

DEVELOPMENT OF A NOVEL FULLY COUPLED SOLVER IN OPENFOAM: STEADY-STATE INCOMPRESSIBLE TURBULENT FLOWS

L. Mangani¹, M. Buchmayr², and M. Darwish³

¹Hochschule Luzern, Technik und Architektur, Horw, Switzerland

²Andritz AG, Graz, Austria; Department of Thermal Turbomachinery, TU Graz, Graz, Austria

³Department of Mechanical Engineering, American University of Beirut, Beirut, Lebanon

In this work a block coupled algorithm for the solution of three-dimensional incompressible turbulent flows is presented. A cell-centered finite-volume method for unstructured grids is employed. The interequation coupling of the incompressible Navier-Stokes equations is obtained using a SIMPLE-type algorithm with a Rhie-Chow interpolation technique. Due to the simultaneous solution of momentum and continuity equations, implicit block coupling of pressure and velocity variables leads to faster convergence compared to classical, loosely coupled, segregated algorithms of the SIMPLE family of algorithms. This gain in convergence speed is accompanied by an improvement in numerical robustness. Additionally, a two-equation eddy viscosity turbulence model is solved in a segregated fashion. The substantially improved performance of the block coupled approach compared to the segregated approach is demonstrated in a set of test cases. It is shown that the scalability of the coupled solution algorithm with increasing numbers of cells is nearly linear. To achieve this scalability, an algebraic multigrid solver for block coupled systems of equations has been implemented and used as linear solver for the system of block equations. The presented algorithm has been entirely embedded into the leading open-source computational fluid dynamics (CFD) library OpenFOAM.

1. INTRODUCTION

Since resolving the pressure and velocity coupling is essential for the performance of any computational fluid dynamics (CFD) code, a lot of effort continues to be directed toward the development of more robust and more efficient coupling algorithms [11, 12]. Over the past decades the pressure-based approach based on the SIMPLE family of algorithms [1, 2, 13, 14] has become the predominant methodology used in the CFD community. The SIMPLE algorithm basically follows a segregated approach in resolving the pressure–velocity coupling, i.e., solving the momentum equation in a predictor step, followed by solving a pressure equation in

Received 13 November 2013; accepted 20 January 2014.

Address correspondence to L. Mangani, Hochschule Luzern, Technik und Architektur, CH-6048 Horw, Switzerland. E-mail: luca.mangani@hslu.ch

Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/unhb.

NOMENCLATURE

A, a	coefficient matrix, coefficient matrix coefficient	$\mathbf{u}$	velocity vector
$\mathbf{b}, b$	source vector, source vector coefficient	$V, \dot{V}$	volume scalar, volume flux scalar
$\mathbf{D}$	Rhie-Chow numerical dissipation tensor	ν	kinematic viscosity scalar
g	geometric interpolation weighting factor	ρ	density constant
k	turbulence kinetic energy	ϕ	general scalar quantity, solution vector
p	pressure	ω	turbulence frequency
$S, \mathbf{S}$	surface scalar, surface normal vector	Superscripts	
u, v, w	velocity components	n	current iteration
		u, v, w	refers to velocity components
		$\bar{\phi}$	linear interpolation to the face

a corrector step. Variants of the SIMPLE algorithm [3–6] have been developed to simulate a variety of fluid flows, increasingly expanding the reach of the method. However, one area in which the SIMPLE algorithm is deficient is in its lack of scalability with mesh size [7]. This shortcoming is inherent and is due in part to the underrelaxation needed to stabilize the algorithm. This relaxation is akin to forcing a pseudo-computational time step onto the numerical simulation that is proportional to the cell volume at hand [8]. Hence, as the grid is refined, the computational pseudo-time step is reduced, thus insuring that the number of iterations needed to resolve the same physical problem is increased. Thus the number of iterations to convergence increases with mesh size somewhat proportionally to the inverse of the average element volumes in the computational domain—a behavior somewhat resembling the performance of iterative solvers with increasing mesh size.

One algorithm that addresses this deficiency is the fully coupled algorithm of Darwish et al. [9]. In their work they show that by accounting for the pressure–velocity coupling more comprehensively, the fully coupled algorithm gains in stability and robustness and avoids using implicit underrelaxation. Thus no constraint is placed on the pseudo-time step, which can be retained at a constant value regardless of the mesh size. This basically allows for retention of performance as the mesh size is increased, as demonstrated in a number of 2-D laminar test cases [9]. Note that in the fully coupled approach, the algebraic equations resulting from the Navier-Stokes equations are solved simultaneously. To achieve good computational performance an algebraic multigrid solver for block coupled systems of equations has been implemented and used as linear solver for the discretized equations.

In this work we propose to extend the methodology to 3-D turbulent industrial applications and implement the algorithm within the context of the widely used OpenFOAM [10] open-source library.

In what follows the discretization procedure of the fully coupled algorithm and its implementation are presented, then its performance is evaluated in four test cases, three of these cases originating from industry. Therein the mesh size scalability is evaluated for a range of mesh sizes and the coupled solver’s performance in terms of computational time is compared to that of a state-of-the-art segregated SIMPLE-C solver, also

based on the OpenFOAM library, that was presented by Casartelli and Mangani [15]. As closure model of turbulence fluctuations the k - ω SST model [16] is used.

2. THE GOVERNING EQUATIONS

The basic equations governing incompressible steady-state flows are the conservation of mass and momentum equations:

$$\nabla \cdot \mathbf{u} = 0 \quad (1)$$

$$\nabla \cdot (\mathbf{u}\mathbf{u}) = -\frac{1}{\rho} \nabla p + \nabla \cdot [\nu_{\text{eff}}(\nabla \mathbf{u})] \quad (2)$$

The density field in the context of isothermal incompressible flow is constant. The laminar kinematic viscosity ν is summed up with the turbulent kinematic viscosity ν_t , yielding ν_{eff} , which will also account for the turbulent stresses arising from the Reynolds averaged eddy viscosity turbulence model. The well-known k - ω shear stress transport (SST) model, of Menter [17], is used for closure of turbulence quantities. For convenience the model is written in the following form:

$$\nabla \cdot (\mathbf{u}k) - \nabla \cdot [(\nu + \nu_t \alpha_K) \nabla k] = \frac{1}{\rho} P_k - \beta^* \omega k \quad (3)$$

$$\nabla \cdot (\mathbf{u}\omega) - \nabla \cdot [(\nu + \nu_t \alpha_\omega) \nabla \omega] = \frac{C_1 P_k}{\mu_t} - C_2 \omega^2 + \frac{2\alpha_\varepsilon(1 - F_1)}{\omega} \nabla k \cdot \nabla \omega \quad (4)$$

3. RESOLVING THE PRESSURE-VELOCITY COUPLING

To avoid forming a saddle-point matrix as a result [18] of the direct discretization of the Navier-Stokes equations, a special treatment is needed for the pressure field. This basically takes the form of a reformulation of the continuity equation into a constraint pressure equation that enforces mass conservation on the velocity fields. This procedure is basically at the core of the SIMPLE family of algorithms [19]. For a collocated grid arrangement, a special velocity interpolation is also needed to overcome any checkerboarding of the pressure field. These issues have been widely addressed over the years [20] and will be only briefly outlined. Still a distinguishing feature of this OpenFOAM based solver is the fully implicit algorithm used to resolve the velocity–pressure coupling that arises from the Navier-Stokes equations. The algorithm was originally presented by Darwish et al. [9] and is implemented within the OpenFOAM framework with minor modifications. Also, the implementation of the turbulence model is enhanced to allow consistent behavior in combination with the coupled solver. In what follows the momentum and continuity equations will be discretized.

