# Implementation of semi-implicit solvers for compressible flow simulations into OpenFOAM

Ing. Martin Kožíšek
Vedoucí práce: Doc. Ing. Jiří Fürst, Ph.D.

***Abstrakt***

*Tento příspěvek se zabývá implementací dvou řešičů metody konečných objemů pro numerickou simulaci proudění stlačitelné tekutiny do výpočetního prostředí OpenFOAM. První řešič obsahuje algoritmus tlakových korekcí, druhý obsahuje algoritmus navzájem svázaných korekcí tlaku a teploty. Oba řešiče používají AUSM interpolaci. Na závěr jsou uvedeny výsledky numerické simulace proudění stlačitelné tekutiny v testovacích úlohách, které potvrzují, že implementace jsou vhodné pro simulace vysokorychlostního proudění zahrnujícího i transsonické a supersonické rychlosti.*

***Abstract***

*This paper shows results of the project concerned with implementation of the pressure correction algorithm and the coupled pressure and temperature correction algorithm into OpenFOAM. Both finite volume method algorithms use AUSM interpolation. Numerical results for compressible flow test cases confirm that implementations are well suited for calculations of high-speed flows including transonic and supersonic flows.*

***Keywords***

*CFD compressible flow FVM turbine cascade AUSM.*

## 1. Introduction

In recent years computational fluid dynamics (CFD) has become inseparable part of turbomachinery design and flow analysis. Commercially available CFD software can be accompanied by in-house (e.g. opensource) software. The aim is to create the CFD solver suitable for simulations of transonic and supersonic flows particularly like flows through turbine cascades. Consequently, we have to solve Navier-Stokes equations with terms which respect the compressibility. Governing equations describing flow of ideal gas read (1), (2), (3) and (4).

$$\frac{\delta\rho}{\delta t} + \nabla \cdot (\rho\vec{u}) = 0 \tag{1}$$

$$\frac{\delta(\rho\vec{u})}{\delta t} + \nabla \cdot (\rho\vec{u} \otimes \vec{u}) + \nabla p = \nabla \cdot \vec{\vec{\tau}} \tag{2}$$

$$\frac{\delta(\rho E)}{\delta t} + \nabla \cdot [(\rho E + p)\vec{u}] = \nabla \cdot \left(\vec{\vec{\tau}} \cdot \vec{u}\right) - \nabla \cdot (\lambda \nabla T) \tag{3}$$

$$p = \rho r T \tag{4}$$

## 2. OpenFOAM

OpenFOAM is a free open source software that can solve a wide range of problems in continuum mechanics using finite volume method. OpenFOAM includes also ready-made solvers for compressible flows, however currently (version 2.1) they are considered more as weakness of OpenFOAM project. Nevertheless we chose OpenFOAM 1.6-ext as C++ toolbox

for implementation of own solver. Correct implementation gave us useful opportunities. We are able to switch between 1D, 2D or 3D space, compute parallel, use turbulent models or use post processing tools and more.

## 3. AUSM interpolation

Finite volume method operates with interpolations of values from cell centres to cell faces. The interpolation should respect a character of partial differential equations they are solving. For example the unsteady inviscid compressible Navier-Stokes equations are a mixed set of hyperbolic-parabolic equations in time. The character changes between subsonic (parabolic) and supersonic (hyperbolic) flows. Accordingly, the Advection Upstream Splitting Method (AUSM) was used. AUSM is Mach dependent interpolation provided by polynomial splitting functions. They are switching between central interpolations for subsonic flows and upwind interpolation for supersonic flows. AUSM is known as a simple, robust and first order accurate method.

