

DEVELOPING FULLY-COUPLED DENSITY-BASED SOLVER FOR TURBULENT COMPRESSIBLE FLUID FLOWS IN OPENFOAM

T. Levý¹, J. Vimmr¹

¹ Department of Mechanics, Faculty of Applied Sciences, University of West Bohemia, Technická 8, 301 00 Pilsen, Czech Republic

Abstract

This paper is devoted to numerical solution of turbulent compressible fluid flow which is governed by the system of the Favre-averaged Navier-Stokes equations closed by the two-equation $k-\omega$ SST turbulence model. The contribution lies in the implementation of finite volume method in fully-coupled density-based manner with AUSM scheme for calculation of numerical convective fluxes using linear reconstruction along with Barth-Jespersen limiter and explicit time integration in two different software environments. The first one is the in-house developed numerical code in MATLAB while in the second case the system of equations is solved in the framework of open-source package OpenFOAM where we have implemented the same numerical techniques. Comparison of both of these above mentioned approaches together with standard OpenFOAM solver is performed on the well-known turbulent flat plate test case.

Keywords: finite volume method, coupled density-based solver, turbulent compressible flow, OpenFOAM.

1 Introduction

Open-source software OpenFOAM is nowadays widely used in the field of computational fluid dynamics. It is a library based on object-oriented C++ programming language for solving equations arising from the modeling of fluid flow allowing the user to utilize its high level programming syntax for manipulation with scalar, vector or tensor fields related to the cells of generally three-dimensional polyhedral meshes, [1]. OpenFOAM extensively uses so-called pressure-based velocity-pressure correction algorithms proposed originally to solve the Navier-Stokes equations of incompressible flow, i.e. PISO (Pressure-Implicit with Splitting of Operators), [2], and SIMPLE (Semi-Implicit Method for Pressure Linked Equations), [3]. On the other hand, for high speed compressible flow problems the so-called density-based solvers are more suitable than the mentioned pressure-based methods. So far, the only density-based solver in the official OpenFOAM distribution is `rhoCentralFoam` based on the central-upwind scheme of Kurganov and Tadmor, [4].

In `foam-extend` package which is a fork of OpenFOAM there is another density-based solver for turbulent compressible flow called `dbnsTurbFoam` that is based on Godunov type schemes, e.g. Rusanov, Roe or HLLC numeric fluxes, combined with an explicit time-marching method. Similar approaches are used also in the `aeroFoam` solver developed by Romanelli et al. [5]. To remove the limit imposed on the maximal size of a timestep and allow faster convergence to the steady state an implicit method for the temporal discretization with the use of Generalised Minimum Residual (GMRES) method and Lower-Upper Symmetric Gauss-Seidel (LU-SGS) preconditioning technique have been implemented in the `hisa` solver, [6, 7]. All of the above mentioned density-based solvers are coupled in the sense of the mean flow equations. The equations of specific turbulence models are kept segregated since it is then possible to use the already implemented turbulent models in OpenFOAM or `foam-extend` distribution.

This contribution is motivated by the deficiency of density-based solvers in OpenFOAM package. Instead of solving the mean flow equations, i.e. the Favre-averaged Navier-Stokes equations, and the equations of turbulence model separately, the effort is put on solving the whole system of equations at once. Implementation of this fully-coupled approach using the $k-\omega$ SST turbulence model was published in diploma thesis of the author, [8], resulting in the in-house developed solver programmed in the MATLAB environment. The aim of the paper is therefore to depict such implementation in the framework of OpenFOAM which is primarily designed for pressure-based solvers. This improvement thus allows a wider range of applications when using OpenFOAM utilities,

higher computational efficiency since C++ is generally faster than MATLAB and the possibility of further development.

In order to verify implemented density-based solver in OpenFOAM a test case of turbulent compressible fluid flow over a flat plate is considered in this work. Results are verified by means of the solution provided by the NASA Turbulence Modeling Resource webpage, [9]. Additionally, results of OpenFOAM built-in pressure-based solver for compressible flow called rhoPimpleFoam are also presented.