3.1. Discretization of the Momentum Equations

Reformulating the momentum equations (2) in integral form yields

$$\oint_S \mathbf{n} \cdot (\mathbf{u}\mathbf{u})_f dS = -\frac{1}{\rho} \oint_S \mathbf{n} p_f dS + \oint_S \left[\mathbf{n} \cdot \left(\nu_{\text{eff}}(\nabla \mathbf{u})_f \right) \right] dS \quad (5)$$

As we are dealing with polygonal elements, the integrals can be evaluated using the midpoint rule over the faces of the elements to yield

$$\sum_{\text{faces}} \dot{V}_f \mathbf{u}_f + \frac{1}{\rho} \sum_{\text{faces}} \mathbf{S}_f p_f - \frac{1}{\rho} \sum_{\text{faces}} \mathbf{S}_f \cdot (\nu_{\text{eff}} \nabla \mathbf{u}_f) = 0 \quad (6)$$

The convection term in Eq. (5) is linearized by computing the convecting flux ($\dot{V}_f = \mathbf{n} \cdot \mathbf{u}_f dS$) using previous iteration values.

Starting with the first term (convection), and using a first-order upwind discretization, we get

$$\begin{aligned} a_C^{uu} &= |\dot{V}_f^n, 0| & a_{NB}^{uu} &= -|\dot{V}_f^n, 0| \\ a_C^{vv} &= |\dot{V}_f^n, 0| & a_{NB}^{vv} &= -|\dot{V}_f^n, 0| \\ a_C^{ww} &= |\dot{V}_f^n, 0| & a_{NB}^{ww} &= -|\dot{V}_f^n, 0| \end{aligned}$$

In the second term (pressure gradient), a linear interpolation is used to express the face pressure in terms of the two cell values straddling the face under concern. With g_f representing the interpolation weight,

$$\begin{aligned} a_C^{up} &= \frac{1}{\rho} S_{f_x} g_f & a_{NB}^{up} &= \frac{1}{\rho} S_{f_x} (1 - g_f) \\ a_C^{vp} &= \frac{1}{\rho} S_{f_y} g_f & a_{NB}^{vp} &= \frac{1}{\rho} S_{f_y} (1 - g_f) \\ a_C^{wp} &= \frac{1}{\rho} S_{f_z} g_f & a_{NB}^{wp} &= \frac{1}{\rho} S_{f_z} (1 - g_f) \end{aligned}$$

The third term (stress) is rewritten in terms of an implicit orthogonal component and an explicit nonorthogonal component, following the treatment of Darwish [9].

$$\mathbf{S}_f \cdot [\nu_{\text{eff}} (\nabla \mathbf{u})_f] = \nu_{\text{eff}} \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} (\mathbf{u}_{NB} - \mathbf{u}_C) + \nu_{\text{eff}} \underbrace{\left(\mathbf{S}_f - \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{S}_f \cdot \mathbf{d}} \mathbf{d} \right)}_{\mathbf{T}} \cdot \overline{\nabla \mathbf{u}_f} \quad (7)$$

The orthogonal part in Eq. (7) is written into the coefficients, while the second part is written into the right-hand side. Thus we get

$$\begin{aligned} a_C^{uu} &= \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} & a_{NB}^{uu} &= -\nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} \\ a_C^{vv} &= \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} & a_{NB}^{vv} &= -\nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} \\ a_C^{ww} &= \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} & a_{NB}^{ww} &= -\nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} \end{aligned}$$

$$\begin{aligned}
b_C^u &= \nu_{\text{eff}}^n \left(T_x \frac{\overline{\partial u}^n}{\partial x_f} + T_y \frac{\overline{\partial u}^n}{\partial y_f} + T_z \frac{\overline{\partial u}^n}{\partial z} \right) \\
b_C^v &= \nu_{\text{eff}}^n \left(T_x \frac{\overline{\partial v}^n}{\partial x_f} + T_y \frac{\overline{\partial v}^n}{\partial y_f} + T_z \frac{\overline{\partial v}^n}{\partial z_f} \right) \\
b_C^w &= \nu_{\text{eff}}^n \left(T_x \frac{\overline{\partial w}^n}{\partial x_f} + T_y \frac{\overline{\partial w}^n}{\partial y_f} + T_z \frac{\overline{\partial w}^n}{\partial z_f} \right)
\end{aligned}$$

The gradient $\overline{\nabla \mathbf{u}}_f$ is evaluated from the previous field values.

We shall now write the momentum equation's coefficient for each cell in a way that makes the subsequent derivation of the Rhie-Chow [20] interpolation technique clearer.

$$\begin{aligned}
& \begin{bmatrix} |\dot{V}_f, 0| + \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} & 0 & 0 \\ 0 & |\dot{V}_f, 0| + \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} & 0 \\ 0 & 0 & |\dot{V}_f, 0| + \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} \end{bmatrix} \cdot \begin{bmatrix} u_C \\ v_C \\ w_C \end{bmatrix} \\
& + \sum_{\text{faces}} \begin{bmatrix} -|\dot{V}_f, 0| - \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} & 0 & 0 \\ 0 & -|\dot{V}_f, 0| - \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} & 0 \\ 0 & 0 & -|\dot{V}_f, 0| - \nu_{\text{eff}}^n \frac{\mathbf{S}_f \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} \end{bmatrix} \\
& \cdot \begin{bmatrix} u_{NB} \\ v_{NB} \\ w_{NB} \end{bmatrix} + V_C \nabla p_C = \begin{bmatrix} \nu_{\text{eff}}^n \left(T_x \frac{\overline{\partial u}^n}{\partial x_f} + T_y \frac{\overline{\partial u}^n}{\partial y_f} + T_z \frac{\overline{\partial u}^n}{\partial z} \right) \\ \nu_{\text{eff}}^n \left(T_x \frac{\overline{\partial v}^n}{\partial x_f} + T_y \frac{\overline{\partial v}^n}{\partial y_f} + T_z \frac{\overline{\partial v}^n}{\partial z_f} \right) \\ \nu_{\text{eff}}^n \left(T_x \frac{\overline{\partial w}^n}{\partial x_f} + T_y \frac{\overline{\partial w}^n}{\partial y_f} + T_z \frac{\overline{\partial w}^n}{\partial z_f} \right) \end{bmatrix}
\end{aligned} \tag{8}$$

A short notation for Eq. (8) is

$$\mathbf{a}_C \cdot \mathbf{u}_C + \sum_{\text{faces}} \mathbf{a}_{NB} \cdot \mathbf{u}_{NB} + \frac{V_C}{\rho} \nabla p_C = \mathbf{b}_C^u \tag{9}$$

or

$$\mathbf{u}_C + \mathbf{a}_C^{-1} \cdot \mathbf{a}_{NB} \cdot \mathbf{u}_{NB} + \mathbf{a}_C^{-1} \cdot \left(\frac{V_C}{\rho} \nabla p_C \right) = \mathbf{a}_C^{-1} \cdot \mathbf{b}_C^u \tag{10}$$

This leads to the momentum equation written in operator form.

$$\mathbf{u}_C + \mathbf{H}_C(\mathbf{u}) + \mathbf{D}_C \cdot \nabla p_C = \tilde{\mathbf{b}}_C^u \tag{11}$$

3.2. Discretization of the Continuity Equations

The continuity equation (1) in integral form reads

$$\oint_S \mathbf{n} \cdot \mathbf{u} dS = 0 \quad (12)$$

Again integrating over the faces of our element yields

$$\sum_{\text{faces}} \mathbf{u}_f \cdot \mathbf{S}_f = 0 \quad (13)$$

$\mathbf{u}_f$ represents the face value of the velocity field. In a staggered grid, this would be obtained directly from the algebraic form of the momentum equations. In a collocated framework, the velocity at the face is obtained by reconstructing a pseudo-momentum equation at the face. This is basically the function of the Rhie-Chow interpolation [20]. We shall start from Eq. (11):

$$\mathbf{u}_f + \mathbf{H}_f(\mathbf{u}) + \mathbf{D}_f \cdot \nabla p_f = \tilde{\mathbf{b}}_C^u \quad (14)$$

where the tensor $\mathbf{D}_f(\mathbf{u})$ at a cell face is assumed to be approximately the adjacent cells' value of $\mathbf{D}$ interpolated to the face.

