J. Martínez, F. Piscaglia, A. Montorfano, A. Onorati, S.M. Aithal
Influence of Momentum Interpolation Methods on the Accuracy and Convergence of Pressure–velocity Coupling Algorithms in OpenFOAM
Journal of Computational and Applied Mathematics, Vol. 309, 2017, p. 654-673
doi:10.1016/j.cam.2016.03.037

Influence of momentum interpolation methods on the accuracy and convergence of pressure-velocity coupling algorithms in OpenFOAM®

J. Martínez*,[a], F. Piscaglia[a], A. Montorfano[a], A. Onorati[a], S. M. Aithal[b]

[a] *Dip. di Energia, Politecnico di Milano, Via Lambruschini 4, 20156, Milano-Italy*
[b] *Argonne National Laboratory, Lemont, IL 60439, United States*

Abstract

Checkerboard pressure is one of the most significant problems arising from the use of collocated grids for fluid dynamic simulations using the finite-volume method. The original Rhie-Chow momentum interpolation technique, termed Original Momentum Interpolation Method (OMIM) was proposed to eliminate the non-physical saw-tooth pressure oscillations. However, it was soon proved that the steady-state solutions obtained with this technique were under-relaxation factor dependent. Nevertheless, standard OMIM is still commonly used in several CFD codes like OpenFOAM®, the software used for this work. In this paper the OMIM and a possible correction for under-relaxation dependency, which has been implemented in OpenFOAM®, are discussed in detail. The proposed methodology is compared and contrasted with OMIM in terms of accuracy of the solution and speed of convergence for several classical pressure-velocity segregated algorithms for steady state solvers; namely SIMPLE, SIMPLE-C, SIMPLE-R

*Corresponding author
Email address: jmartrubio@gmail.com (J. Martínez)

and PISO. A classical laminar 2D cavity is used as the base test-case. The study is then extended to a more complex 2D airfoil profile (NACA0012). In the cases considered mesh uniformity and orthogonality are progressively reduced and turbulence starts playing an important role, limiting therefore the convergence of the cases and the performance of the correction.
Key words: Rhie-Chow, Momentum interpolation, SIMPLE algorithm, pressure-velocity coupling, OpenFOAM

---

1. Introduction

The use of co-located grid arrangement for finite-volume simulations of incompressible flows has gained high popularity in the recent years in both general-purpose and commercial flow solvers. While an improved stability and robustness can be obtained with a staggered grid, the co-located grid arrangement can significantly reduce memory storage and calculation time, and simultaneously simplify the implementation of solvers; these are in fact the main reasons explaining its extended use.

One of the main problems arising from the use of co-located grid arrangement is the checkerboard pressure field. Due to the nature of the Navier Stokes equations, pressure appears in the momentum equations inside a gradient term. Application of central-difference spatial discretization to this term in a co-located grid produces a decoupling of pressure and velocity cell values, leading to saw-tooth pressure oscillations. In 1983, Rhie and Chow [1] proposed a technique for momentum-based interpolation of mass fluxes on cell faces, imitating the staggered-grid discretization. This interpolation is based on formulating a discretized momentum equation for the face, so that

the computation of the driving pressure force involves the pressure value at the nodes adjacent to the face in question and, therefore at the node itself. This technique removes the checkerboarding problem for the most part, which is the reason of its wide acceptance and intensive use in unstructured grid solvers. This Momentum Interpolation Method underwent extensive development for complex geometries [2], unsteady flows [3, 4, 5, 6] or flows with large body forces [7].

Velocity under-relaxation is usually required to achieve convergence. When this is the case, the Original Momentum Interpolation by Rhie-Chow presents some additional problems. Majumdar [8] and Miller et al. [9], independently reported that solutions obtained with the original Rhie-Chow interpolation method are dependent on the velocity under-relaxation factor. Furthermore, the use of very small under-relaxation factors could make the checkerboard pressure reappear. The corrected version of momentum interpolation proposed in [8] termed as the Majumdar Momentum Interpolation Method (MMIM) completely eliminates this dependency. The main objective of this paper is to conduct a systematic study of the MMIM on the solution accuracy and convergence speed applied to four well-known segregated pressure-velocity coupling algorithms, namely, SIMPLE (Semi-Implicit Method for Pressure Linked Equations) [10], SIMPLER (SIMPLE-Revised) [11], SIMPLEC (SIMPLE-Consistent) [12] and PISO (Pressure Implicit with Splitting of Operators) [13].

The open-source finite-volume CFD code OpenFOAM® [14] is used in the present study. The authors have conducted extensive development of this computational tool for LES simulations [15, 16, 17, 18, 19, 20, 21, 22].

This development is extended to include the implementation of MMIM and alternative pressure-velocity coupling algorithms not present in the standard version at time the work has been performed; namely SIMPLE-C and SIMPLE-R.

In order to test the influence that MMIM has on the different algorithms commented before, two test-cases are selected, namely a simple 2D laminar lid-driven cavity and a NACA0012 airfoil. In the first case, the theory behind MMIM is analyzed. Several mesh types (orthogonal uniform, orthogonal non-uniform, triangle, unstructured hex-dominant) and refinement degrees are evaluated. For the second case a more complex airfoil profile (NACA 0012) is tested. A 2D structured mesh is used to compute pressure, drag and lift coefficients for different angles of attack, and compare the results obtained with different velocity under-relaxation factors when OMIM or MMIM are used. This case is convenient since turbulence might limit the maximum convergence achieved by the solver and therefore the correction of momentum interpolation, which ensures under-relaxation factor independency when fully convergence of the solver is achieved, might not completely cancel the under-relaxation factor dependency.

2. Rhie-Chow Momentum Interpolation Method

In the following OMIM is described in detail. Dependency of the steady-state OMIM solutions on the under-relaxation factor is discussed.

The governing equations for an steady laminar incompressible flow, in the absence of other body or external forces are:

$$\frac{\partial}{\partial x_i}(\rho u_i) = 0 \tag{1}$$

$$\frac{\partial}{\partial x_j}(\rho u_i u_j) = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} \tag{2}$$

where x_i is te cartesian framework, u_i is the velocity component in the x_i coordinate, p is the pressure and ρ denotes the density. Assuming a Newtonian fluid, shear stress τ_{ij} can be determined by:

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3}\mu \frac{\partial u_k}{\partial x_k}\delta_{ij} \tag{3}$$

where μ is the dynamic dynamic viscosity and δ_{ij} is the Kronecker delta. For the sake of simplicity, the OMIM will be described in a two-dimensional orthogonal grid as the one shown in Fig. 1.

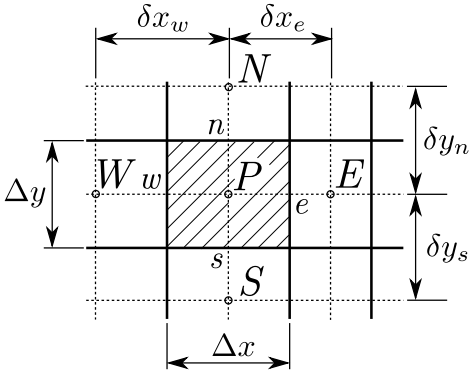


Fig. 1: Two-dimensional orthogonal grid. N, S, E and W correspond to neighbour cells of cell P; n, s, e and w denote cell P faces; Δx and Δy are cell P dimensions in the x and y spatial coordinates; δy_n, δy_s, δx_e and δx_w correspond to cell-center to cell-center distances from cell P to neighbour cells.

Under this considerations, the conservation equation for a general flow variable ϕ in the steady laminar case described above, in the absence of any source term, can be written as:

$$\frac{\partial}{\partial x}(\rho u \phi) + \frac{\partial}{\partial y}(\rho v \phi) = \frac{\partial}{\partial x}\left(\Gamma_\phi \frac{\partial \phi}{\partial x} \right) + \frac{\partial}{\partial y}\left(\Gamma_\phi \frac{\partial \phi}{\partial y} \right) \tag{4}$$

where u and v are the x and y components of the velocity field and Γ_ϕ is the diffusion coefficient. Integrating Eq. (4) in the computational cell, and applying Green-Gauss theorem the semi-discretized form of the equation can be written as follows:

$$\Delta y\left[(\rho u\phi)_e-(\rho u\phi)_w\right]+\Delta x\left[(\rho v\phi)_n-(\rho v\phi)_s\right]= \\ \Delta x\left[\frac{\Gamma_n}{\delta y_n}(\phi_N-\phi_P)-\frac{\Gamma_s}{\delta y_s}(\phi_P-\phi_S)\right]+\Delta y\left[\frac{\Gamma_e}{\delta x_e}(\phi_E-\phi_P)-\frac{\Gamma_w}{\delta x_w}(\phi_P-\phi_W)\right] \tag{5}$$

where central differencing scheme has been used for diffusive terms. In the discretization of divergence terms, many schemes are available in literature (first-order upwind, second-order upwind, central differencing scheme, QUICK [23]). Second and higher-order schemes have been widely used in applications involving orthogonal and uniform meshes. However, the stability of higher-order schemes for applications involving turbulent flows in complex geometries is not guaranteed and convergence may be difficult to achieve. First or second-order methods are best suited for such applications.

Application of first-order upwind discretization scheme to convective terms of Eq. (5) yields the following final discretized form of Eq. (4).

$$A_P\phi_P=A_E\phi_E+A_W\phi_W+A_N\phi_N+A_S\phi_S+b_P \tag{6}$$

where

$$
\begin{aligned}
A_E =& \frac{\Gamma_e \Delta y}{\delta x_e} + \max(-\rho u_e \Delta y, 0) \\
A_W =& \frac{\Gamma_w \Delta y}{\delta x_w} + \max(\rho u_w \Delta y, 0) \\
A_N =& \frac{\Gamma_n \Delta x}{\delta y_n} + \max(-\rho v_n \Delta x, 0) \\
A_S =& \frac{\Gamma_s \Delta x}{\delta y_s} + \max(\rho v_s \Delta x, 0) \\
b_P =& - \max(\rho u_e \Delta y, 0)(\phi_e - \phi_P) + \max(-\rho u_e \Delta y, 0)(\phi_e - \phi_E) \\
& - \max(-\rho u_w \Delta y, 0)(\phi_w - \phi_P) + \max(\rho u_w \Delta y, 0)(\phi_w - \phi_W) \\
& - \max(\rho v_n \Delta x, 0)(\phi_n - \phi_P) + \max(-\rho u_n \Delta x, 0)(\phi_n - \phi_N) \\
& - \max(-\rho v_s \Delta x, 0)(\phi_s - \phi_P) + \max(\rho u_s \Delta x, 0)(\phi_s - \phi_S)
\end{aligned}
\tag{7}
$$

where deferred-correction procedure [24] is used for the term b_P.