2 Governing equations

The mathematical model of turbulent compressible fluid flow is represented by the nonlinear system of the Favre-averaged Navier-Stokes equations expressing the fundamental laws of the conservation of mass, momentum and energy of the mean flow, [10]. The system of equations can be written in conservative form as

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) &= 0 \\ \frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) + \nabla p &= \nabla \cdot (\boldsymbol{\sigma} + \boldsymbol{\tau}), \\ \frac{\partial(\rho E)}{\partial t} + \nabla \cdot [(\rho E + p)\mathbf{u}] &= \nabla \cdot [(\boldsymbol{\sigma} + \boldsymbol{\tau}) \cdot \mathbf{u}] + \nabla \cdot [(\lambda + \lambda_T) \nabla T], \end{aligned} \quad (1)$$

where ρ is the density, $\mathbf{u}$ is the vector of velocity, p is the pressure, E is the specific total energy, T is the temperature, $\boldsymbol{\sigma}$ and $\boldsymbol{\tau}$ are the viscous stress tensor and turbulent stress tensor (also termed the Reynolds stress tensor) respectively and similarly λ is the thermal conductivity and λ_T is its turbulent counterpart. Here $\mathbf{u} \otimes \mathbf{u}$ is the tensor with components $u_i u_j$, $i, j = 1, \dots, N$, where N represents the space dimension of a specific problem. The pressure is coupled with the system (1) by an equation of state of the ideal gas in the form

$$p = (\kappa - 1) \left(\rho E - \frac{1}{2} \|\mathbf{u}\|^2 \right), \quad (2)$$

where $\kappa = 1.4$ is the Poisson adiabatic constant.

For thermal conductivities the following relations holds

$$\lambda = c_p \frac{\mu}{Pr}, \quad \lambda_T = c_p \frac{\mu_T}{Pr_T}, \quad (3)$$

where c_p is the specific heat at constant pressure and Prandtl number Pr and turbulent Prandtl number Pr_T are set to $Pr = 0.72$ and $Pr_T = 0.9$ respectively. The fluid is assumed to be Newtonian and the dynamic viscosity μ is therefore computed using the Sutherland's law, [11]. In such case the dynamic viscosity is function of temperature only. Turbulent viscosity μ_T which is also component of the Reynolds stress tensor represents the effect of the turbulent fluctuations and need to be modeled by a turbulence model in order to close the system of equations.

In this work we will use Menter's two equation $k-\omega$ SST turbulence model, [12], in the form

$$\begin{aligned} \frac{\partial(\rho k)}{\partial t} + \nabla \cdot (\rho k \mathbf{u}) &= \nabla \cdot [(\mu + \sigma_k \mu_T) \nabla k] + \mathcal{P} - \beta^* \rho \omega k, \\ \frac{\partial(\rho \omega)}{\partial t} + \nabla \cdot (\rho \omega \mathbf{u}) &= \nabla \cdot [(\mu + \sigma_\omega \mu_T) \nabla \omega] + \frac{\gamma \rho}{\mu_T} \mathcal{P} - \beta \rho \omega^2 + 2(1 - F_1) \frac{\rho \sigma_{\omega 2}}{\omega} \nabla k \cdot \nabla \omega, \end{aligned} \quad (4)$$

where k is the turbulent kinetic energy and ω is the specific dissipation rate, σ_k , σ_ω , $\sigma_{\omega 2}$, β , β^* and γ are model coefficients and F_1 is model function. The production term $\mathcal{P} = \boldsymbol{\tau} : \nabla \mathbf{u}$ is approximated here by $\mu_T \|\boldsymbol{\Omega}\|^2$, where $\boldsymbol{\Omega}$ is the vorticity tensor, [13]. Finally, the turbulent viscosity is obtained by the relation

$$\mu_T = \frac{\rho a_1 k}{\max(a_1 \omega, \|\boldsymbol{\Omega}\| F_2)}, \quad (5)$$

where a_1 is model coefficient and F_2 is model function.