$$\mathbf{D}_f(\mathbf{u}) \approx \overline{\mathbf{D}_f(\mathbf{u})} \quad (15)$$

Making the same assumption for the $\mathbf{H}_f(u)$ operator gives

$$\mathbf{H}_f(\mathbf{u}) \approx \overline{\mathbf{H}_f(\mathbf{u})} \approx -\overline{\mathbf{u}}_f - \overline{\mathbf{D}_f \cdot \nabla p_f} + \tilde{\mathbf{b}}_C^u \quad (16)$$

Substituting into Eq. (11) we get

$$\mathbf{u}_f - \overline{\mathbf{u}}_f - \overline{\mathbf{D}_f \cdot \nabla p_f} + \mathbf{D}_f \cdot \nabla p_f = \underbrace{\tilde{\mathbf{b}}_C^u - \tilde{\mathbf{b}}_C^u}_{\approx 0} \quad (17)$$

or the more standard form

$$\mathbf{u}_f = \overline{\mathbf{u}}_f - \overline{\mathbf{D}_f} \cdot (\nabla p_f - \overline{\nabla p_f}) \quad (18)$$

Substituting this equation into the continuity equation (1), we get

$$\sum_{\text{faces}} \mathbf{S}_f \cdot \left[\overline{\mathbf{u}}_f - \overline{\mathbf{D}_f} \cdot (\nabla p_f - \overline{\nabla p_f}) \right] = 0 \quad (19)$$

The velocity part of Eq. (19) yields the following implicit coefficients:

$$\begin{aligned} a_C^{pu} &= S_{f_x} (1 - g_f) & a_{NB}^{pu} &= S_{f_x} g_f \\ a_C^{pv} &= S_{f_y} (1 - g_f) & a_{NB}^{pv} &= S_{f_y} g_f \\ a_C^{pw} &= S_{f_z} (1 - g_f) & a_{NB}^{pw} &= S_{f_z} g_f \end{aligned}$$

The implicit pressure gradient part is discretized similar to the viscous term of the continuity equations (6); the interpolated pressure gradient part is treated purely explicitly. Again, sublooping will lead to a converged solution of the system. Note that the Rhie-Chow diffusion part will not vanish completely for a converged solution, since the terms are not discretized equally. However, with decreasing mesh size the remainder tends to zero.

Since the method is based on unstructured grids, the implicit pressure gradient has to be split into an implicit part along the line connecting two neighboring cell centroids and a correction part that has to be evaluated explicitly,

$$\begin{aligned}
 -\mathbf{S}_f \cdot \overline{\mathbf{D}_f} \cdot \nabla p_f &= -\frac{\mathbf{S}_f \cdot \overline{\mathbf{D}_f} \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} (p_{NB} - p_C) \\
 &\quad - \underbrace{\left(\mathbf{S}_f \cdot \overline{\mathbf{D}_f} - \frac{\mathbf{S}_f \cdot \overline{\mathbf{D}_f} \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} \mathbf{d} \right)}_{\mathbf{N}} \cdot \overline{\nabla p_f} \\
 a_C^{uu} &= \frac{\mathbf{S}_f \cdot \overline{\mathbf{D}_f} \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} & a_{NB}^{uu} &= -\frac{\mathbf{S}_f \cdot \overline{\mathbf{D}_f} \cdot \mathbf{S}_f}{\mathbf{d} \cdot \mathbf{S}_f} \\
 b_C^p &= \mathbf{N} \cdot \overline{\nabla p_f}
 \end{aligned} \tag{20}$$

The explicit pressure gradient of Eq. (19) yields

$$b_C^p = -\mathbf{S}_f \overline{\mathbf{D}_f} \cdot \overline{\nabla p_f}$$

A more detailed description of the Laplacian discretization for unstructured, nonorthogonal, collocated grids is given by Muzaferija [21] and by Ferziger [22].

The obtained discretized block coupled system of equations now contains extradiagonal elements, for both diagonal and off-diagonal block coefficients. For the sake of brevity the block coefficients are written down such that a surface integration over a cell is assumed, the cell C sharing its faces with neighboring cells NB . Like this, the block coefficients a_C are directly added to the diagonal block coefficient array, whereas the neighboring block coefficients a_{NB} are injected into the off-diagonal block coefficient arrays.

Equation (21) shows the resulting block coefficient filling.

$$\begin{bmatrix} a_C^{uu} & a_C^{uv} & a_C^{uw} & a_C^{up} \\ a_C^{vu} & a_C^{vv} & a_C^{vw} & a_C^{vp} \\ a_C^{wu} & a_C^{wv} & a_C^{ww} & a_C^{wp} \\ a_C^{pu} & a_C^{pv} & a_C^{pw} & a_C^{pp} \end{bmatrix} \cdot \begin{bmatrix} u_C \\ v_C \\ w_C \\ p_C \end{bmatrix} + \sum_{\text{faces}} \begin{bmatrix} a_{NB}^{uu} & a_{NB}^{uv} & a_{NB}^{uw} & a_{NB}^{up} \\ a_{NB}^{vu} & a_{NB}^{vv} & a_{NB}^{vw} & a_{NB}^{vp} \\ a_{NB}^{wu} & a_{NB}^{wv} & a_{NB}^{ww} & a_{NB}^{wp} \\ a_{NB}^{pu} & a_{NB}^{pv} & a_{NB}^{pw} & a_{NB}^{pp} \end{bmatrix} \cdot \begin{bmatrix} u_{NB} \\ v_{NB} \\ w_{NB} \\ p_{NB} \end{bmatrix} = \begin{bmatrix} b_C^u \\ b_C^v \\ b_C^w \\ b_C^p \end{bmatrix} \tag{21}$$

4. BOUNDARY CONDITIONS

The most common boundary conditions such as the von Neumann or Dirichlet boundary conditions for single primitive variables are implemented identically to those in segregated algorithms. Boundary conditions that act on various primitive

variables at a time, such as the total pressure boundary condition or a wall boundary condition, have to be treated implicitly in order to preserve the benefit of block coupling. Derivations of such boundary conditions can be found in [9, 23].

5. LINEAR SOLVER

Once the partial differential equations have been discretized and assembled into the sparse block matrix structure, they are ready to be solved. It is essential that the equations are solved efficiently, since the matrix system contains 16 times the number of entries than result from the discretization of one equation on the same mesh. This means that a linear solver that does not scale linearly with the number of cells would drastically affect the overall convergence, and the gain that we wish to obtain from block coupling would be basically offset.

Multigrid methods as introduced by Federenko [24], Poussin [25], or Brandt [26] are considered to be among the most efficient techniques for the numerical solution of partial differential equations. The basic idea of the multigrid approach is to diminish not only high-but also low-frequency errors efficiently through restricting the problem to coarser grids. For unstructured grids, algebraic multigrid methods are very well suited because by definition no specified mesh structure is needed for the restriction. In the given work the authors implemented an algebraic multigrid solver based on the additive correction approach of Hutchinson [27] or Keller [28], and a preconditioned block-ILU is used as a smoother in the multigrid cycle (see Figure 1).

Details on an efficient implementation of a multigrid block solver can be found in [9]. Also note that the turbulence equations are solved with the same multigrid solver, although no interequation coupling is employed for the turbulence equations.

6. THE SOLUTION PROCEDURE

While the multigrid solver is used to solve the linearized system of equations, an outer loop is needed to resolve the nonlinearities in these equations, this iteration process can be outlined as follows [9].