A_P coefficients can be determined by

$$A_P = A_E + A_W + A_N + A_S + A_b \tag{8}$$

where A_b is the mass residual. The A_b term is usually dropped since divergence-free conditions are required for the velocity field:

$$A_b = \rho u_e \Delta y - \rho u_w \Delta y + \rho v_n \Delta x - \rho v_s \Delta x \tag{9}$$

Eq. (8) is true as long as conservative discretization schemes are used [24]. It should also be noticed that a diagonally dominant matrix ($A_P \geq \sum_{NB} |A_{NB}|$ where NB refers to the neighbour cells of cell P) is a sufficient condition for convergence of iterative methods and the inequality must be satisfied at least at one node [24].

In order to reach convergence under-relaxation factors are typically applied to dependent variables to limit the change in consecutive iterations. From Eq. (6) we can write the final value of ϕ_P as:

$$\phi_P = \frac{\alpha_\phi}{A_P}(A_E\phi_E + A_W\phi_W + A_N\phi_N + A_S\phi_S + b_P) + (1-\alpha_\phi)\phi_P^0 \tag{10}$$

where superscript 0 is used for the quantities calculated at the previous iteration and α_ϕ is the under-relaxation factor for variable ϕ. Based on the above discussion, Eq. (6) can be written as:

$$A'_P\phi_P = A_E\phi_E + A_W\phi_W + A_N\phi_N + A_S\phi_S + b'_P \tag{11}$$

where $A'_P = A_P/\alpha_\phi$ and $b'_P = b_p + \frac{(1-\alpha_\phi)}{\alpha_\phi}A_P\phi_P^0$.

Let us now consider the equation for x-component of the velocity field, u. Before under-relaxation u_P would follow an equation similar to Eq. (6). If the pressure term is separated from the source we can write:

$$u_P = \frac{\sum_{NB} A_{NB}u_{NB} + b_P}{A_P} - \frac{\Delta y(p_e - p_w)}{A_P} \tag{12}$$

which can also be expressed in general form:

$$u_P = H_P - D_P(\nabla p)_P \tag{13}$$

where $H_P = \frac{\sum_{NB} A_{NB}u_{NB}+b_P}{A_P}$, $D_P = \frac{\Delta x\Delta y}{A_P}$ and $(\nabla p)_P$ is the x component of the pressure gradient in cell P. Therefore, for the two contiguous cells E and P we could write the following equations:

$$u_P = \frac{(\sum_{NB} A_{NB}u_{NB} + b_P)_P}{(A_P)_P} - \frac{\Delta y(p_e - p_w)_P}{(A_P)_P} \tag{14}$$

$$u_E = \frac{(\sum_{NB} A_{NB}u_{NB} + b_P)_E}{(A_P)_E} - \frac{\Delta y(p_e - p_w)_E}{(A_P)_E} \tag{15}$$

Mimicking formulation followed for u_E and u_P, Rhie-Chow proposed a pseudo-equation for the face velocity u_e:

$$u_e = \frac{\left(\sum_{NB} A_{NB} u_{NB} + b_P\right)_e}{(A_P)_e} - \frac{\Delta y (p_E - p_P)}{(A_P)_e} \tag{16}$$

which can also be expressed as:

$$u_e = H_e - D_e (\nabla p)_e \tag{17}$$

where H_e is the first term in the RHS of Eq. (17), $D_e = \frac{\delta x_e \Delta y}{(A_P)_e}$ and $(\nabla p)_e$ is the x component of pressure gradient in face e. In the OMIM of Rhie and Chow, unknown terms of RHS of Eq. (16) are obtained by linear interpolation as follows:

$$\left(\frac{\sum_{NB} A_{NB} u_{NB} + b_P}{A_P}\right)_e = f_e^+ \left(\frac{\sum_{NB} A_{NB} u_{NB} + b_P}{A_P}\right)_E + (1 - f_e^+) \left(\frac{\sum_{NB} A_{NB} u_{NB} + b_P}{A_P}\right)_P \tag{18}$$

$$\frac{1}{(A_P)_e} = f_e^+ \frac{1}{(A_P)_E} + (1 - f_e^+) \frac{1}{(A_P)_P} \tag{19}$$

where f_e^+ is a weighting factor that for the mesh shown in Fig. 1 can be determined as $f_e^+ = \frac{\Delta x_P}{2 \delta x_e}$. Therefore, Eq. (17) becomes:

$$u_e = \overline{H_e} - \overline{\left(\frac{1}{A_e}\right)} \Delta y (p_E - p_P) \tag{20}$$

where the over-bar denotes linear interpolation.

A similar procedure can be applied for other face velocities of cell P. For the sake of simplicity and brevity, this procedure is shown only for face e.

Substituting Eq. (18) into Eq. (16), considering Eqs. (14) and (15), and

re-ordering terms we obtain:

$$
\begin{aligned}
u_e = & \left[f_e^+ u_E + (1 - f_e^+) u_P \right] \\
& - \frac{\Delta y (p_E - p_P)}{(A_P)_e} \\
& + \left(f_e^+ \frac{\Delta y (p_e - p_w)_E}{(A_P)_E} + (1 - f_e^+) \frac{\Delta y (p_e - p_w)_P}{(A_P)_P} \right)
\end{aligned} \tag{21}
$$

The first term in the RHS of Eq. (21) corresponds to the linear interpolation of cell values, while the last two terms can be regarded as a correction term that smooths the pressure field, and removes the undesired checkerboard behavior. Assuming $D_e \approx \overline{D_e}$ and $\overline{(D \nabla p)_e} \approx \overline{D_e}\, \overline{\nabla p_e}$, and considering Eqs. (13) and (17), Eq. (21) can be re-written as:

$$
u_e = \overline{u_e} + \overline{D_e} \left(\overline{\nabla p_e} - (\nabla p)_e \right) \tag{22}
$$

which is the classical Rhie-Chow interpolation formula.

However, in general under-relaxation is required for the momentum equations. When considering under-relaxation, Eq. (14) becomes:

$$
\begin{aligned}
u_P &= \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB} + b_P \right)_P}{(A_P)_P} + (1 - \alpha_u) u_P^0 - \frac{\alpha_u \Delta y (p_e - p_w)_P}{(A_P)_P} \\
&= \alpha_u \left[H_P - \frac{\Delta y (p_e - p_w)_P}{(A_P)_P} \right] + (1 - \alpha_u) u_P^0 \\
&= h_P - \frac{\alpha_u \Delta y (p_e - p_w)_P}{(A_P)_P}
\end{aligned} \tag{23}
$$

where $h_P = \alpha_u H_P + (1 - \alpha_u) u_P^0$. Similarly, for cell E we find:

$$
u_E = h_E - \frac{\alpha_u \Delta y (p_e - p_w)_E}{(A_P)_E} \tag{24}
$$

and $h_E = \alpha_u H_E + (1 - \alpha_u) u_E^0$.

If Rhie-Chow momentum interpolation is now introduced, following Eq. (20) we can write:

$$\begin{aligned} u_e &= \overline{h_e} - \alpha_u \overline{\left(\frac{1}{A_e}\right)} \Delta y (p_E - p_P) \\ &= \left(f_e^+ h_E + (1 - f_e^+) h_P\right) - \alpha_u \overline{\left(\frac{1}{A_e}\right)} \Delta y (p_E - p_P) \end{aligned} \tag{25}$$

Using definitions of h_E and h_P in terms of H_P and H_E, u_e can be written as:

$$u_e = \alpha_u \underbrace{\left[\overline{H_e} - \overline{\left(\frac{1}{A_e}\right)} \Delta y (p_E - p_P)\right]}_{\text{Momentum interpolation}} + (1 - \alpha_u) \underbrace{\left[f_e^+ u_E^0 + (1 - f_e^+) u_P^0\right]}_{\text{Linear Interpolation}} \tag{26}$$

The above discussion makes it clear that the direct application of Rhie-Chow technique to the under-relaxed momentum equations would yield a solution that will not converge to the desired momentum interpolation (first term), but will contain a portion of linear interpolation. Therefore, when a small enough under-relaxation factor is used, the second term in Eq. (26) will be the main contribution, and pressure oscillations may re-appear, since no momentum interpolation is used. Besides, Eq. (26) shows that final solution will depend on the value of the under-relaxation factor. Majumdar [8] and Miller [9] independently realized this problem and proposed a solution to eliminate this dependency.

3. Majumdar Momentum Interpolation Method

In [8], convergence of u_e is forced to the full momentum interpolation value by applying explicit relaxation:

$$u_e = \alpha_u \left[\overline{H_e} - \overline{\left(\frac{1}{A_e}\right)} \Delta y (p_E - p_P)\right] + (1 - \alpha_u) u_e^0 \tag{27}$$

In an iterative procedure, u_e will converge to the momentum interpolation value, as described in Eq. (20). Expressing $\overline{H_e}$ in terms of H_E and H_P, and later in terms of h_E and h_P, similarly to what has been done in Eqs. (23) and (24), it follows:

$$u_e = \left[f_e^+ h_E + (1 - f_e^+) h_P\right] + \alpha_u \overline{\left(\frac{1}{A_e}\right)} \Delta y (p_E - p_P) + (1 - \alpha_u) \left[u_e^0 - f_e^+ u_E^0 - (1 - f_e^+) u_P^0\right] \tag{28}$$

where last term is commonly known as the Majumdar correction.

As pointed out by B. Yu et al. [4], face velocities are used for three different computations in the overall solution procedure, namely, (i) to determine the coefficients in the discretization of the momentum equations, (ii) to derive the pressure equation and (iii) they are required in the mass residual coefficient (if it is not dropped). Following the suggestion of [4], u_e is calculated using Eq. (28) for any of the three computations listed above in all simulations discussed in this paper.

4. Pressure-velocity coupling algorithms

Due to the simplicity of their implementation and the lower peak memory requirements, segregated pressure-velocity coupling algorithms are commonly preferred over coupled algorithms. In this section four possible pressure-velocity coupling algorithms for steady state simulations are described together with a summary of the iterative procedure followed in each of them.

