *Alleviating adjoint solver robustness issues via mimetic CFD discretization schemes*. Opt-I, International Conference on Engineering and Applied Sciences Optimization, Kos (Greece). June 4-6.

28 Piar L., Babik F., Herbin R., Latch J.C. (2013). *A formally second order cell centered scheme for convection-diffusion equations on unstructured non-conforming grids*. International Journal for Numerical Methods in Fluids, 71. 873-890.

29 Smith R., Hutton A. (1982). *The numerical treatment of advection: a performance comparison of current methods*. Numerical Heat Transfer, 5. 439-461.

30 Veersteg H., Malalasekera W. (2007). *An introduction to Computational Fluid Dynamics: the Finite Volume method*; Pearson Education, Harlow.

31 Venkatakrishnan V. (1993). *On the accuracy of limiters and convergence to steady-state solution*. AIAA paper 93-0880.