1. Initialize values for volume flux $\dot{V}^{(n)}$, pressure $p^{(n)}$, and velocities $\mathbf{u}^{(n)}$.
2. Assemble source and matrix coefficients for momentum equations.
3. Evaluate the $\mathbf{D}$ tensor field from momentum equations' matrix coefficients.
4. Assemble source and matrix coefficients for continuity equation.
5. Solve simultaneously for pressure $p^{(n+1)}$ and velocities $\mathbf{u}^{(n+1)}$.
6. Solve the turbulence equations sequentially and adapt the kinematic turbulent viscosity ν_t .
7. Extract volume flux $\dot{V}^{(n+1)}$ from continuity equation.
8. Return to step 1 and loop until convergence.

7. RESULTS

The performance of the fully coupled solver is evaluated in four test problems, and comparisons to a SIMPLE-C solver by Mangani [15] are presented. The first

case is that of the NACA 0012 test problem. It is used to establish the accuracy of the solver by comparing its results with experimental data. The next test problem is a backward-facing step problem (Section 7.2) that is part of the test cases that are bundled with openFOAM. The third test case is an industrial-size test problem (Sections 7.3) used to demonstrate the computational performance and scalability of the coupled solver as compared to that of the segregated solver. Finally, an industrial test case, namely, a Pelton distributor (Section 7.4), is selected to evaluate the performance of the coupled solver with very large mesh domains.

In all the above test cases the root mean square (RMS) residuals for each field are evaluated as

$$\text{RMS}(\phi) = \frac{\sqrt{\frac{1}{N} \sum_{i=0}^N \left\{ \text{res}[\phi(i)] / a_C^{\phi} \right\}^2}}{\max(\phi, 0) - \min(\phi, 0)} \quad (22)$$

7.1. NACA 0012 Airfoil

The numerical studies were carried out based on measurement values of the flow field around a NACA 0012 airfoil section with a rounded (body of revolution) wing tip, based on the work of Dacles-Mariani et al. [29]. The detailed experimental results of these studies have been used to develop turbulence models more tuned to reflect the increased production of turbulent kinetic energy accompanied with rotating flows as for the case of wingtip vortices.

The wing has a 1.22m chord length and a semispan of 0.91 m. Complete geometry including the walls of the wind tunnel is given in Figure 2a. Distances are given in terms of chord length for generality. The mesh was built in agreement with the restrictions of the low-speed wind tunnel used during the measurements. To better investigate the development of the wingtip vortices, the domain was extended behind the trailing edge, as can be seen from Figure 2c. An O-grid was used around the airfoil and the complete computational domain is built up as a structured grid. A detail of the grid around the airfoil is given in Figure 2b. The complete mesh consists of approximately 1.5 million hexahedral cells. For the boundary conditions, a uniform Dirichlet field was applied to the inlet, where a turbulence length scale and a turbulence intensity were prescribed for the turbulence quantities. A von Neumann-type boundary condition was applied for the pressure at the inlet as well

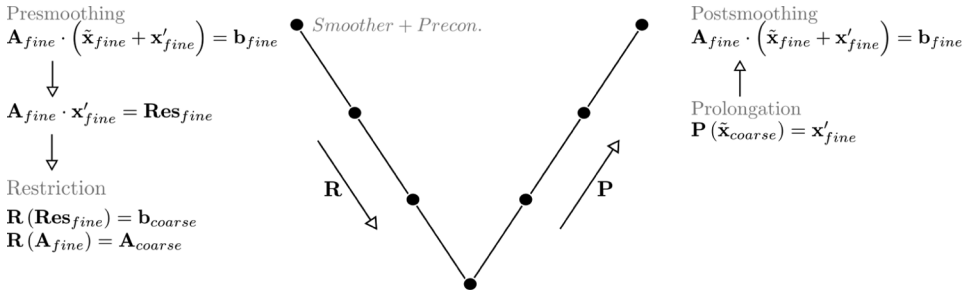


Figure 1. Multigrid cycle with restriction, prolongation and pre/post smoothing.

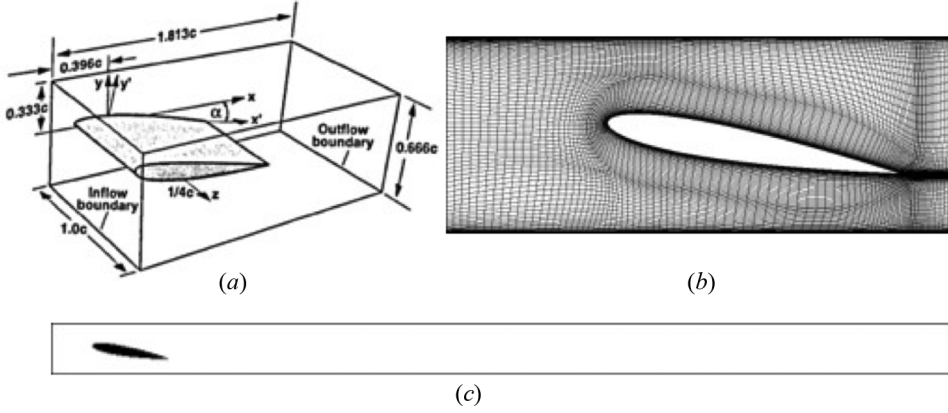


Figure 2. Experimental setup (a), computational grid (b), and computational domain (c).

as for velocities and turbulence quantities at the outlet. For the pressure field at the outlet a uniform field was applied. The density of the fluid is set to 1.225 kg/m^3 and kinematic viscosity is $1.46e-05 \text{ m}^2/\text{s}$. The performance of the coupled and segregated solvers in terms of computational time is shown in Table 1 and Figure 3. The runs were stopped at a specified RMS convergence threshold of $1e-5$.

Different numerical schemes and turbulence models have been used to investigate the influence on wingtip vortices. A converged solution is given in Figure 4, including the two evaluation planes. The first evaluation plane is placed right at the position of the trailing edge, the second on 24% of chord length downstream. Figure 5 shows the mentioned wingtip vortex, and the eddy-viscosity ratio is plotted on the evaluation planes. To compare simulation and experimental data, a cross-flow velocity was computed, defined as $U_{\text{crossflow}} = \sqrt{v^2 + w^2}/U_{\text{Inlet}}$.

7.2. Backward Facing Step

The backward-facing step test case was chosen to demonstrate the good scalability of the outlined solver with respect to the number of grid cells. The geometry of the test case can be seen in Figure 7. The test case has been carried out with three different grid sizes. For all grid sizes the same flow field has been obtained for both the block coupled and the segregated algorithm. The difference in performance and scalability is outlined in Table 2. The runs have been stopped at a specified RMS convergence threshold of $1e-5$.