4.1. SIMPLE

The SIMPLE algorithm (Semi-Implicit Method for Pressure Linked Equation) is probably one of the most widely used and it forms the basis for the derivation of the rest of models that will be explained later. The basic idea of the SIMPLE algorithm is to update velocity and pressure field in each iteration so that the continuity equation is always satisfied, and velocity equations approach their solution progressively. To accomplish this goal, a *projection method* is used. For simplicity, SIMPLE equations will be shown for the case of OMIM (Majumdar correction will not be shown in the equations).

At the beginning of the m-iteration, known pressure p^{m-1} is used to compute a velocity field u^* by solving the velocity equation in the momentum predictor step. For instance, the semi-discretized form of x-velocity is:

$$A_P u_P^* = \sum_{NB} A_{NB} u_{NB}^* + b_P - \Delta y (p_e^{m-1} - p_w^{m-1}) \tag{29}$$

where the pressure gradient contribution to the source term has been separated from the rest of source terms, represented in b_P. Typically velocity-equation would be under-relaxed, as explained in Section 2:

$$u_P^* = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^* + b_P\right)_P}{(A_P)_P} + (1-\alpha_u) u_P^{m-1} - \frac{\alpha_u \Delta y (p_e^{m-1} - p_w^{m-1})_P}{(A_P)_P} \tag{30}$$

The term $(1-\alpha_u)u_P^{m-1}$ can be added to the source term $(b_P)_P$. Eq. (30) becomes:

$$u_P^* = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^* + B_P\right)_P}{(A_P)_P} - \frac{\alpha_u \Delta y (p_e^{m-1} - p_w^{m-1})_P}{(A_P)_P} \tag{31}$$

where $(B_P)_P = (b_P)_P + (A_P)_P(1-\alpha_u)u_P^{m-1}$. Using the OMIM, the face velocities will follow an equation similar to Eq. (25):

$$u_e^* = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^* + B_P\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p_E^{m-1} - p_P^{m-1})}{(A_P)_e} \tag{32}$$

Since u^* does not satisfy continuity, velocity has to be updated using a correction u' so that $u^m = u^* + u'$. Pressure field is updated by adding a small correction that should vanish as the solution approaches convergence $p^m = p^{m-1} + p'$. Pressure does not have its natural equation in incompressible flows, therefore, an equation for p' is derived so as to guarantee that final velocity field u^m is divergence free. At convergence, an equation similar to Eq. (33) should yield:

$$u_e^m = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^m + B_P\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p_E^m - p_P^m)}{(A_P)_e} \tag{33}$$

Substracting Eq. (32) from Eq. (33) we obtain:

$$u_e' = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}'\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p_E' - p_P')}{(A_P)_e} \tag{34}$$

In the SIMPLE algorithm the first term in the RHS of Eq. (34) is neglected so that velocity correction becomes:

$$u_e' = -\frac{\alpha_u \Delta y (p_E' - p_P')}{(A_P)_e} \tag{35}$$

Neglecting the first term in the above equation can lead to convergence issues in the use of SIMPLE algorithm in some applications. Using Eq. (35), the face velocity becomes:

$$u_e^m = u_e^* - \frac{\alpha_u \Delta y (p_E' - p_P')}{(A_P)_e} \tag{36}$$

Since u_e^* is now known, following the same procedure for the other velocity component v and for the rest of the faces, one may use the continuity equation to derive an equation for p'. Eq. (1) can be integrated in the cell volume by using Green-Gauss theorem leading to:

$$\sum_e \rho \mathbf{u_e} \mathbf{S_e} = 0 \tag{37}$$

$$\Delta y(\rho u_e^m) - \Delta y(\rho u_w^m) + \Delta x(\rho v_n^m) - \Delta x(\rho v_s^m) = 0 \tag{38}$$

When Eq. (36) and similar equations for u_w^m, v_n^m and v_s^m are introduced in Eq. (38), an equation for p' is obtained. Calculation of p' allows us to correct pressure and velocity fields. For pressure correction an under-relaxation factor is usually introduced, so that new pressure becomes $p^m = p^{m-1} + \alpha_P p'$. Cell center velocities are computed by an equation derived from Eq. (36):

$$u_P^m = u_P^* - \frac{\alpha_u \Delta y(p'_e - p'_w)}{(A_P)_P} \tag{39}$$

where p'_e and p'_w are obtained by linear interpolation of cell center values.

A slight variation of the SIMPLE algorithm (hereafter referred to as *simplePcorr*) can be considered where pressure equation is formulated directly to solve for p and not for p' (referred to as *simple*). In fact, substituting u_e^* from Eq. (32) into Eq. (36):

$$u_e^m = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^* + B_P\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y(p_E^m - p_P^m)}{(A_P)_e} \tag{40}$$

where the first term in the RHS is the part of the velocity field without the pressure term. Introducing Eq. (40) into the continuity equation yields an equation for p^m. Now pressure won't be directly taken to be the calculated

value of p^m but it will be relaxed following $p^m = \alpha_p p^m + (1-\alpha_p)p^{m-1}$. New cell center velocities would be computed as:

$$u_P^m = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^* + B_P\right)}{A_P} - \frac{\alpha_u \Delta y (p_e^m - p_w^m)}{A_P} \tag{41}$$

For the simulations shown in this paper, pressure under-relaxation factor α_p has been set to $1-\alpha_u$ as recommended by [24].

4.2. SIMPLE-C

The SIMPLE-C (SIMPLE-Consistent) algorithm tries to reduce the lack of convergence of the standard SIMPLE algorithm by avoiding the fairly drastic assumption made from Eq. (34) to Eq. (35). It has shown to accelerate convergence in problems where the pressure-velocity coupling is the main source of deterrent to obtaining a solution. SIMPLE-C procedure is shown next.

From Eq. (34) we can derive the following:

$$(A_P)_e u'_e = \alpha_u \left(\sum_{NB} A_{NB} u'_{NB}\right)_e - \alpha_u \Delta y (p'_E - p'_P) \tag{42}$$

By subtracting $\sum_{NB} A_{NB} u'_e$ from both sides of the equation we obtain:

$$\left((A_P)_e - \sum_{NB} A_{NB}\right) u'_e = \alpha_u \left(\sum_{NB} A_{NB} \left(u'_{NB} - u'_e\right)\right)_e - \alpha_u \Delta y (p'_E - p'_P) \tag{43}$$

First term in the RHS of Eq. (43) is now neglected, leading to a new relation between velocity and pressure corrections:

$$u'_e = -\frac{\alpha_u \Delta y (p'_E - p'_P)}{(A_P - \sum_{NB} A_{NB})_e} \tag{44}$$

The face velocity becomes:

$$u_e^m = u_e^* - \frac{\alpha_u \Delta y (p'_E - p'_P)}{(A_P - \sum_{NB} A_{NB})_e} \tag{45}$$

By using this definition of the face velocity in the continuity equation we can again obtain a pressure equation for p'. New pressure would be obtained from under-relaxation $p^m = p^{m-1} + \alpha_p p'$ and cell center velocities are computed from an equation derived from Eq. (45):

$$u_P^m = u_P^* - \frac{\alpha_u \Delta y (p'_e - p'_w)}{(A_P - \sum_{NB} A_{NB})} \tag{46}$$

A slightly different version of the explained SIMPLE-C named (*simpleCP-corr*) can be obtained by deriving an equation for p^m and not for p' (*simpleC*). Substituting u_e^* from Eq. (32) into Eq. (45):

$$u_e^m = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^* + B_P\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p_E^{m-1} - p_P^{m-1})}{(A_P)_e} - \frac{\alpha_u \Delta y (p'_E - p'_P)}{(A_P - \sum_{NB} A_{NB})_e} \tag{47}$$

which can be transformed into:

$$\begin{aligned} u_e^m = & \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^* + B_P\right)_e}{(A_P)_e} \\ & - \alpha_u \left(\frac{1}{(A_P)_e} - \frac{1}{(A_P - \sum_{NB} A_{NB})_e} \right) \Delta y (p_E^{m-1} - p_P^{m-1}) \\ & - \frac{\alpha_u \Delta y (p_E^m - p_P^m)}{(A_P - \sum_{NB} A_{NB})_e} \end{aligned} \tag{48}$$

First two terms depend only on already calculated variables and can be computed. Inserting u_e^m into the continuity equation yields an equation for p^m. As in the SIMPLE algorithm, pressure is under-relaxed $p^m = \alpha_p p^m + (1 - \alpha_p) p^{m-1}$. New cell velocities are computed as:

$$u_P^m = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^* + B_P\right)}{A_P} - \alpha_u \left(\frac{1}{A_P} - \frac{1}{A_P - \sum_{NB} A_{NB}}\right) \Delta y (p_e^{m-1} - p_w^{m-1}) - \frac{\alpha_u \Delta y (p_e^m - p_w^m)}{A_P - \sum_{NB} A_{NB}} \tag{49}$$

4.3. PISO

PISO (Pressure Implicit with Splitting of Operators) is another derivation of the standard SIMPLE algorithm. Once velocity and pressure have been corrected with the standard simple procedure $u^{**} = u^* + u'$, $p^{**} = p^{m-1} + p'$, new corrections ($p^m = p^{**} + p''$, $u^m = u^{**} + u''$) may be considered to avoid neglecting the first term in RHS of Eq. (34) which becomes:

$$u_e'' = \frac{\alpha_u \left(\sum_{NB} A_{NB} u'_{NB}\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p''_E - p''_P)}{(A_P)_e} \tag{50}$$

Since u' has already been calculated, first term of the RHS can now be computed. Following ($u^m = u^{**} + u''$), the face velocity is now:

$$u_e^m = u_e^{**} + \frac{\alpha_u \left(\sum_{NB} A_{NB} u'_{NB}\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p''_E - p''_P)}{(A_P)_e} \tag{51}$$

Using again the continuity equation (note that u_e^{**} already satisfies continuity), Eq. (51) yields an equation for p''. Cell pressure field is then updated to $p^m = p^{**} + p''$ and cell velocities are calculated as:

$$u_P^m = u_P^{**} + \frac{\alpha_u \left(\sum_{NB} A_{NB} u'_{NB}\right)}{A_P} - \frac{\alpha_u \Delta y (p''_e - p''_w)}{A_P} \tag{52}$$