3 Numerical solution

The system of equations (1) and (4) can be rewritten in the integral form on a computational domain $\Omega \subset \mathbb{R}^N$ with boundary $\partial\Omega$ as

$$\frac{\partial}{\partial t} \int_{\Omega} \mathbf{W} \, d\Omega + \oint_{\partial\Omega} (\mathbf{F}^C - \mathbf{F}^V) \, dS = \int_{\Omega} \mathbf{Q} \, d\Omega, \quad (6)$$

where $\mathbf{W} = [\rho, \rho\mathbf{u}, \rho E, \rho k, \rho\omega]$ is the vector of conservative variables and $\mathbf{F}^C$, $\mathbf{F}^V$ and $\mathbf{Q}$ are the vectors of the convective fluxes, the viscous fluxes and the source terms respectively. For spatial discretization of the system (6) the cell centered finite volume method (FVM), [14], is used. The FVM is based on the piecewise constant approximation of a solution by taking

$$\mathbf{W}_i(t) = \frac{1}{|\Omega_i|} \int_{\Omega_i} \mathbf{W}(\mathbf{x}, t) \, d\Omega \quad (7)$$

on each mesh cell Ω_i . Here $|\Omega_i|$ denotes the magnitude of the i -th mesh cell. Hence (6) can be approximated for particular cell Ω_i as

$$\frac{d}{dt} \mathbf{W}_i = -\frac{1}{|\Omega_i|} \sum_{j=1}^{S(i)} (\mathbf{F}_{ij}^C - \mathbf{F}_{ij}^V) |\Gamma_{ij}| + \mathbf{Q}_i \equiv \mathbf{R}_i(\mathbf{W}), \quad (8)$$

where $S(i)$ is the set of cells which are adjacent to mesh cell Ω_i , $|\Gamma_{ij}|$ is the magnitude of face shared by cell Ω_i and Ω_j and $\mathbf{F}_{ij}^C$, $\mathbf{F}_{ij}^V$ are the numerical fluxes approximating the convective and viscous fluxes on j -th face. The source term is assumed to be constant inside the mesh cell.

The discretization of convective fluxes is made by means of approximate Riemann solver. AUSM scheme of Liou, [15], is employed in this work together with piecewise linear reconstruction in order to improve accuracy of the scheme. Reconstructed value of $\mathbf{W}_i$ on the j -th face of the cell limited by the Barth-Jespersen limiter to prevent spurious oscillations in solution is given as, [16],

$$\mathbf{W}_{ij} = \mathbf{W}_i + \Phi_i \nabla \mathbf{W}_i \mathbf{r}_j, \quad (9)$$

where $\mathbf{r}_j$ is the vector connecting the center of the cell Ω_i and center of the face between cell Ω_i and Ω_j and $\nabla \mathbf{W}_i$ indicates gradient of the components of the vector $\mathbf{W}_i$. The limiter itself is defined as

$$\Phi_i = \min_{j \in S(i)} \Phi_{ij}, \quad (10)$$

with

$$\Phi_{ij} = \begin{cases} \min \left(1, \frac{\mathbf{W}_{\max} - \mathbf{W}_i}{\nabla \mathbf{W}_i \mathbf{r}_j} \right), & \text{if } \nabla \mathbf{W}_i \mathbf{r}_j > 0, \\ \min \left(1, \frac{\mathbf{W}_{\min} - \mathbf{W}_i}{\nabla \mathbf{W}_i \mathbf{r}_j} \right), & \text{if } \nabla \mathbf{W}_i \mathbf{r}_j < 0, \\ 1, & \text{if } \nabla \mathbf{W}_i \mathbf{r}_j = 0. \end{cases} \quad (11)$$

Here $\mathbf{W}_{\max} = \max(\mathbf{W}_i, \max_{j \in S(i)} \mathbf{W}_j)$ and $\mathbf{W}_{\min} = \min(\mathbf{W}_i, \min_{j \in S(i)} \mathbf{W}_j)$.

Viscous fluxes are discretized with central scheme. In the case of MATLAB implementation the values of the conservative variables on faces are determined using the dual cells but in OpenFOAM implementation are these obtained only with the values in adjacent cell centers, [17]. Hence the MATLAB code uses a larger stencil of mesh cells. Detailed information about the discretization is covered in the diploma thesis of the author, [8].