From Table 2 it can clearly be seen that the block coupled algorithm outperforms the segregated algorithm in terms of convergence speed. More important, the backward-facing step test case proves the superiority of the coupled solver over the segregated

Table 1. Performance comparison of the coupled (C) and segregated (S) solvers

# CV	(C) time [s]	(C) time/CV [s]	(S) time [s]	(S) time/CV [s]	S/C
1,552 k	1,775.44	0.001144	34,459.20	0.022209	19.41

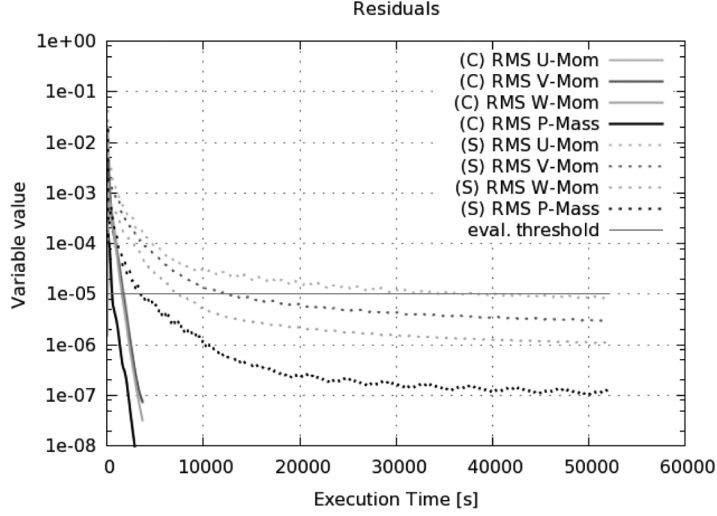


Figure 3. Convergence histories for NACA 0012 airfoil test case: segregated (S), dotted lines; coupled (C), full lines.

solver with respect to scalability with increasing grid size. In Figure 6 a scaling factor is plotted as a function of the grid size in order to show the superiority of the coupled solver over the segregated one. It can be seen that the coupled solver scales almost linearly, whereas the segregated solver's convergence behavior deteriorates a lot with increasing mesh sizes.

$$\text{Scale factor} = \frac{(\text{time}/\text{c.v.})_{n\text{Cells}}}{(\text{time}/\text{c.v.})_{\text{ref}}} \quad (23)$$

Figure 7 compares the velocity profiles at the indicated position of the coupled and the segregated approach. The small difference of the flow field is related to a slightly different boundary treatment.

With respect to convergence behavior, the coupled solver shows a smooth and steady convergence with a very good convergence rate. In Figure 8 it can be seen

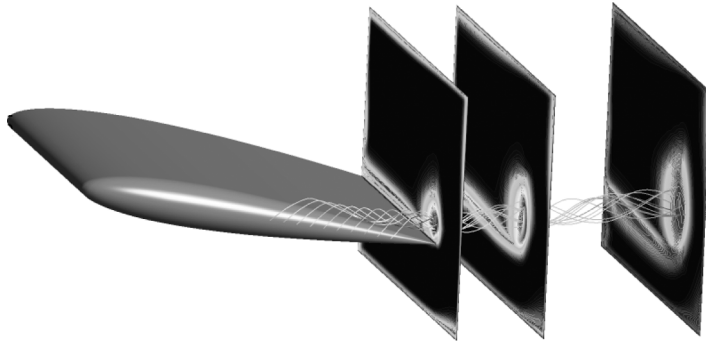


Figure 4. Wing-tip vortex.

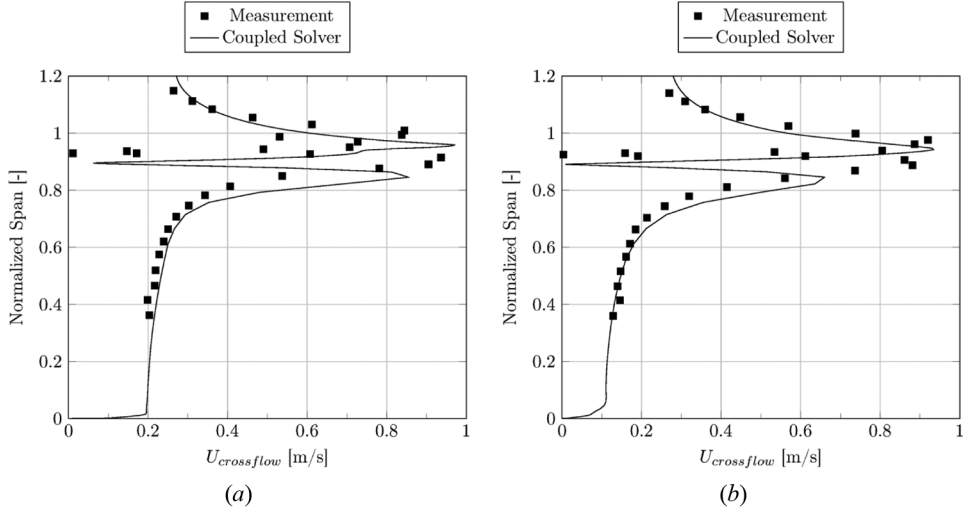


Figure 5. Comparison of $U_{crossflow}$ in spanwise direction pos. 0% span (a) and pos. 24% span (b).

Table 2. Backward-facing step: performance comparison of the coupled (C) and segregated (S) algorithms

# CV	(C) time [s]	(C) time/CV [s]	(S) time [s]	(S) time/CV [s]	S/C
12 k	4.3	0.000351	32.5	0.002653	7.6
48 k	24.1	0.000493	393.7	0.008051	16.3
195 k	139.9	0.000715	5,888.0	0.030102	42.1

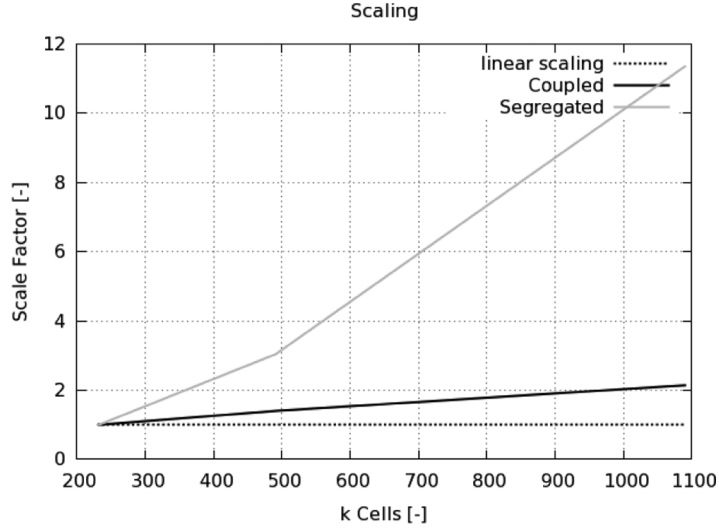


Figure 6. Backward-facing step: mesh-size scaling.

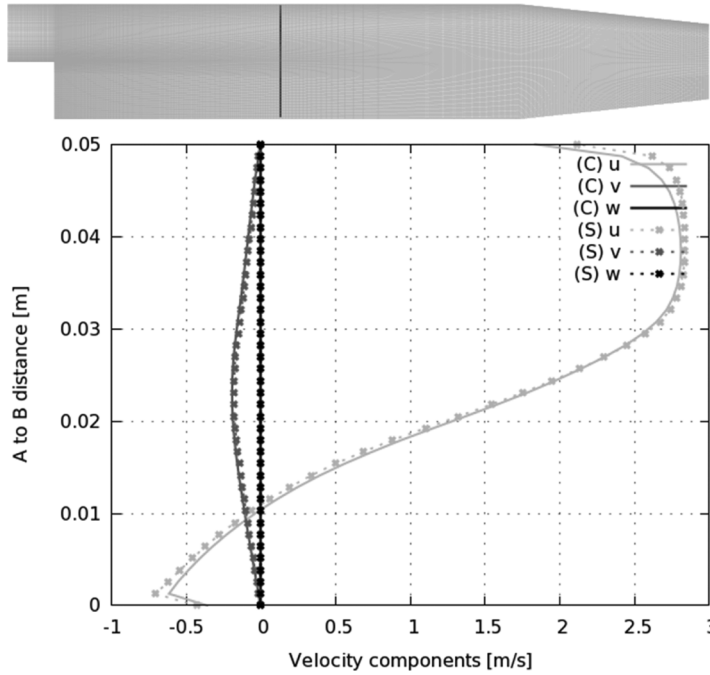


Figure 7. Velocity profiles at line indicated in black.

that the segregated solver, on the other hand, converges slowly, and its convergence shows fluctuating behavior. These fluctuations of the outer iterations, which arise from the weak variable coupling, are considered to be a sign of bad robustness.