The standard PISO (known hereafter as *pisoPcorr*) can be modified so that pressure equation solves directly for p^m (*piso*). After the first corrector

step, u^{**} and p^{**} are linked by an equation similar to Eq. (40):

$$u_e^{**} = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^{*} + B_P\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p_E^{**} - p_P^{**})}{(A_P)_e} \tag{53}$$

Substitution of u_e^{**} in Eq. (51) yields:

$$u_e^{m} = \frac{\alpha_u \left(\sum_{NB} A_{NB} (u_{NB}^{*} + u'_{NB}) + B_P\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p_E^{**} - p_P^{**})}{(A_P)_e} - \frac{\alpha_u \Delta y (p''_e - p''_w)}{(A_P)_P} \tag{54}$$

which can also be expressed as:

$$u_e^{m} = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^{**} + B_P\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p_E^{m} - p_P^{m})}{(A_P)_e} \tag{55}$$

where the first term in the RHS is the part of the velocity field u^{**} not containing the pressure term. The continuity equation yields an equation for p^m, which should not be under-relaxed, and cell velocity can be then updated to:

$$u_P^{m} = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^{**} + B_P\right)}{A_P} - \frac{\alpha_u \Delta y (p_e^{m} - p_w^{m})}{A_P} \tag{56}$$

4.4. SIMPLE-R

The SIMPLE-R (SIMPLE-Revised) algorithm is another slight variation of SIMPLE-like methods. For some particular flow problems, neglecting the $\sum_{NB} A_{NB} u'_{NB}$ terms in pressure correction equation produces too large pressure corrections (which then require under-relaxation). It could also occur that initial guess of pressure field is much worse than initial guess of velocity field. Using the standard SIMPLE algorithm, a good initial velocity field can be greatly altered in the momentum predictor step if one were to use a pressure field far from the final solution. To address these situations it is convenient to set a separate equation for pressure computation, which will be

solved at the beginning of the iteration, and later construct a pressure correction equation that will only be used to correct the velocity field. Following these steps, a face velocity could be computed using u^{m-1}:

$$u_e = \frac{\alpha_u \left(\sum_{NB} A_{NB} u_{NB}^{m-1} + B_P\right)_e}{(A_P)_e} - \frac{\alpha_u \Delta y (p_E^m - p_P^m)}{(A_P)_e} \tag{57}$$

If this face velocity is used in the continuity equation we obtain an equation for p^m. It is clear that no approximation has been made here and no term has been neglected. Calculated p^m is now used to construct and solve velocity equations:

$$(A_P/\alpha_u)\, u_P^* = \sum_{NB} A_{NB} u_{NB}^* + B_P - \Delta y (p_e^m - p_w^m) \tag{58}$$

Once u^* is known, Eqs. (36) and (38) can be used to derive a pressure correction equation for p', following the exact same steps of the SIMPLE algorithm. However, in this case, pressure is not corrected (p' is not added to current estimation of p), but it is only used to correct the velocity field, following Eq. (39).

For each of the solvers presented here, the introduced names correspond to the OMIM (such as *simpleC*), while they will be referred to as "solver name" + *Majumdar* when MMIM is used (such as *simpleCMajumdar*).

5. Numerical experiments

In order to test the accuracy and convergence speed of each solver with and without Majumdar correction, two cases are selected. Simple cases are chosen in order to perform a systematic study of under-relaxation factor, mesh type and refinement.

First, a 2D lid driven cavity case with a Reynolds number of $Re = 1000$ and an orthogonal uniform mesh serves as a base test to check the theory behind Majumdar correction, since no other effect could have an influence on the final solution. This is on account of the fact that no turbulence is present and mesh can be uniform and perfectly orthogonal. The effect of mesh refinement is then analyzed for this simplified mesh. Different types of meshes (non-uniform orthogonal, triangles or unstructured hex-dominant) are used to study the impact of the meshing on the stability, accuracy and robustness of the methods. Tests are then repeated for a higher Reynolds number ($Re = 5000$) where, even though the flow is still laminar, instabilities are more likely to appear in the numerical solvers. For each of the cases, velocity profiles along cavity center-line are used to test the accuracy of the solution. The second case-study for testing the solvers involves the RANS simulation of a NACA 0012 airfoil profile. Given the shape of the airfoil, it is not possible to obtain a perfectly uniform orthogonal mesh. Furthermore, convergence of the solver might be limited by the convergence of turbulence quantities. Classical characteristic coefficients of the profile; namely pressure, lift and drag coefficients on the upper boundary, are used to compare numerical solution with experimental data.

For each of the cases analyzed here, convergence of the solvers is checked by calculating residuals for the continuity and all components of the velocity equations. Convergence is assumed to be achieved when all normalized residuals go below a given threshold. In OpenFOAM® normalized residual for a given variable ϕ following Eq. (6) are calculated as follows:

$$res_\phi = \frac{\sum_{i=1}^{N} | \left[A_P \phi_P - \sum_{NB} A_{NB} \phi_{NB} - b_P \right]_i |}{normFactor_\phi} \tag{59}$$

where the first summation is performed in all the cells and normalization factor is calculated as:

$$normFactor_\phi = \sum_{i=1}^{N} \left| \left[A_P \phi_P - \sum_{NB} A_{NB} \phi_{NB} \right]_i - \left[A_P - \sum_{NB} A_{NB} \right]_i \overline{\phi} \right| + \tag{60}$$

$$+ \sum_{i=1}^{N} \left| [b_P]_i - \left[A_P - \sum_{NB} A_{NB} \right]_i \overline{\phi} \right| \tag{61}$$

where $\overline{\phi}$ is a reference value for the field, calculated as mean value over the total number of cells ($\overline{\phi} = \sum_i [\phi_P]_i / N$). At convergence normalization factor becomes $normFactor_\phi = 2 \sum_{i=1}^{N} |[b_P]_i|$.

The continuity error is computed as a volume weighted average of the mass residual A_b calculated as explained in Eq. (9).

Simple linear matrix solvers have been used for all cases. Pressure and pressure correction equations are solved with a Preconditioned Conjugate Gradient solver (PCG), using diagonal incomplete-Cholesky preconditioner (DIC), while velocity and turbulence quantities are solved using Preconditioned Bi-conjugate Gradient solver (PBiCG) and Diagonal Incomplete LU preconditioner (DILU). Each outer iteration matrices are solved until an absolute residual of 10^{-12} or a relative residual (compared to initial residual in present iteration) of 10^{-3} is achieved.

5.1. Lid-driven cavity

Dimensionless laminar lid-driven cavity case (as shown in Fig. 2) has been studied for $Re = 1000$ and $Re = 5000$.

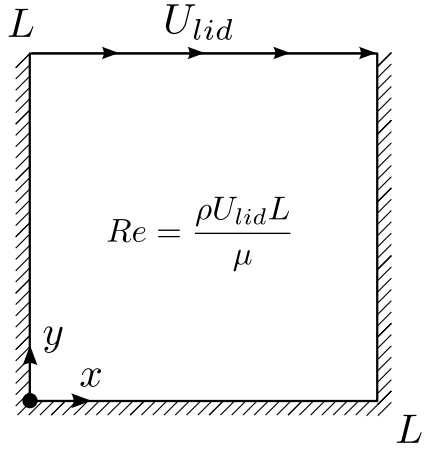


Fig. 2: Dimensionless cavity geometry

Different mesh types are shown in Fig. 3. Number of cells and non-orthogonality of each of them is shown in Table 1.

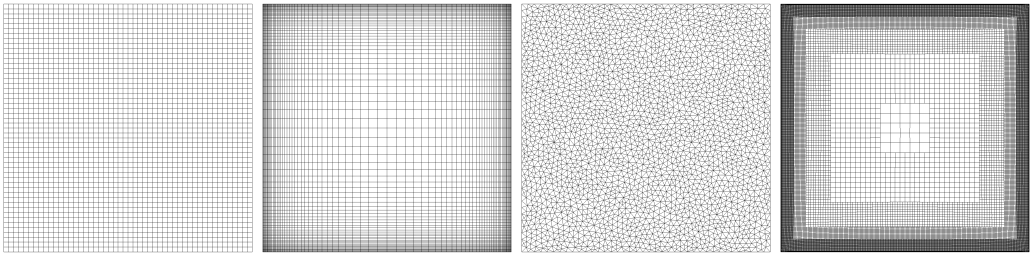


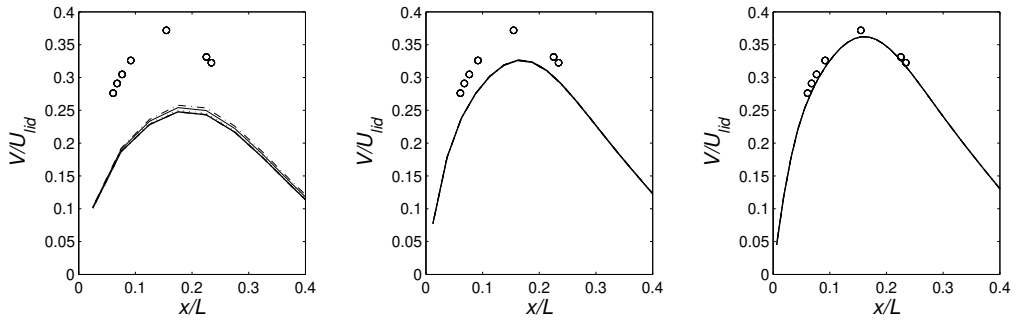
Fig. 3: Mesh types tested for cavity case; from left to right: uniform orthogonal, non-uniform orthogonal, triangles and unstructured hex-dominant

Fig. 4 compares the vertical velocity profile V/U_{lid} along the horizontal centerline for both OMIM and MMIM for different mesh refinements using the SIMPLE algorithm. Mesh refinement from 20×20 to 80×80 with

Mesh	Number of cells	Maximum non-orthogonality
Uniform orthogonal	2500	0
Non-uniform orthogonal	6400	0
Triangles	4862	37.6
Hex-dominant	48825	47.6

Table 1: Mesh characteristics for cavity case

a uniform mesh are considered. Fig. 4(a) shows that if a 20×20 mesh is used, there is a 4% difference in peak velocity between $\alpha_u = 0.2$ and 0.8. Dependency of the solution quickly reduces with mesh refinement, while Majumdar correction completely eliminates the dependency. As shown in Fig. 4, for a resolution higher than 40×40 the difference in the solution is insignificant.