For the temporal discretization of the system of ordinary differential equations (8) the explicit two-stage Runge-Kutta method is used in the form

$$\begin{aligned} \mathbf{W}_i^{(1)} &= \mathbf{W}_i^n, \\ \mathbf{W}_i^{(2)} &= \mathbf{W}_i^{(1)} + \alpha_1 \Delta t \mathbf{R}_i(\mathbf{W}^{(1)}), \\ \mathbf{W}_i^{n+1} &= \mathbf{W}_i^{(1)} + \alpha_2 \Delta t \mathbf{R}_i(\mathbf{W}^{(2)}), \end{aligned} \quad (12)$$

where $\alpha_1 = 1/2$ and $\alpha_2 = 1$.

4 Implementation in OpenFOAM

The basis of the presented solver is built on the `dbnsFoam` solver of the `foam-extend` package for solving inviscid Euler equations. This solver also uses the Godunov method, see e.g. [18], for discretization of convective fluxes and the explicit multi-stage Runge-Kutta method similar to (12) for time integration of the semidiscrete system of ODEs which makes it an ideal candidate for the extension to our system of equations (8). In `foam-extend`, there is also `dbnsTurbFoam` solver for the turbulent compressible fluid flow but it uses the built-in turbulence models which are discretized in both space and time in a different way and hence it is not possible to use it in our case of fully-coupled equations.

In order to solve (8) a `dbnsFoam` solver was modified and compiled under the current version of OpenFOAM together with implementing the AUSM scheme for convective fluxes. Then it was extended by viscous fluxes and source terms. Like in the case of `foam-extend`, the OpenFOAM built-in turbulence models could not be used because of the different method of its solution. For this purpose it was necessary to create a new C++ class to deal with the closure coefficients and auxiliary relations of the k - ω SST turbulence model. Hence the use of other turbulence models is not straightforward and its proper implementation is needed instead of selecting the already implemented turbulence model from the OpenFOAM package database.

5 Results

The two-dimensional subsonic turbulent flow over a flat plate with zero pressure gradient which is part of the NASA Turbulence Modeling Resource (NASA TMR) database, [9], has been chosen as a benchmark. The flow enters the computational domain $\Omega \subset \mathbb{R}^2$ at boundary $\partial\Omega_I$, see Fig. 1, in distance 0.33 [m] in front of the leading edge of the flat plate with total pressure $p_{t,\infty} = 104190.5$ [Pa], total temperature $T_{t,\infty} = 302.4$ [K] and turbulent intensity $Tu = 0.08\%$ at zero angle of attack. The flat plate itself is 2 [m] long and is assumed to be adiabatic solid wall $\partial\Omega_W$ with no-slip condition $\mathbf{u} = \mathbf{0}$ and $\mu_T = 0$, $k = 0$ and $\omega = 60\mu/\rho\beta_1 y_1^2$ for the turbulent quantities, where β_1 is a model constant and y_1 is the distance of the first cell center from the wall. At the outlet $\partial\Omega_O$ a value of pressure $p = 101325$ [Pa] is prescribed and other quantities are extrapolated. At the other boundaries $\partial\Omega_S$ of computational domain Ω a symmetry boundary condition is prescribed. The flow is further characterized by a Reynolds number of $Re = 5 \cdot 10^6$ based on unit length.