7.3. Draft Tube

Draft tubes are very huge constructional elements that are placed behind hydraulic turbines in order to minimize efficiency losses at the turbine runners outlet by decreasing there the static pressure using a diffuser. Hence, the draft tube test case is a particularly difficult test case, because of its diffuser characteristic, which leads to flow detachment at its separation pier. The geometry, showed in Figure 9, has sharp edges and the mesh contains highly skewed cells at the butt of the pier. At the inlet a swirling flow is prescribed, meaning that not only a nonuniform axial velocity field is prescribed, but also a circumferential field that accounts for the preswirl generated by a thought turbine runner. For the pressure a Neumann boundary condition is set at the inlet. The turbulence quantities at the inlet are chosen to be uniformly constant for the sake of simplicity. At the outlet a Neumann condition is used for the velocity and turbulence quantities and a constant Dirichlet field is applied for the pressure. At the walls, blended wall functions are used to evaluate the shear stress accordingly.

The difference in performance and scalability is outlined in Table 3. The run times have been evaluated at a specified RMS convergence threshold of $1e-5$. The segregated method seems not to attain this convergence level for the finest grid,

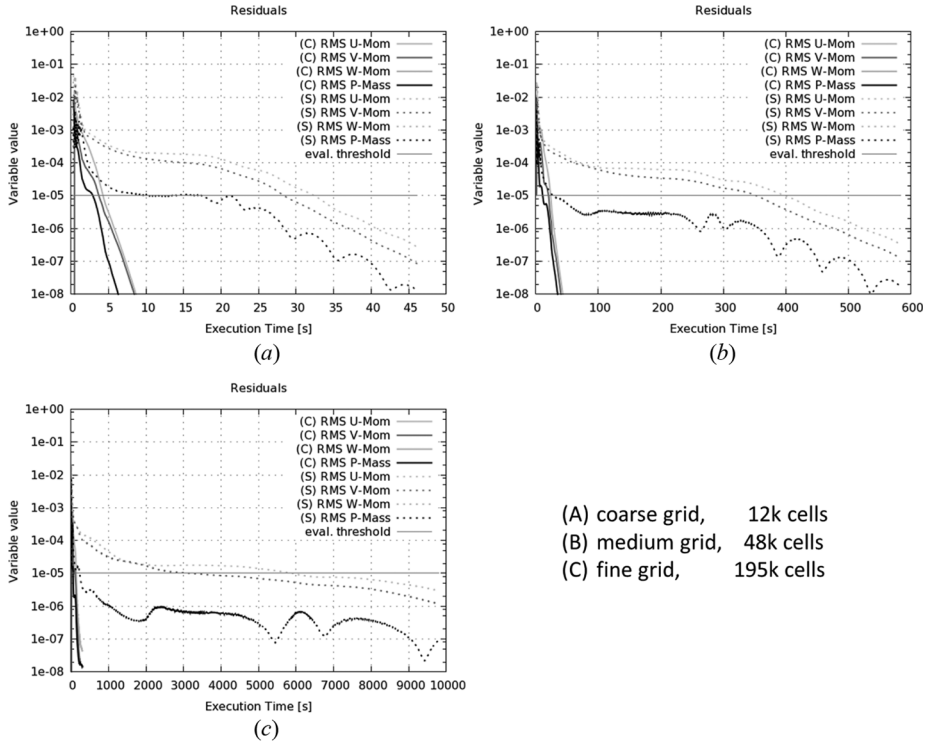


Figure 8. Convergence histories for backward-facing step test case: segregated (S), dotted lines; coupled (C), full lines.

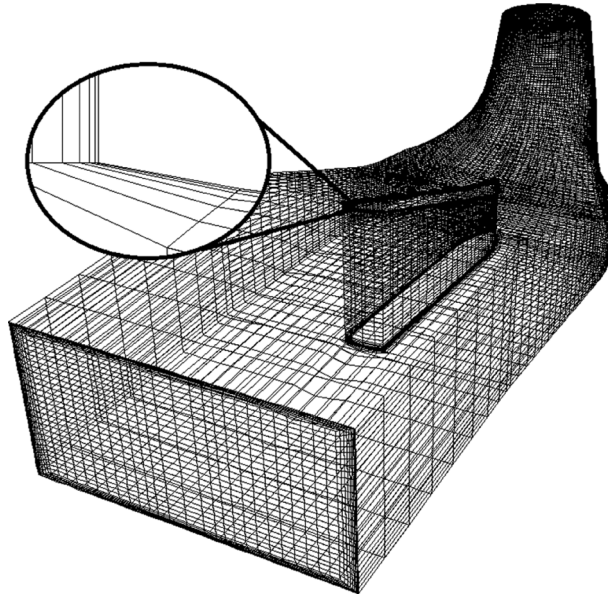


Figure 9. Computational grid of draft tube test case.

Table 3. Draft tube: performance comparison of the coupled (C) and segregated (S) algorithms

# CV	(C) time [s]	(C) time/CV [s]	(S) time [s]	(S) time/CV [s]	S/C
232 k	535.9	0.002304	1,342.2	0.005771	2.50
491 k	988.6	0.002013	3,656.7	0.007447	3.70
1,090 k	1,997.0	0.001831	x	x	x

or only very slowly. Table 3 indicates that the coupled solver converges much faster than its segregated counterpart and that the segregated solver is being outperformed in terms of scaling.

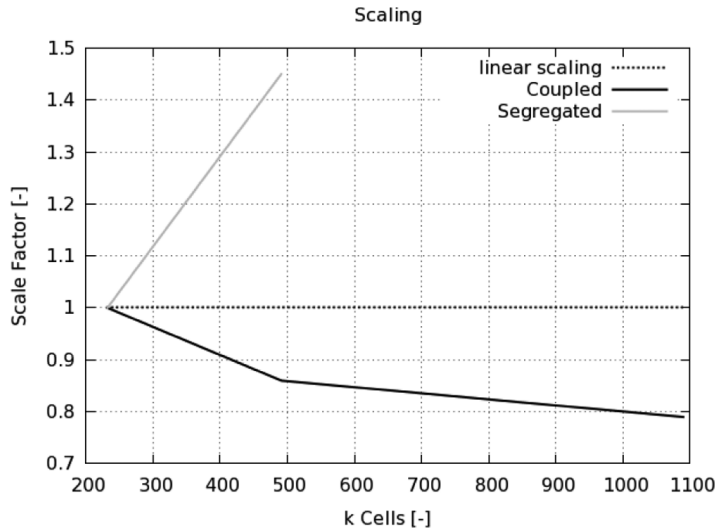
Astonishingly, the mesh size scalability is even sublinear for the draft tube test case (see Figure 10). The reason for this overperformance is due to extremely skewed cells in the butt region of the pier (see Figure 9). Since the quality of the mesh is increased with an increasing number of cells, the convergence rate also seems to increase for finer meshes.

Figure 11 illustrates the good performance of the coupled approach compared to the segregated approach. For the fine-mesh configuration, the segregated solver's convergence rate almost stalls.

The velocity contour plot of a slice through the draft tube shows very similar flow patterns (see Figure 12). The differences again arise from the alternate boundary treatment, which leads to slightly different detachment positions.