(a) (b) (c)

Fig. 4: Vertical velocity profiles along horizontal center-line for OMIM (thin lines) and Majumdar correction (thick lines) for several under-relaxation factors: $\alpha_u = 0.2$ ($- \cdot -$); $\alpha_u = 0.5$ (—); $\alpha_u = 0.8$ ($\cdots$). Results are compared to benchmark solution ($\circ \circ \circ$). Mesh refinement increases from left to right: 20×20, 40×40, 80×80.

If the error in the determination of vertical velocity at point (0.1548, 0.5) (peak velocity location in the centerline for reference data) is represented with respect to the number of cells for the uniform mesh case (Fig. 5), it becomes clear that α_u dependency of OMIM is almost negligible for a fine enough mesh, while Majumdar solution is always independent of α_u.

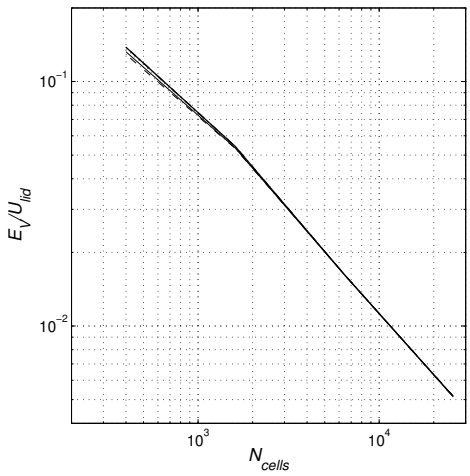


Fig. 5: Error in the determination of vertical velocity at (0.1548, 0.5) for OMIM (thin lines) and Majumdar correction (thick lines) for several under-relaxation factors: $\alpha_u = 0.2$ $(- \cdot -)$; $\alpha_u = 0.5$ (—); $\alpha_u = 0.8$ $(\cdots)$.

It was verified that for a given mesh, the solution obtained by all the pressure-velocity coupling algorithms presented in Section 4 was the same for a fixed value of α_u with OMIM, or for any value of α_u if the Majumdar correction was applied. For the case of $Re = 1000$ convergence speed of each combination algorithm - momentum interpolation method, has been checked for two possible values of α_u, 0.2 and 0.8, and for the four meshes described in Table 1. In particular, the required number of iterations to achieve a residual lower than 10^{-6} for both velocity components and continuity is represented

in Fig. 6.

Analysis of Fig. 6 shows that no significant difference in terms of speed of convergence is observed for this case and for the algorithms considered, and that Majumdar correction does not influence the convergence speed. For a given mesh, number of iterations for all solvers is approximately the same, as far as the same velocity under-relaxation is considered. Certainly, all cases with higher under-relaxation factor ($\alpha_u = 0.8$) converge faster than those with a lower under-relaxation factor ($\alpha_u = 0.2$). As a reminder, all $PISO$-derived and $SIMPLER$-derived algorithms solve twice for a pressure correction equation, and no pressure under-relaxation is performed, while for the rest of the algorithms pressure under-relaxation is set to $\alpha_p = 1 - \alpha_u$ and only one pressure correction equation is solved.

When considering the pressure correction equation, solving for pressure correction p' instead of a direct solution for p^m slightly increases the number of iterations until convergence. For a given solver, the influence of applying Majumdar correction is does not show a clear trend. For instance, when considering the non-uniform mesh, Majumdar correction accelerates convergence for $\alpha_u = 0.2$ for almost all solvers but it has a negative influence when $\alpha_u = 0.8$. Considering the hex-dominant mesh, the influence is the opposite, slightly positive for $\alpha_u = 0.8$ and slightly negative for $\alpha_u = 0.2$.

When comparing solvers, $SIMPLEC$-like solvers tend to converge slightly faster but again no clear trend is observed.

Taking into account the small differences found for $Re = 1000$, the study is now extended to the case of $Re = 5000$. Firstly, mesh convergence of standard OMIM and Majumdar correction is tested, using in this case either

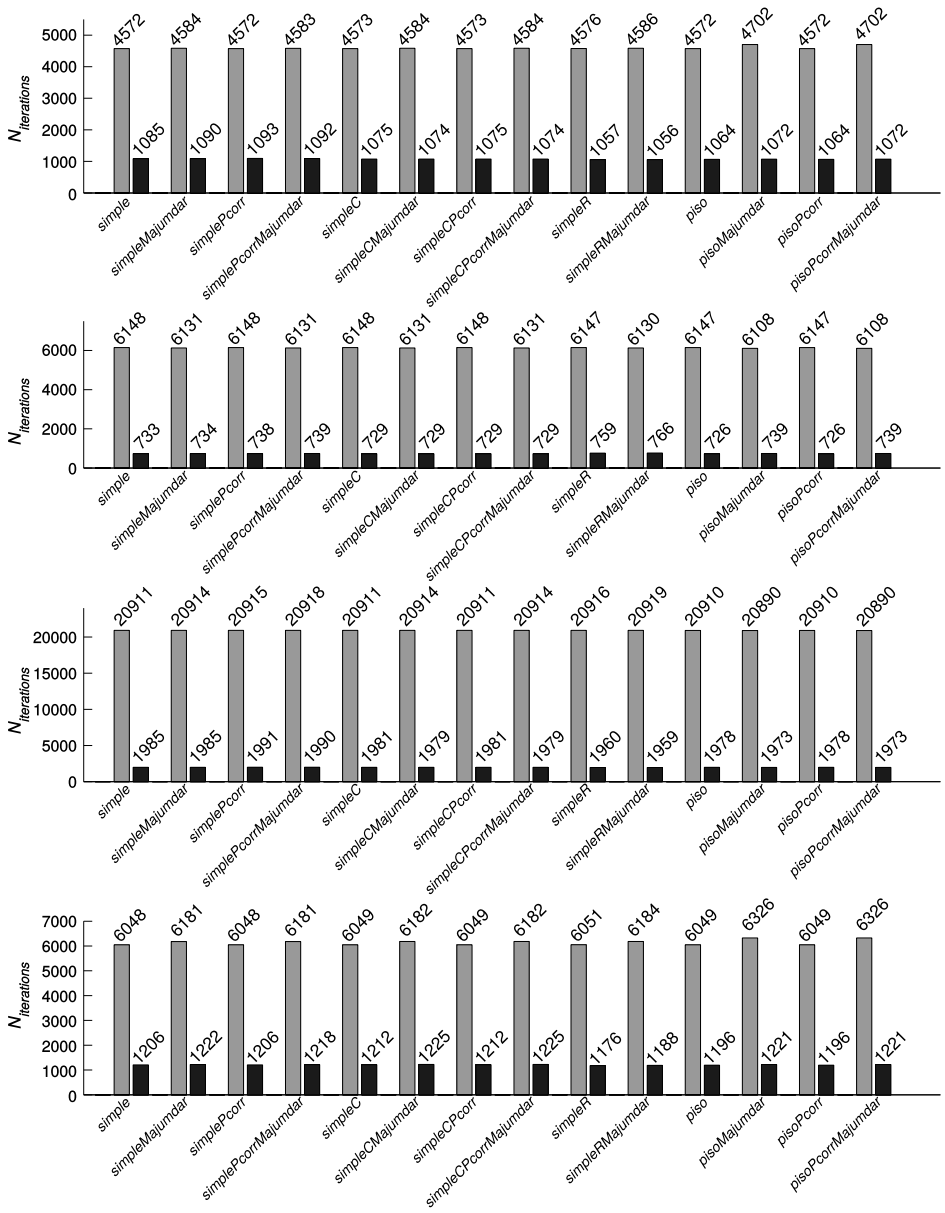
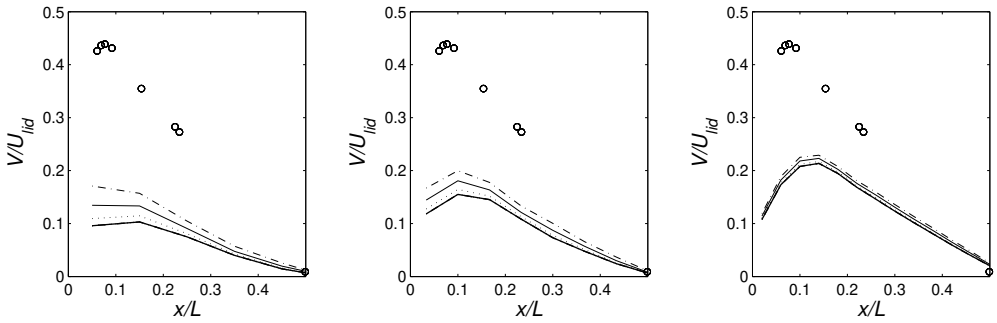


Fig. 6: Required number of iterations to reduce all residuals below 10^{-6} when α_u is fixed to $\alpha_u = 0.2$ (grey) and $\alpha_u = 0.8$ (black). Results are reported for uniform (first), non-uniform orthogonal (second), hex-dominant (third) and triangles (fourth) meshes.

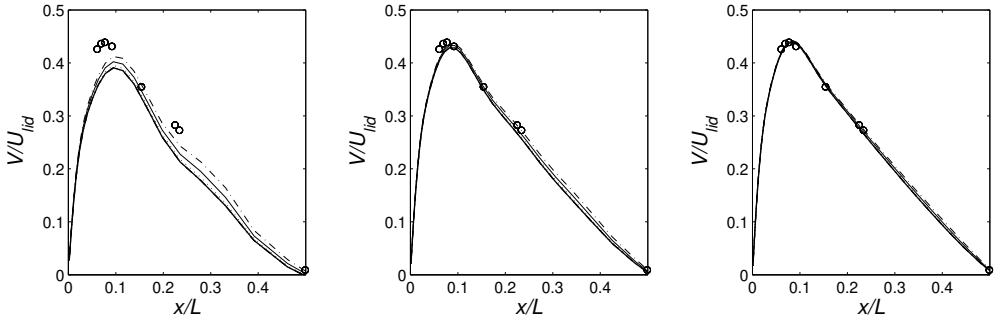
a uniform mesh or a non-uniform orthogonal mesh. For this higher Reynolds case differences are more evident. For the uniform orthogonal mesh, only coarse meshes allow solvers to converge. Fig. 7 shows how solution changes with α_u for fairly coarse uniform meshes, while Fig. 8 shows the profiles with more refined non-uniform meshes. Again, Majumdar correction completely removes the dependency as it can be seen in Fig. 9 where errors in the determination of vertical velocity at (0.06072, 0.5), corresponding to peak velocity in the experimental reference, are shown.