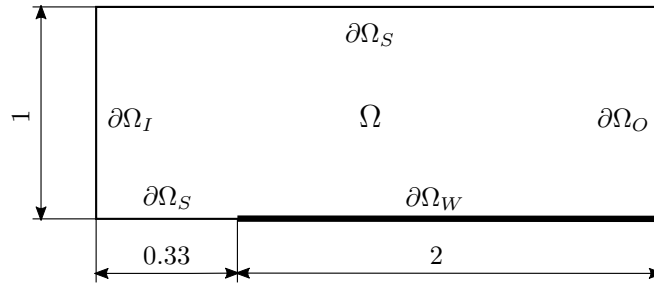
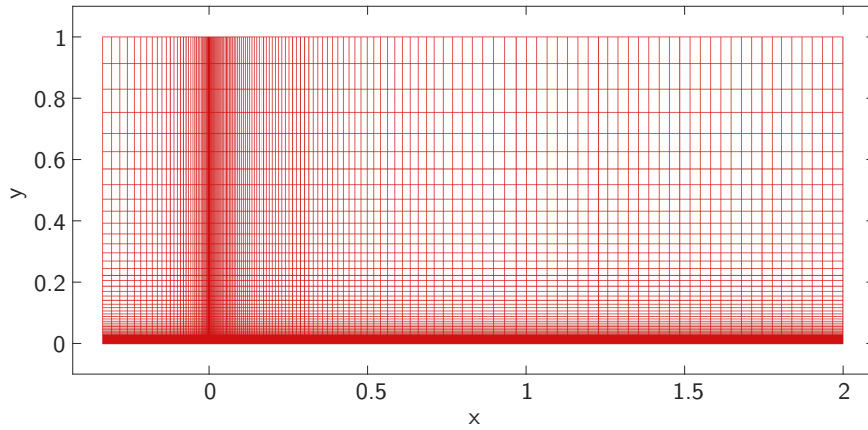


Figure 1: Computational domain $\Omega \subset \mathbb{R}^2$ for the flat plate with the boundary $\partial\Omega = \partial\Omega_I \cup \partial\Omega_O \cup \partial\Omega_S \cup \partial\Omega_W$

The solution was obtained using a structured quadrilateral mesh with 140×100 cells, Fig. 2, which is stretched in the distance normal to the wall and clustered near the flat plate leading edge. The first cell center near the wall is in distance $y_1 = 5.2 \cdot 10^{-6}$ [m] giving an approximate average of nondimensional wall distance $y^+ \approx 0.9$ over the plate.

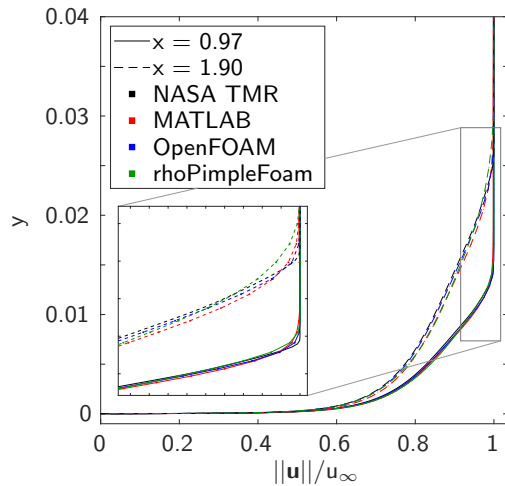
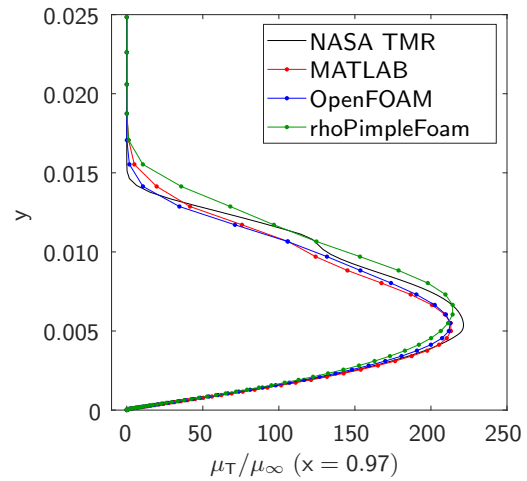
The results obtained with the in-house developed code in MATLAB and implementation in OpenFOAM are compared with the solution from NASA TMR for which a greater mesh resolution with 544×384 cells has been used. The test case was also computed with OpenFOAM built-in solver for compressible flow called `rhoPimpleFoam` which uses the PIMPLE algorithm and standard

Figure 2: Structured mesh with 140×100 quadrilaterals

implementation of turbulence models. The PIMPLE algorithm is a combination of PISO, [2], and SIMPLE, [3], algorithms usually used for solving of incompressible fluid flows. For compressible flows therefore this solver can be somewhat diffusive but for considered case where the freestream Mach number is $Ma_\infty \approx 0.2$ this should not cause a problem.