7.4. Pelton Distributor

The function of a Pelton turbine distributor is to distribute water coming from a high-altitude basin to a couple of injector nozzles that will continuously apply

**Figure 10.** Draft tube mesh-size scaling.

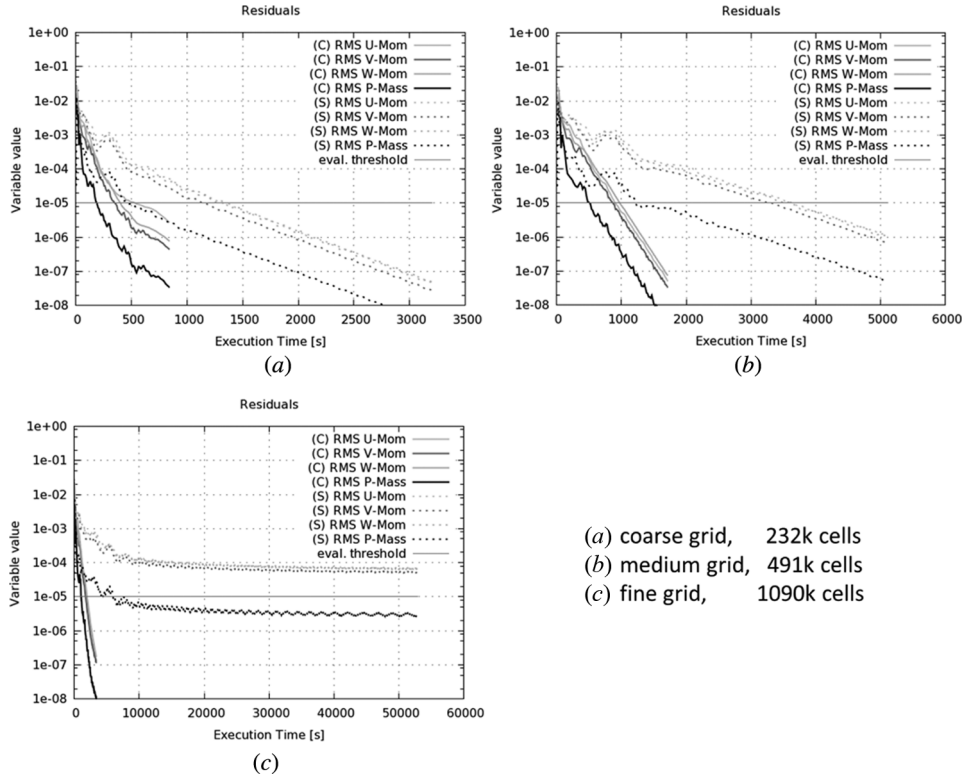


Figure 11. Convergence histories for draft tube test case: segregated (S), dotted lines; coupled (C), full lines.

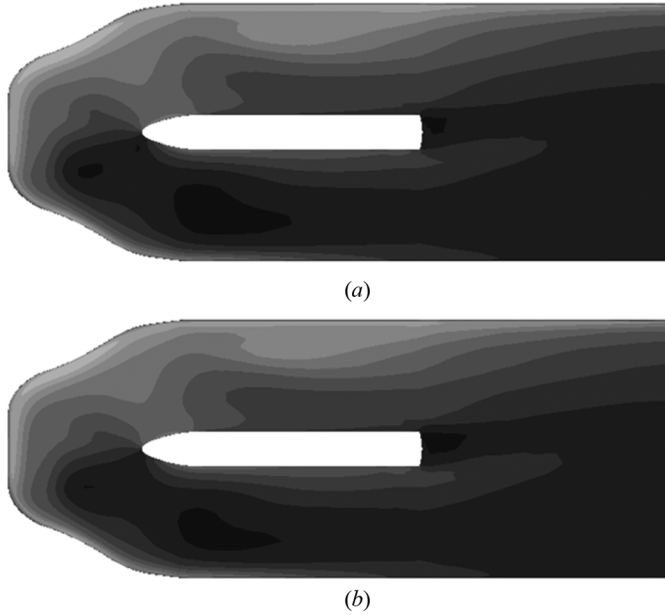


Figure 12. Velocity contour plot of draft tube test case: (S) left; (C) right.

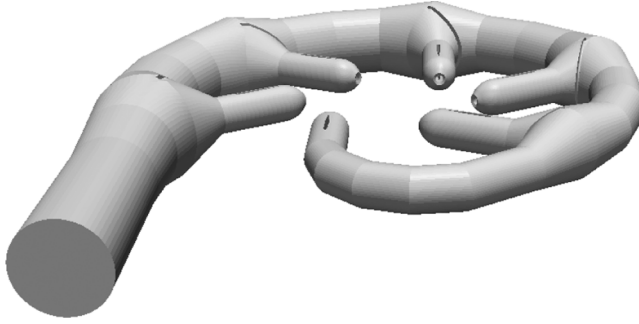


Figure 13. Pelton distributor.

jets of water to a Pelton turbine runner. The flow in Pelton turbine distributors is very demanding for CFD applications, because plenty of detachment zones exist, when can lead to difficulties in convergence for steady-state flow. That is, there are flow detachment areas at the beginning of the distributor's injectors, and furthermore, the grid contains highly skewed cells at the bifurcations (see Figure 13). The Pelton distributor test case is evaluated with two different mesh sizes in order to

Table 4. Pelton distributor: Mesh-size scaling of the coupled (C) algorithm

# CV	(C) time [s]	(C) time/CV [s]
2,726 k	2,932	0.001075
6,253 k	6,068	0.000970

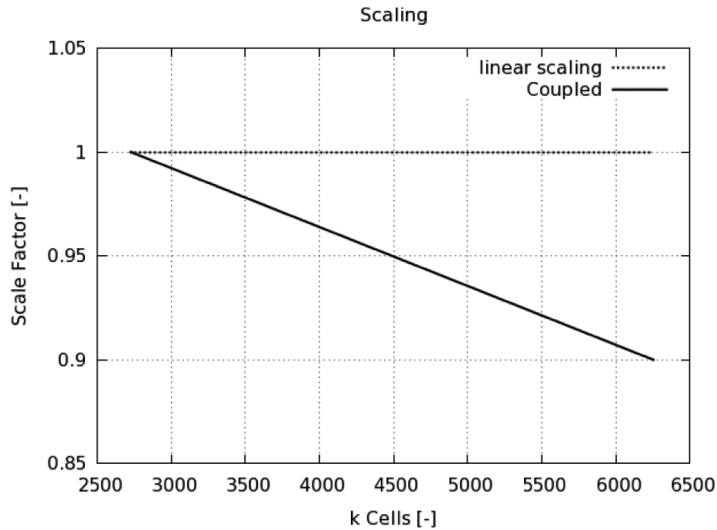


Figure 14. Pelton distributor mesh-size caling.

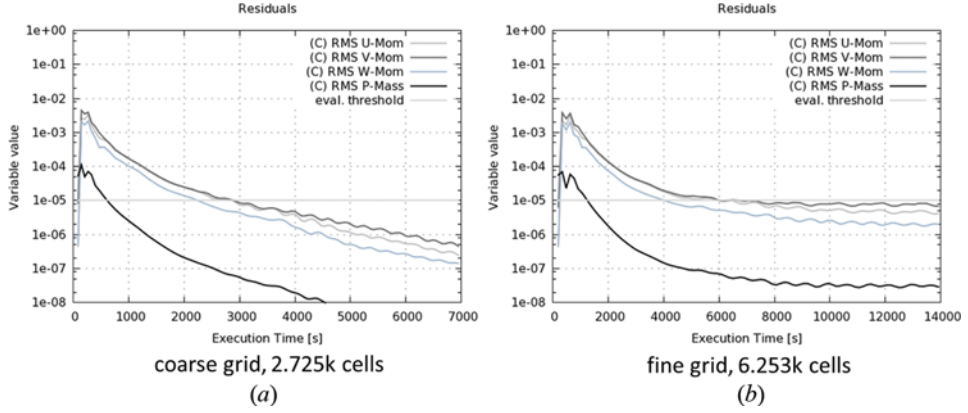


Figure 15. Convergence history for Pelton distributor test case (2.726 k cells).

investigate the mesh size scaling properties of the coupled solver. The mesh sizes are computationally quite demanding (with 2.7 and 6.3 million cells, respectively).

The difference in performance compared to the benchmark solver is outlined in Table 4. The run time has been evaluated at a specified RMS convergence threshold of $1e-5$.