## 4. Implementation

The recently presented solver [1] has involved own implementation of explicit AUSM method named explicitAUSMFoam. This solver is well suited for calculations of transonic flows for its accurate capturing shocks, but impractically small time step (5) is required. Hence we decided for semi-implicit algorithm, which allows calculating with greater time step. Algorithm is designed in order to disappear speed of sound $c$ from CFL condition (6). The time step is significantly increased in a boundary layer, where is fine mesh and low speed.

$$\Delta t_{1\mathrm{D}} \leq \frac{\Delta x \cdot CFL_{max}}{u+c} \tag{5}$$

$$\Delta t_{1\mathrm{D}} \leq \frac{\Delta x \cdot CFL_{max}}{u} \tag{6}$$

### 4.1 Pressure correction algorithm

Predictor step for density (7) and momentum (8) is the first step in this algorithm [2]. Predictor step is indicated with a superscript *. Other predictor quantities are computed from these values.

$$\frac{\rho^* - \rho^n}{\Delta t}\Delta x + \nabla \cdot (\rho^* u^n) = 0 \tag{7}$$

$$\frac{(\rho u)^* - (\rho u)^n}{\Delta t}\Delta x + \nabla \cdot [(\rho u)^* u^n] = -\nabla(p^n) \tag{8}$$

Predictor step is followed by a corrector step denoted ´. The pressure correction equation is derived from equation (9), where is the total energy expended. Final pressure correction equation reads (10), where $A_{i,j}$ and $C_i$ are linear equation system coefficients. The momentum corrector equation (11) comes from subtraction of momentum equations for new time level and predictor level. Indeed, some simplifications are needed. The last step in this algorithm is an update (12) on the new time level. Equations are written for 1D space. The implementation we have named implicitAUSMFoam.

$$(\rho E)^{n+1} = (\rho E)^* + \left.\frac{\delta(\rho e)^*}{\delta p}\right|_{T=cst} \cdot p' + \left.\frac{\delta(\rho e)^*}{\delta T}\right|_{p=cst} \cdot T' \tag{9}$$

$$A_{i,i-1}{p'}_{i-1} + A_{i,i}{p'}_{i} + A_{i,i+1}{p'}_{i+1} = C_i \tag{10}$$

$$(\rho u)' = -\Delta t \cdot \nabla(p') \tag{11}$$

$$p^{n+1} = p^n + p' \text{ , } (\rho u)^{n+1} = (\rho u)^* + (\rho u)' \text{ and so on.} \tag{12}$$

### 4.2 Coupled pressure and temperature correction algorithm

The pressure correction algorithm is a special simplified case of the coupled pressure and temperature correction algorithm, because temperature correction is considered as zero. Unfortunately, the pressure correction algorithm can simulate only flows of perfect gas without heat transfer. Therefore we aimed to the coupled pressure and temperature correction algorithm [2], which has not mentioned restriction. Predictor step is the same as in previous algorithm. Pressure correction equation (13) now contains a temperature correction part. Temperature correction equation is derived from equation (14) and can be written as (15). Note that corrector equations (13) and (15) depend each other and thus we have to solve them in a coupled way. Coupled solution requires to build and solve the block matrix (16). Obviously, block matrix operations take more CPU time as the corrector step from pressure correction algorithm. Finally momentum correction is calculated from (11) and new time level can be determined. The implementation of this algorithm into OpenFOAM was named pTCoupledFoam.

$$A_{i,i-1}p'_{i-1} + A_{i,i}p'_i + A_{i,i+1}p'_{i+1} + B_{i,i}T'_i = C_i \tag{13}$$

$$\rho^{n+1} = \rho^* + \left.\frac{\delta\rho^*}{\delta p}\right|_{T=cst} \cdot p' + \left.\frac{\delta\rho^*}{\delta T}\right|_{p=cst} \cdot T' \tag{14}$$

$$G_{i,i-1}p'_{i-1} + G_{i,i}p'_i + G_{i,i+1}p'_{i+1} + J_{i,i-1}T'_{i-1} + J_{i,i}T'_i + J_{i,i+1}T'_{i+1} = K_i \tag{15}$$

$$\begin{bmatrix} A_{i,i-1} & 0 \\ G_{i,i-1} & J_{i,i-1} \end{bmatrix}\begin{bmatrix} p'_{i-1} \\ T'_{i-1} \end{bmatrix} + \begin{bmatrix} A_{i,i} & B_{i,i} \\ G_{i,i} & J_{i,i} \end{bmatrix}\begin{bmatrix} p'_i \\ T'_i \end{bmatrix} + \begin{bmatrix} A_{i,i+1} & 0 \\ G_{i,i+1} & J_{i,i+1} \end{bmatrix}\begin{bmatrix} p'_{i+1} \\ T'_{i+1} \end{bmatrix} = \begin{bmatrix} C_i \\ K_i \end{bmatrix} \tag{16}$$