Fig. 3 shows a comparison of the velocity profiles scaled by the freestream velocity for each solver in distance 0.97 [m] and 1.9 [m] from the flat plate leading edge. In Fig. 4 - 6 distribution of turbulent quantities near the wall is shown.

One can see that in the case of the turbulent viscosity nondimensionalized by the freestream dynamic viscosity the PIMPLE solver slightly overestimates the position of maximal μ_T . A similar situation occurs in the distribution of k and ω where rapid changes are present at the boundary layer edge. Also, both solutions obtained in the OpenFOAM framework show slight deviation from NASA TMR results at the wall. Nevertheless, all solvers give similar results for velocities. For proper analysis a better mesh resolution need to be used.

Figure 3: Velocity profiles in locations $x = 0.97$ and $x = 1.9$ Figure 4: Profile of nondimensional turbulent viscosity at $x = 0.97$

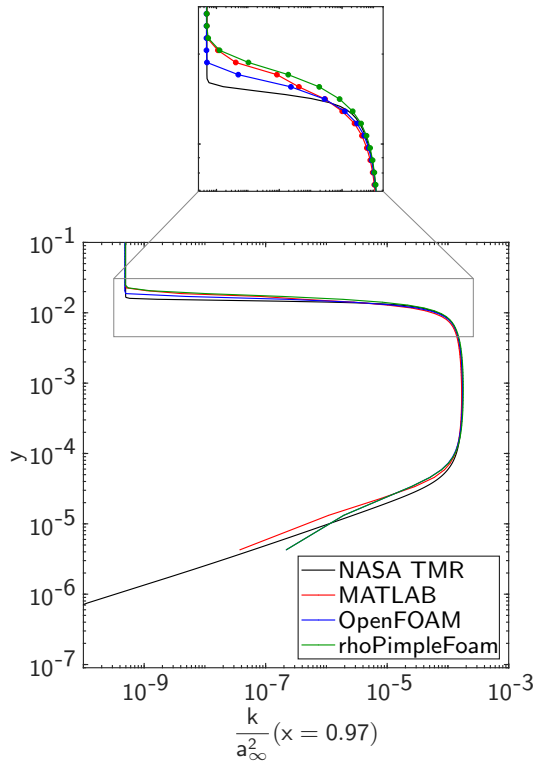


Figure 5: Profile of nondimensional k at $x = 0.97$

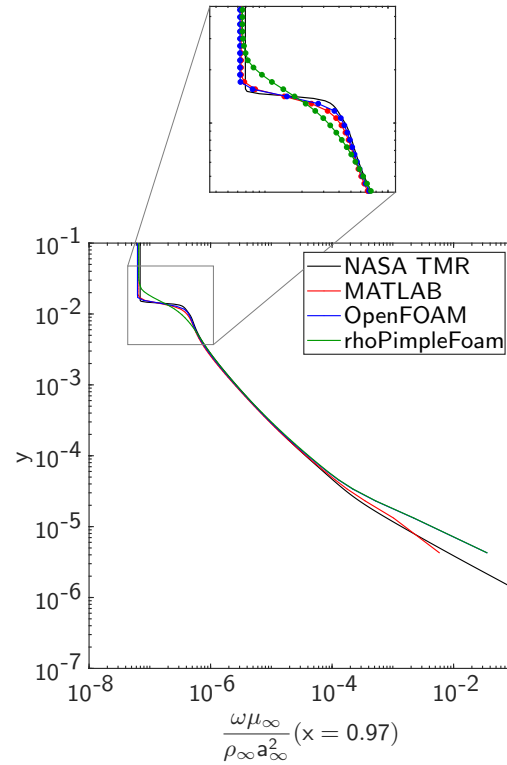


Figure 6: Profile of nondimensional ω at $x = 0.97$

6 Conclusion

The fully-coupled density-based solver for turbulent compressible flow has been successfully implemented to OpenFOAM. Compared to the rhoPimpleFoam solver which is part of OpenFOAM installation our solver provides better resolution of sharp behavior of turbulent quantities near the edge of the boundary layer on the computational mesh with the same number of cells. It is important to note that the resolution of the PIMPLE solver can be further improved by appropriate tuning of parameters depending on particular case while presented solver does not need any additional information. However, since it uses an explicit time integration it puts strong limit on the maximal allowable size of the timestep Δt so the convergence to the steady state is relatively slow. Implementation of some implicit scheme to the developed OpenFOAM solver is therefore needed to increase the computational efficiency.

Acknowledgment

The financial support for the present study was partly provided by the project L01506 and by the project SGS-2019-009 of the Ministry of Education, Youth and Sports of the Czech Republic.

References

- [1] Weller, H. G., Tabor, G., Jasak, H. & Fureby, C.: A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics*, vol. 12, no. 6: (1998) pp. 620-631.