Table 4 and Figure 14 demonstrate again that the coupled solver scales excellently with increasing numbers of cells, even for very big meshes. As was the case for the draft tube (Section 7.3), the coupled solver even over-performs, since the time per control volume for the bigger-mesh case is even smaller than for the smaller-mesh case. However, for the fine-grid case, the coupled algorithm experiences a slowdown in convergence rate, after passing the $1e-5$ threshold of the RMS residuals (see Figure 15).

7.5. Test Case Summary

In order to summarize the obtained results, Table 5 outlines once again the coupled solver's good performance and mesh size scalability.

Table 5. Test case summary of mesh-size scaling for coupled (C) and segregated (S) algorithms

Test case	Grid incr. factor	(C) time/CV [s]	(S) time/CV [s]	S/C
NACA 0012	1	0.001144	0.022209	19.41
Backward-facing step	1	0.000351	0.002653	7.6
	4	0.000493	0.008051	16.3
	16	0.000715	0.030102	42.1
Draft tube	1	0.002304	0.005771	2.50
	2.12	0.002013	0.007447	3.70
	4.70	0.001831	x	x
Pelton distr.	1	0.001075	x	x
	2.29	0.000970	x	x

8. CONCLUSION

A pressure-based, fully implicit coupled solver was developed and implemented within the OpenFOAM framework. The coupled solver demonstrated substantially improved performance in terms of CPU and iterations to convergence compared to segregated algorithms. Additionally, its smoother RMS residuals convergence history is a good indicator of its robustness. The fully implicit pressure-velocity coupling needs to iterate to resolve the nonlinear part of the equations; this is different from the segregated algorithms; which need to resolve both the nonlinearities in the equations and the pressure-velocity coupling that is treated explicitly.

Just as important, it was shown that the coupled solver has good scalability with increasing mesh sizes in terms of computational time to convergence. This is a clear advantage over segregated algorithms, especially when dealing with the large meshes that arise in industrial-size cases. The qualitative results have been shown to be almost identical for both approaches, with minor differences, mainly due to a different treatment of the boundary conditions.

FUNDING

The financial aid granted by the Austrian research fund FFG, for projects 828688-HydroSim and M 1543-N30, is greatly acknowledged.

REFERENCES

1. S. Patankar and D. B. Spalding, A Calculation Procedure for Heat, Mass and Momentum Transfer in Three-dimensional Parabolic Flows, *Int. J. Heat Mass Transfer*, 1972.
2. J. V. Doormaal and G. Raithby, An Evaluation of the Segregated Approach for Predicting Incompressible Fluid Flows, *National Heat Transfer Conference*, ASME Paper 85-HT-9, Denver, CO, 1986.
3. M. Darwish and F. Moukalled, A Review of Boundary Conditions and Their Implementations in CFD Codes, *Int. J. Numer. Meth. in Fluids*, 2000.
4. F. Moukalled and M. Darwish, A Unified Formulation of the Segregated Class of Algorithms for Fluid Flow at All Speeds, *Numer. Heat Transfer, B*, 2000.
5. M. Darwish, F. Moukalled, and B. Sekar, A Unified Formulation for the Segregated Class of Algorithms for Multi-Fluid Algorithm at All Speeds, *Numer. Heat Transfer B*, 2001.
6. D. Singh, B. Premachandran, and S. Kohli, Numerical Simulation of the Jet Impingement Cooling of a Circular Cylinder, *Numer. Heat Transfer A*, 2013.
7. M. Darwish, D. Asmar, and F. Moukalled, A Comparative Assessment within a Multigrid Environment of Segregated Pressure Based Algorithms for Fluid Flow at all Speeds, *Numer. Heat Transfer part B*, 2004.
8. J. V. Doormaal and G. Raithby, Enhancements of the Simple Method for Predicting Incompressible Fluid Flows, *Numer. Heat Transfer*, 1984.
9. M. Darwish, I. Sraj, and F. Moukalled, A Coupled Finite Volume Solver for the Solution of Incompressible Flows On Unstructured Grids, *J. Comput. Phys.*, 2008.
10. H. Weller, G. Tabor, H. Jasak, and C. Fureby, A Tensorial Approach to Computational Continuum Mechanics Using Object-Oriented Techniques, *Comput. in Phys.*, 1998.
11. C. Vradis and K. J. Hammad, Strongly Coupled Block-Implicit Solution Technique for Non-Newtonian Convective Heat Transfer Problems, *Numer. Heat Transfer B*, vol. 33, pp. 79–97, 1998.

12. M. J. S. de Lemos, Flow and Heat Transfer in Rectangular Enclosures Using a New Block-Implicit Numerical Method, *Numer. Heat Transfer B*, vol. 37, pp. 489–508, 2000.
13. Y. G. Laia, An Unstructured Grid Method for a Pressure-Based Flow and Heat Transfer Solver, *Numer. Heat Transfer B*, vol. 32, pp. 267–281, 1997.
14. P. L. Woodfield, K. Suzuki, and K. Nakabe, Performance of a Three-Dimensional, Pressure-Based, Unstructured Finite-Volume Method for Low-Reynolds-Number Incompressible Flow and Wall Heat Transfer Rate Prediction, *Numer. Heat Transfer B*, vol. 43, pp. 403–423, 2003.
15. E. Casartelli, L. Mangani, and S. Hug, Numerical Comparison between Model and Prototype Flow in a Pump-Turbine Distributor, *International Conference and Exhibition Innovative Approaches to Global Challenges* Bilbao, Spain, 2012.
16. F. Menter, Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications, *AIAA J.*, 1994.
17. F. Menter, M. Kuntz, and R. Langtry, Ten Years of Industrial Experience with the SST Turbulence Model, *Turbulence, Heat Mass Transfer*, 2003.
18. M. Benzi, G. Golub, and J. Liesen, Numerical Solution of Saddle Point Problems, *Acta Numer.*, pp. 1–137, 2005.
19. S. Patankar, *Numerical Heat Transfer and Fluid Flow*, Hemisphere, Washington, DC, 1980.
20. C. Rhie and W. Chow, A Numerical Study of Turbulent Flow Past an Isolated Airfoil with Trailing Edge Separation, *AIAA J.*, 1983.
21. S. Muzaferija, Adaptive Finite Volume Method for Flow Predictions Using Unstructured Meshes and Multigrid Approach, Ph.D. thesis, University of London, London, UK, 1994.
22. J. Ferziger and M. Peric, *Computational Methods for Fluid Dynamics*, Springer–Verlag, Berlin, 1994.
23. M. Darwish and F. Moukalled, TVD Schemes for Unstructured Grids, *Int. J. Heat and Mass Transfer*, 2003.
24. R. Federenko, A Relaxation Method for Solving Elliptic Difference Equations, *Zh. Vychisl. Mat. Mat. Fiz.*, pp. 922–927.
25. F. Poussin, An Accelerated Relaxation Algorithm for Iterative Solution of Elliptic Methods, *SIAM J. Numer. Anal.*, 1968.
26. A. Brandt, Multi-Level Adaptive Solutions to Boundary Value Problems, *Math. Comput.*, 1977.
27. B. Hutchinson and G. D. Raithby, A Multigrid Method Based on the Additive Correction Strategy, *Numer. Heat Transfer*, 1986.
28. S. Keller, Additive Correction Multigrid Method Applied to Diffusion Problems with Unstructured Grids, *Proceedings of the 10th Brazilian Congress of Thermal Sciences and Engineering*, Rio de Janeiro, Brazil, 2004.
29. J. Dacles-Mariani, G. Zilliac, S. Chow, and P. Bradshaw, Numerical/Experimental Study of a Wingtip Vortex in the Near Field, *AIAA J.*, vol. 33, pp. 1561–1568, 1995.