(a) (b) (c)

Fig. 7: Vertical velocity profiles along horizontal center-line obtained with uniform orthogonal meshes for OMIM (thin lines) and Majumdar correction (thick lines) for several under-relaxation factors: $\alpha_u = 0.2$ $(- \cdot -)$; $\alpha_u = 0.5$ $(—)$; $\alpha_u = 0.8$ $(\cdots)$. Results are compared to benchmark solution $(\circ \circ \circ)$. Mesh refinement increases from left to right: 10×10, 15×15, 25×25.

Following the procedure of the previous case, convergence speed is evaluated for the four meshes described in Table 1. As the Reynolds number increases, convergence becomes more difficult and, depending on the value of under-relaxation factor, it might not be achievable. It is also observed that, if convergence is achieved, the solution is exactly the same as long as

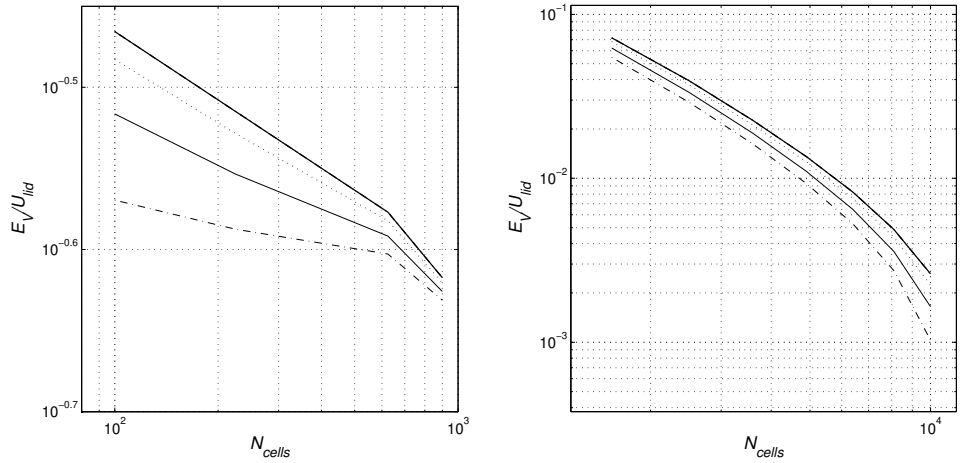


(a) (b) (c)

Fig. 8: Vertical velocity profiles along horizontal center-line obtained with non-uniform orthogonal meshes for OMIM (thin lines) and Majumdar correction (thick lines) for several under-relaxation factors: $\alpha_u = 0.2$ $(- \cdot -)$; $\alpha_u = 0.5$ $(—)$; $\alpha_u = 0.8$ $(\cdots)$. Results are compared to benchmark solution $(\circ\ \circ\ \circ)$. Mesh refinement increases from left to right: 40×40, 60×60, 80×80.

the same α_u is used for any of the algorithms, or Majumdar correction is applied. Number of iterations required to achieve convergence is represented in Fig. 10, where (-1) refers to cases that do not converge.

With the exception of few cases (*simplePcorr-simplePcorrMajumdar* in non-uniform orthogonal and hex-dominant meshes, and *simpleR-simpleRMajumdar* in hex-dominant mesh), a large velocity under-relaxation factor $\alpha_u = 0.8$ helps the convergence of the cases (which might not be achievable for $\alpha_u = 0.2$ as in the triangles mesh), or increases convergence speed (as in the non-uniform orthogonal or the hex-dominant mesh). When considering Majumdar correction applied to SIMPLE or SIMPLE-C, the general trend is that it slightly improves convergence speed, even if the effect can be opposite for some combinations of mesh-α_u. In the selection of algorithm, SIMPLE-C is the most promising, with or without Majumdar correction (considering than



(a) (b)

Fig. 9: Error in the determination of vertical velocity at (0.06070, 0.5) for OMIM (thin lines) and Majumdar correction (thick lines) for several under-relaxation factors: $\alpha_u = 0.2$ $(- \cdot -)$; $\alpha_u = 0.5$ (—); $\alpha_u = 0.8$ $(\cdots)$. Results are shown for uniform mesh (left) and non-uniform orthogonal mesh (right).

only one pressure correction equation is solved for iteration).

Summing up, when fully converged results are achieved, Majumdar correction eliminates α_u-dependency, without changing convergence speed significantly. Dependency of OMIM on α_u is quickly reduced as mesh refinement increases. When several algorithms are considered, SIMPLE-C seems the most promising alternative to the standard SIMPLE algorithm. Under these considerations effects of momentum interpolation and pressure-velocity algorithm will now be tested on a more complex geometry where coupling between velocity field and turbulence variables becomes important.

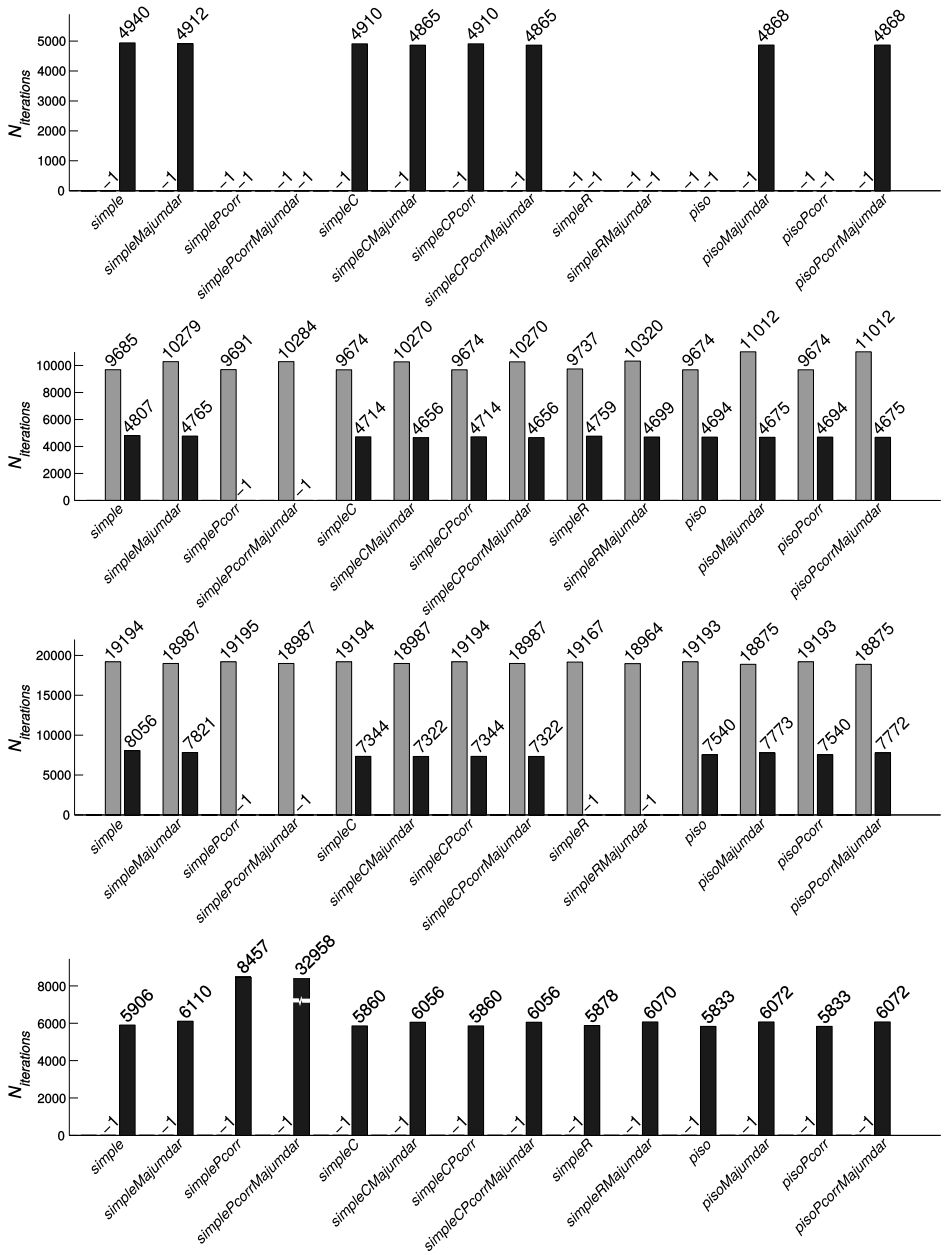


Fig. 10: Required number of iterations to reduce all residuals below 10^{-6} when α_u is fixed to $\alpha_u = 0.2$ (grey) and $\alpha_u = 0.8$ (black). Results are reported for uniform (first), non-uniform orthogonal (second), hex-dominant (third) and triangles (fourth) meshes.

5.2. NACA 0012 airfoil profile

The turbulent flow ($Re \approx 6 \cdot 10^6$ based on the chord length) around a NACA 0012 airfoil profile is studied under incompressible conditions ($M = 0.15$). As proposed in ERCOFTAC references, a structured C-grid of 14336 cells (225×65 points with 129 points on airfoil surface) is employed, giving an approximate average $y^+ \approx 1$ over the airfoil profile. Fig. 11 shows a representation of the mesh, which satisfies the characteristics shown in Table 2.