## 5. Results

Newly implemented CFD solvers were validated using following test cases.

### 5.1 Turbine cascade SE1050

ImplicitAUSMFoam, pTCoupledFoam and standard explicit OpenFOAM solver rhoCentralFoam were applied for solving 2D inviscid transonic flow through SE1050 turbine cascade (Fig. 1). The transonic regime $M_{2is}$ = 1,198 and inlet angle $\alpha$ = 19.3° are considered. The grid consists of 6000 triangular cells. Inlet boundary conditions are: total pressure $p_0$ = 100 kPa, total temperature $T_0$ = 293.15 K and inlet flow angle. At the outlet are: static pressure $p_2$ = 41.301 kPa and homogeneous Neumann conditions for velocity and temperature. The blade boundary is non-permeable wall. The rest of boundaries are the periodic type. Convergence speed (Fig. 2) of pTCoupledFoam confirms, that CPU time consumption is higher via block matrix operations. The pressure on the blade is shown in Figure 4. In-house solvers give comparable results with experiment data [3]. The unphysical peak on the leading edge highlighted by arrow is generated by rhoCentralFoam.

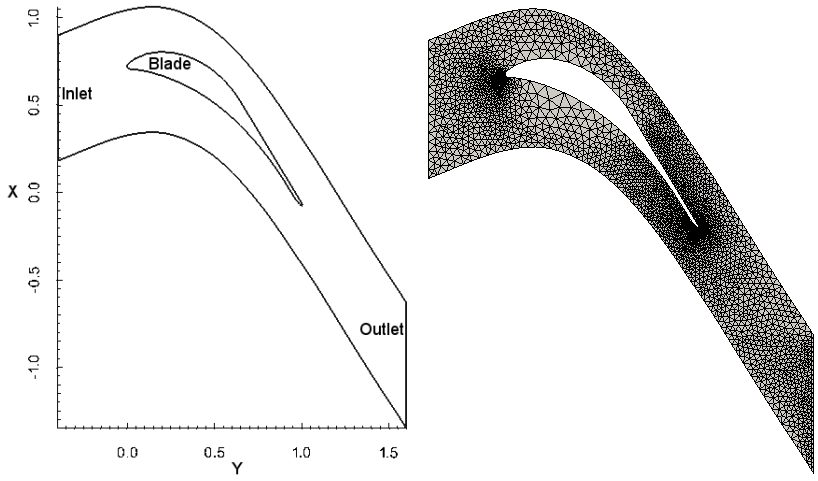


***Fig. 1.*** *2D geometry of SE 1050 and the unstructured Mesh with 6000 triangular cells.*

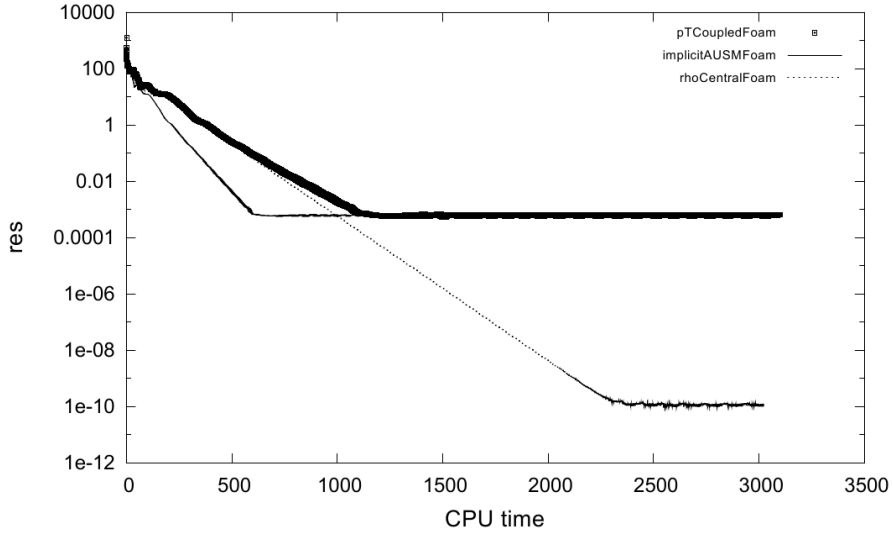


***Fig. 2.*** *Convergence history diagram.*

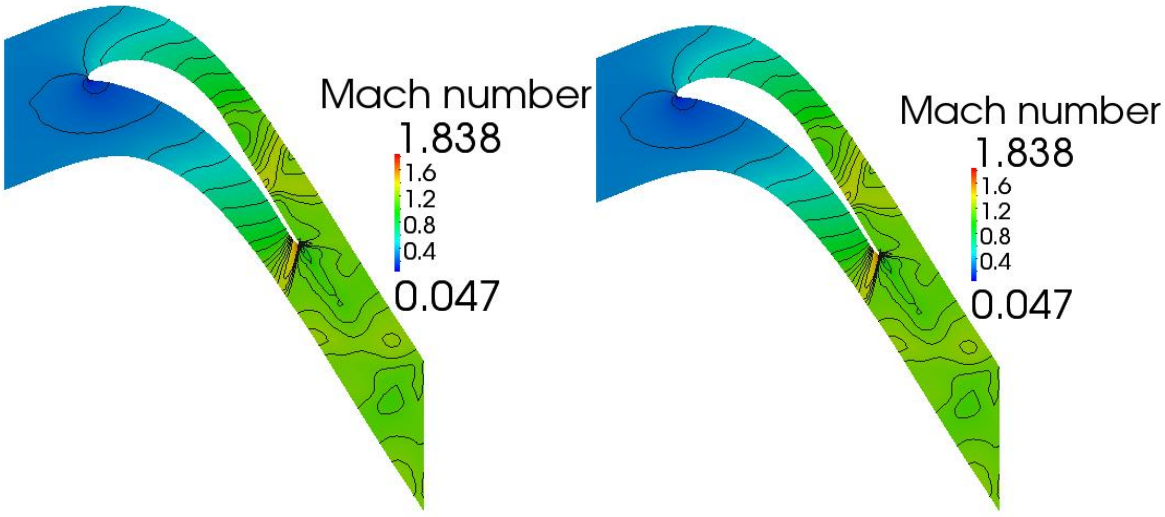


***Fig. 3.*** *Mach number distribution (left implicitAUSMFoam, right pTCoupledFoam). Results are 1st order accurate in space and time.*