- [2] Issa, R. I., Gosman, A. D. & Watkins, A. P.: The Computation of Compressible and Incompressible Recirculating Flow by a Non-Iterative Implicit Scheme. *Journal of Computational Physics*, vol. 62, no. 1: (1986) pp. 429-442.

-
- [3] Ferziger, J. & Perić. M.: *Computational Methods for Fluid Dynamics*. 3rd edition. Springer, Berlin, 2002.
 - [4] Greenshields, C., Weller, H., Gasparini, L. & Reese, J.: Implementation of semidiscrete, non-staggered central schemes in a colocated, polyhedral, finite volume framework, for highspeed viscous flows. *International Journal for Numerical Methods in Fluids*, vol. 63, no. 1: (2009) pp. 1-21.
 - [5] Romanelli, G., Seroli, E. & Mantegazza, P.: A new accurate compressible inviscid solver for aerodynamic applications. In *3rd OpenFOAM Workshop by Dipartimento di Energetica, Politecnico di Milano*, 2008.
 - [6] Heyns, J.A., Oxtoby, O.F. & Steenkamp, A.: Modelling high-speed flow using a matrix-free coupled solver. In *9th OpenFOAM Workshop*, Zagreb, Croatia, 2014.
 - [7] *HiSA*: <https://hisa.gitlab.io/>. Accessed: 2020-01-08.
 - [8] Levý, T.: *Modeling of compressible turbulent flows using suitable RANS turbulence models*. Diploma thesis, University of West Bohemia, Faculty of Applied Sciences, 2019.
 - [9] Rumsey, C. L.: *NASA Turbulence Modeling Resource*. Langley Research Center. <https://turbmodels.larc.nasa.gov/>. Accessed: 2020-01-08.
 - [10] Wilcox, D. C.: *Turbulence Modeling for CFD*. 3rd edition. San Diego: DCW Industries, 2006.
 - [11] White, F. M.: *Viscous fluid flow*. 3rd edition. New York: McGraw-Hill, 2006.
 - [12] Menter F. R.: Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA Journal*, vol. 32, no. 8: (1994) pp. 1598-1605.
 - [13] Menter F. R.: Influence of freestream values on k-omega turbulence model predictions. *AIAA Journal*, vol. 30, no. 6: (1992) pp. 1657-1659.
 - [14] Blazek J.: *Computational Fluid Dynamics: Principles and Applications*. 3rd edition. Butterworth-Heinemann, 2015.
 - [15] Liou M.-S. & Steffen C. J.: A New Flux Splitting Scheme. *Journal of Computational Physics*, vol. 107, no. 1: (1993) pp. 23-39.
 - [16] Barth, T. & Jespersen, D.: The design and application of upwind schemes on unstructured meshes. In *27th Aerospace Sciences Meeting*. Reston, Virginia: AIAA, 1989.
 - [17] Moukalled, F., Mangani, L. & Darwish, M.: *The Finite Volume Method in Computational Fluid Dynamics: An Advanced Introduction with OpenFOAM and Matlab*. 1st edition. Springer Publishing Company, 2015.
 - [18] LeVeque R. J.: *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.