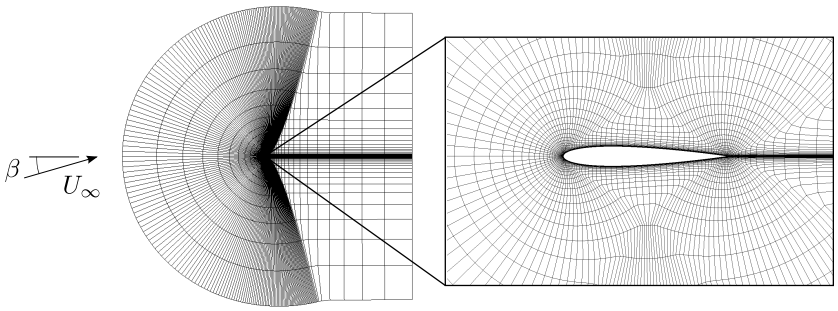


Fig. 11: Representation of the mesh used in NACA 0012 case

Number of cells	14336
Max. non-orthogonality	77.2
Avg. non-orthogonality	10.2
Max. skewness	0.62

Table 2: Mesh characteristics for cavity case

A freestream turbulence intensity of 0.052% together with a freestream turbulent viscosity $\nu_t = 0.009\nu$ are used to set the turbulence boundary conditions. First order upwind spatial discretization is used for divergence terms

of turbulence variables and LUST (Linear Upwind Stabilized Transport) is used for convective term in velocity equations. While velocity and pressure under-relaxation are set to different values during this study, turbulence variables under-relaxation factor has been fixed to 0.8.

Typical flow features are used for the evaluation of numerical solutions. In particular, experimental results for pressure distribution [25], lift and drag coefficients [26] on the upper wall for different angles of attack (β) are used as reference. The mesh used in this study is chosen after a mesh convergence analysis. Standard SIMPLE (*simple*) algorithm with OMIM and $\alpha_u = 0.7$ is used with four increasingly refined meshes; coarse (3584 cells), medium (14336 cells), fine (57344 cells) and very fine (229376 cells) leading to the drag coefficient values and pressure coefficient profiles shown in Figs. 13 and 12 for $\beta = 0$.

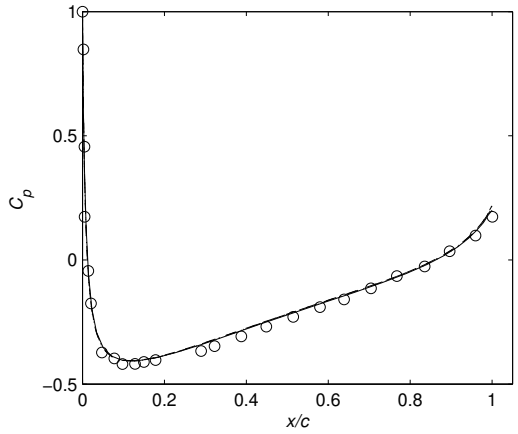


Fig. 12: Mesh convergence of pressure coefficient on upper wall: coarse(—); medium (− − −); fine(− · −); very fine (· · ·). Results are compared to experimental results (∘ ∘ ∘).

According to this, medium mesh is selected for the study, since it is the coarsest mesh for which mesh convergence is acceptable.

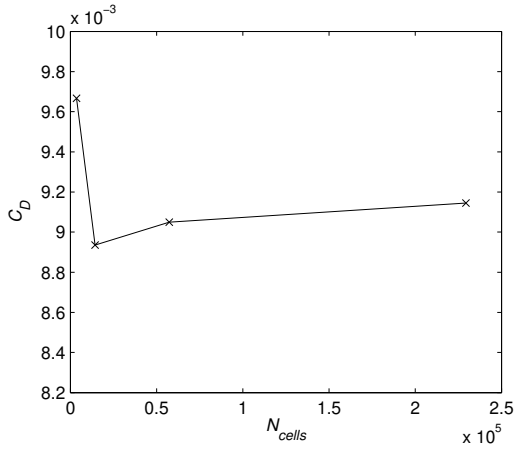
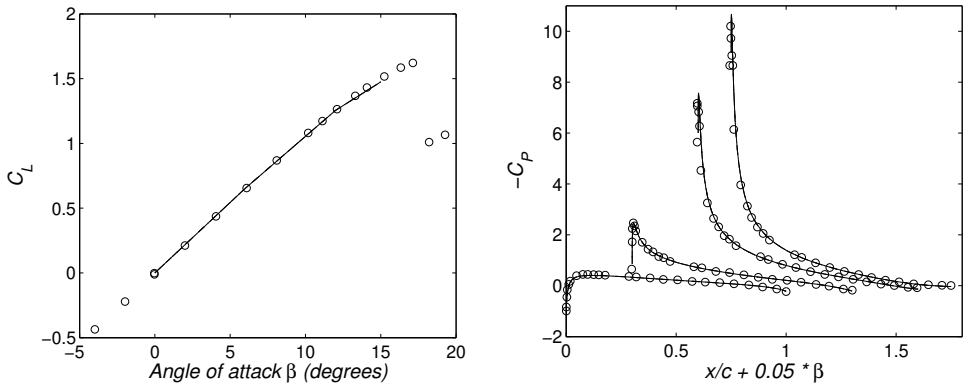


Fig. 13: Mesh convergence of drag coefficient

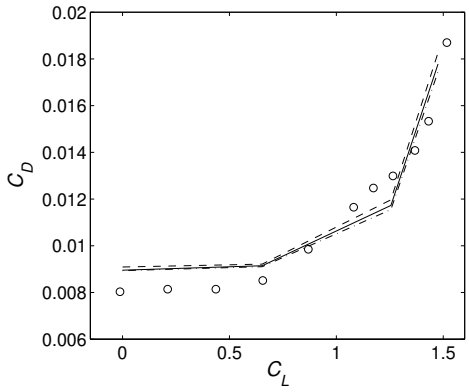
As a starting point for momentum interpolation comparison, standard SIMPLE algorithm with OMIM or MMIM is used to compute C_p, C_D and C_L for a range of angles of attack β: 0, 6, 12 and 15 degrees and for different values of α_u: 0.2, 0.5 and 0.8. Fig. 14 shows the results obtained with OMIM while Fig. 15 shows the same results when Majumdar correction is considered in the solver. For each case simulation was run until a residual lower than 10^{-10} is achieved for all variables.

While dependency of C_L (Fig. 14(a)) and C_p (Fig. 14(b)) on α_u is negligible , showing that calculated pressure field barely changes with α_u, results for C_D (Fig. 14(c)) do change more significantly. It is observed that small variations in velocity field produce larger variations in velocity gradients and therefore drag force computation. This barely affects C_L, for which pressure force is the main contribution, but it does affect C_D.

When considering the same cases with SIMPLE algorithm and MMIM, dependence of results on α_u is almost completely eliminated (see Fig. 15) for the level of convergence achieved in the simulations. In order to bet-



(a) (b)



(c)

Fig. 14: C_L, C_p and C_D coefficients SIMPLE algorithm with OMIM for several angles of attack β: 0, 6, 12 and 15 degrees. Results are shown for $\alpha_u = 0.2$ $(- - -)$; $\alpha_u = 0.5$ $(—)$; $\alpha_u = 0.8$ $(- \cdot -)$.

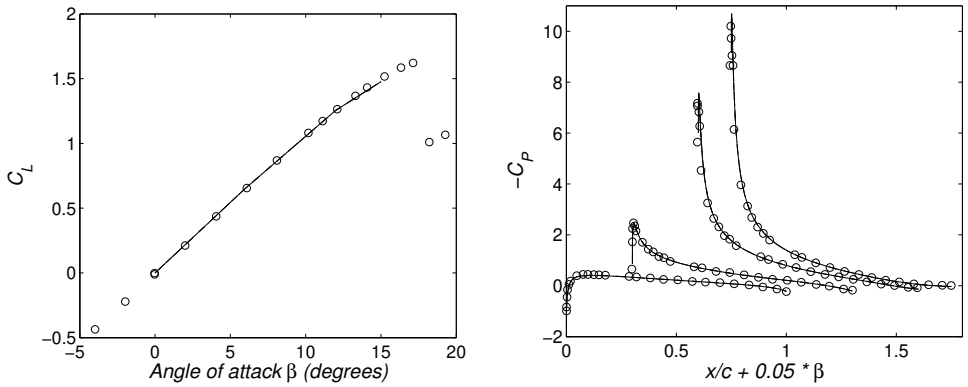
ter evaluate how much of the dependency is eliminated, a single case with $\beta = 12$ degrees (where OMIM shows the highest dependency) is considered. Calculated C_L and C_D for SIMPLE algorithm with and without Majumdar correction are shown in Table 3 and Table 4.

Coefficient		OMIM	Majumdar
C_L	$\alpha_u = 0.2$	1.2551915	1.2557273
	$\alpha_u = 0.8$	1.2552688	1.2554411
	Rel. Change	0.006 %	0.022%
C_D	$\alpha_u = 0.2$	0.011967419	0.011588629
	$\alpha_u = 0.8$	0.011548746	0.011450845
	Rel. Change	3.498 %	1.189%

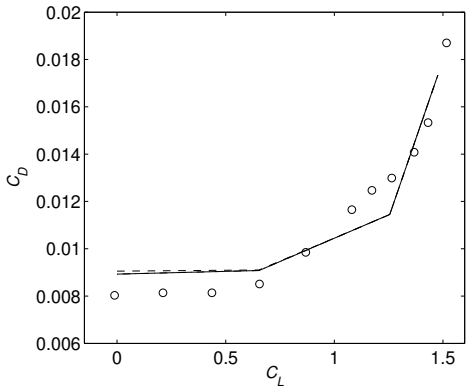
Table 3: Dependency of C_D, C_L on α_u for a convergence of $res < 10^{-7}$

Coefficient		OMIM	Majumdar
C_L	$\alpha_u = 0.2$	1.2543421	1.2554539
	$\alpha_u = 0.8$	1.2552675	1.2554399
	Rel. Change	0.133 %	0.001%
C_D	$\alpha_u = 0.2$	0.011948155	0.011446795
	$\alpha_u = 0.8$	0.011549183	0.011451306
	Rel. Change	3.339 %	0.039%

Table 4: Dependency of C_D, C_L on α_u for a convergence of $res < 10^{-12}$



(a) (b)



(c)

Fig. 15: C_L, C_p and C_D coefficients SIMPLE algorithm with Majumdar correction for several angles of attack β: 0, 6, 12 and 15 degrees. Results are shown for $\alpha_u = 0.2$ $(---)$; $\alpha_u = 0.5$ $(—)$; $\alpha_u = 0.8$ $(- \cdot -)$.

While for a standard convergence Majumdar correction only partially eliminates the α_u-dependence, when convergence is good enough, Majumdar correction reduces the relative change in predicted value by two orders of magnitude. The lack of convergence of a case due to turbulence could there-

fore limit the effect of canceling under-relaxation factor dependency of the Majumdar correction.

We proceed now to evaluate how Majumdar correction affects the convergence speed of each of the pressure-velocity coupling algorithms described in Section 4. For the case of $\beta = 12$ degrees, for each algorithm with or without Majumdar correction, the number of iterations required to achieve residuals lower than 10^{-6} and 10^{-7} is computed for two values of α_u: 0.2 and 0.8 and represented in Fig. 16.

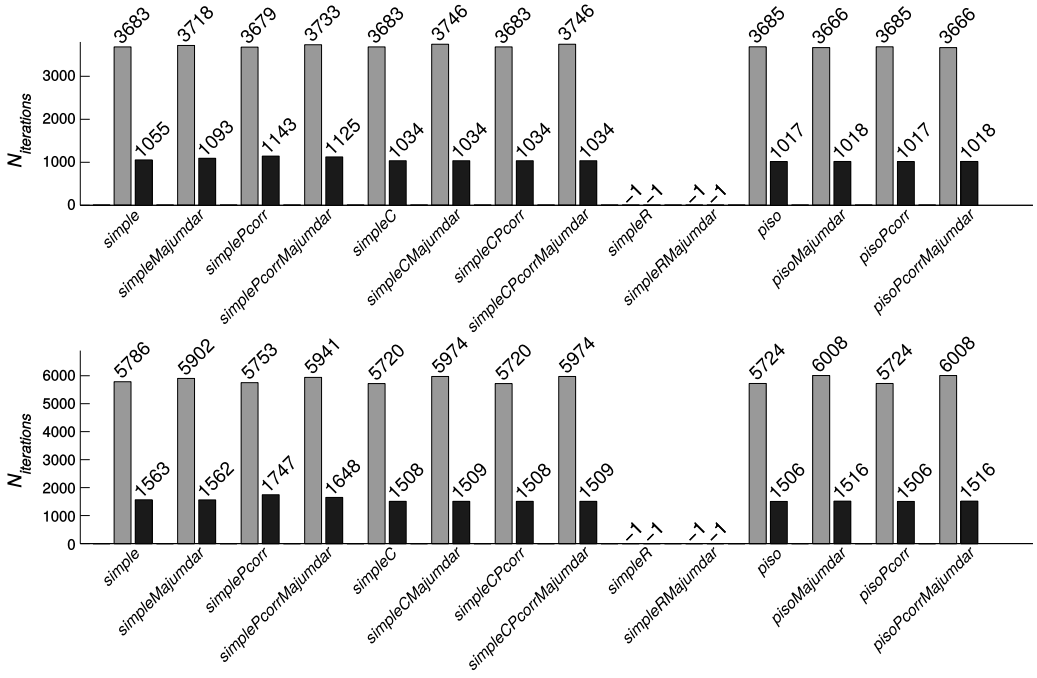


Fig. 16: Required number of iterations to reduce all residuals below 10^{-6} (top) or 10^{-7} (bottom) when α_u is fixed to $\alpha_u = 0.2$ (grey) and $\alpha_u = 0.8$ (black).

First it should be noted that SIMPLE-R algorithm did not converge to the level of residuals considered here. For both *simpleR*, *simpleRMajumdar*, minimum achieved residual for velocity equations was about 10^{-4}. When

comparing the rest of cases Majumdar correction tends to increase the required number of iterations for a low value of α_u, and has almost no effect when considering the large value of α_u. SIMPLE-C is again the best performing algorithm here for both small and large values of velocity under-relaxation factor.

6. Conclusions

Performance of MMIM when compared to OMIM is studied for several pressure-velocity coupling algorithms in OpenFOAM®. Numerical simulations confirm that Majumdar correction correctly eliminates the under-relaxation factor dependency of OMIM, with no significant influence on convergence speed. Analysis of a lid-driven cavity case shows that dependence of the solution on α_u is almost negligible for a Reynolds number of $Re = 1000$ when mesh refinement is enough to approach benchmark solution, while higher dependency is found for a larger Reynolds number $Re = 5000$. In both cases MMIM correctly eliminates the dependency but generates velocity profiles further from the benchmark profiles when grid is not as fine as that of the benchmark solution. When a more complex case as a NACA 0012 airfoil profile is considered, convergence of turbulence quantities makes full independence of solution with respect to α_u not achievable. However Majumdar correction significantly reduces the differences in computed values of drag and lift coefficients. Majumdar correction could in principle be applied to any pressure-velocity coupling algorithms. Simulations show that in general Majumdar correction does not greatly modify the number of iterations required to achieve a given level of convergence. When comparing pressure-

velocity coupling algorithms, SIMPLE-C was the best performing option for the numerical simulations carried out here.

Acknowledgments

We gratefully acknowledge the computing resources provided on Blues, a high-performance computing cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory.

References

[1] C. Rhie, W. L. Chow, A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation, AIAA J. 21 (1983) 1525–1532.

[2] S. Choi, H. Nam, M. Cho, Use of momentum interpolation method for numerical solution of incompressible flows in complex geometries: choosing cell face velocities, Numerical Heat Transfer, Part B: Fundamentals 23 (1) (1993) 21–41. doi:10.1080/10407799308914888.

[3] S. Choi, Note on the Use of Momentum interpolation Method for Unsteady Flows, Numerical Heat Transfer A: Applications 36 (5) (1991) 545–550. doi:10.1080/104077899274679.

[4] B. Yu, Y. Kawaguchi, W. Tao, H.Ozoe, Checkerboard Pressure Predictions Due to the UnderrelaxationFactor and Time Step Size for a Nonstaggered Grid with Momentum Interpolation Method, Numerical Heat Transfer B 41 (1) (2002) 85–94.

[5] B. Yu, W. Tao, J. J. W. et al, Discussion on momentum interpolation method for collocated grids of incompressible flow, Numerical Heat Transfer, Part B: Fundamentals 42 (2) (2002) 85–94. doi:10.1080/10407790190053879.

[6] W. Shen, J. A. Michelsen, J. N. Sorensen, Improved Rhie-Chow interpolation for unsteady flow computations, AIAA Journal 39 (12) (2001) 2406–2409. doi:10.2514/2.1252.

[7] S. Choi, Use of the momentum interpolation method for flows with large body force, Numerical Heat Transfer, Part B: Fundamentals 43 (3) (2003) 267–287. doi:10.1080/713836204.

[8] S. Majumdar, Role of underrelaxation in momentum interpolation for calculation of flow with nonstaggered grids, Numerical Heat Transfer 13 (1988) 125–132.

[9] T. Miller, F. Schmidt, Use of a Pressure-Weighted Interpolation Method for the Solution of Incompressible Navier-Stokes Equations on a Non-Staggered Grid System, Numerical Heat Transfer 14 (1988) 213–233.

[10] S. Patankar, D. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, International Journal of Heat and Mass Transfer 15 (1971) 1787–1806.

[11] S. Patankar, A calculation procedure for two dimensional elliptic situations, Numerical Heat Transfer 14 (1984) 409–425.

[12] J. V. Doormaal, G. Raithby, Enhancements of the SIMPLE method for

predicting incompressible fluid flows, Numerical Heat Transfer 7 (1984) 147–163.

[13] R. Issa, A. Gosman, A. Watkins, The computation of compressible and incompressible recirculating flows by a non-iterative implicit scheme, Journal of Computational Physics 62 (1986) 66–82.

[14] OpenCFD, OpenFOAM Programmer's Guide, Springer, 2004.

[15] A. Montorfano, F. Piscaglia, A. Onorati, A LES Study on the Evolution of Turbulent Structures in Moving Engine Geometries by an Open-Source CFD Code, SAE Technical Paper 2014-01-1147 (2014). doi:10.4271/2014-01-1147.

[16] F. Piscaglia, A. Montorfano, A. Onorati, Development of a non-reflecting boundary condition for multidimensional nonlinear duct acoustic computation, Journal of Sound and Vibration 332 (4) (2013) 922–935. doi:10.1016/j.jsv.2012.09.030.

[17] F. Piscaglia, A. Montorfano, A. Onorati, F. Brusiani, Boundary Conditions and SGS Models for LES of Wall-Bounded Separated Flows: An Application to Engine-Like Geometries, Oil and Gas Science and Technology - Rev. IFP Energies nouvelles 69 (1) (2014) 11–27. doi:10.2516/ogst/2013143.

[18] F. Piscaglia, A. Montorfano, A. Onorati, Towards the LES Simulation of IC Engines with Parallel Topologically Changing Meshes, SAE International Journal of Engines 6 (2) (2013) 926–940. doi:10.4271/2013-01-1096.

[19] F. Piscaglia, A. Montorfano, A. Onorati, Development of Fully-Automatic Parallel Algorithms for Mesh Handling in the OpenFOAM-2.2.x Technology, SAE Technical Paper 2013-24-0027 (2013). doi:10.4271/2013-24-0027.

[20] F. Piscaglia, A. Montorfano, A. Onorati, A Scale Adaptive Filtering Technique for Turbulence Modeling of Unsteady Flows in IC Engines, SAE Paper 2015-01-0395. To appear in SAE International Journal of Engines, April 2015.

[21] A. Montorfano, F. Piscaglia, A. Onorati, An extension of the dynamic mesh handling with topological changes for LES of ICE in OpenFOAM®, in proceedings of SAE 2015 World Congress & Exhibition, 2015.

[22] J. Martínez, F. Piscaglia, A. Montorfano, A. Onorati, S. M. Aithal, Influence of spatial discretization schemes on accuracy of explicit LES: Canonical problems to engine-like geometries, Computers & Fluids 117 (2015) 62–78. doi:10.1016/j.compfluid.2015.05.007.

[23] B. Leonard, A Stable and Accurate Convective Modelling Procedure Based on Quadratic Upstream Interpolation, Comput. Meth. Appl. Mech. Eng. 19 (1979) 59–98.

[24] J. H. Ferziger, M. Perić, Computational Methods for Fluid Dynamics, 3rd Edition, Springer, 2002.

[25] N. Gregory, C. L. O'Reilly, Low-Speed Aerodynamic Characteristics of

NACA 0012 Aerofoil Sections, including the Effects of Upper-Surface Roughness Simulation Hoar Frost, NASA R&M 3726 (1970).

[26] C. L. Ladson, A. S. Hill, W. G. Johnson, Pressure Distributions from high Reynolds Number Transonic Tests of a NACA 0012 Airfoil in the Langley 0.3 Meter Transonic Cryogenic Tunnel, NASA TM 100826 (1987).