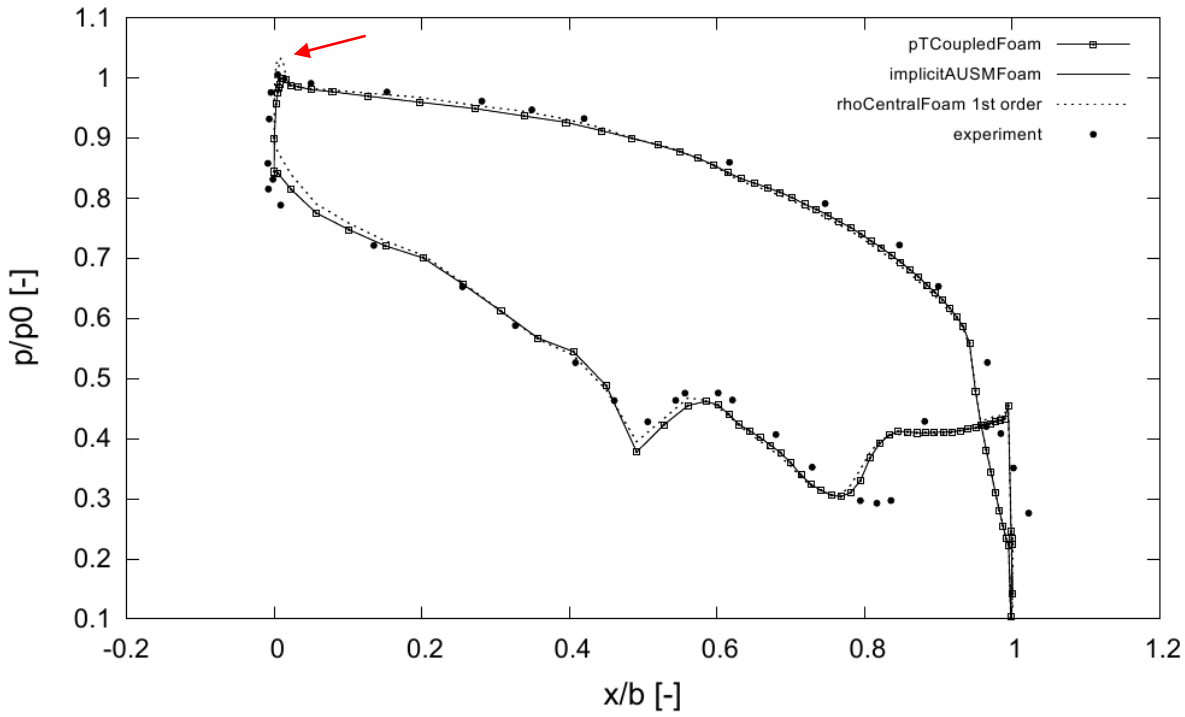


***Fig. 4.*** *Pressure distribution on the blade for first order accurate simulation. The results are scaled by chord length b and inlet total pressure $p_0$.*

## 5.2 GAMM channel

As a second test case 2D flow through GAMM channel is considered. Figure 5 illustrates the geometry of the channel. Inlet boundary conditions are: total pressure $p_0$ = 100 kPa, total temperature $T_0$ = 293.15 K and homogeneous Neumann condition for velocity. At the outlet are: static pressure $p_2$ = 73.7 kPa and homogeneous Neumann conditions for velocity and temperature. The rest of boundaries are non-permeable walls. Transonic regime $M_{2is}$ = 0.675 is provided by these boundary conditions. Numerical solution was obtained from ImplicitAUSMFoam, pTCoupledFoam and standard implicit OpenFOAM solvers sonicFoam and rhoSimplecFoam. Mach number distribution along a lower wall of channel is shown in Figure 7. It is obvious, that implicitAUSMFoam and pTCoupledFoam captured the shock wave more exactly. OpenFOAM solvers smeared the shock wave in spite of the fact that they are intended for compressible flow simulations.

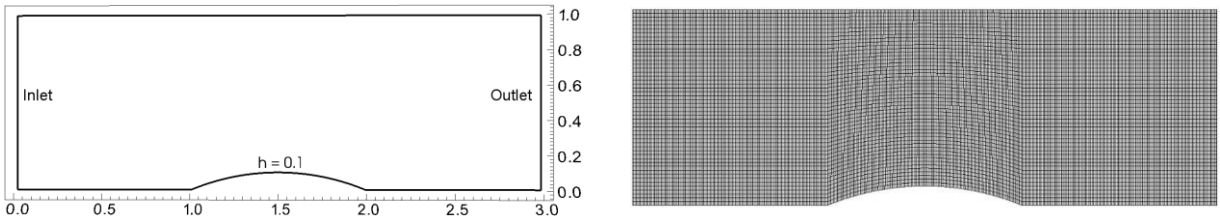


***Fig. 5.*** *The geometry of 2D GAMM channel and structured quadrilateral grid with 150x50 cells.*

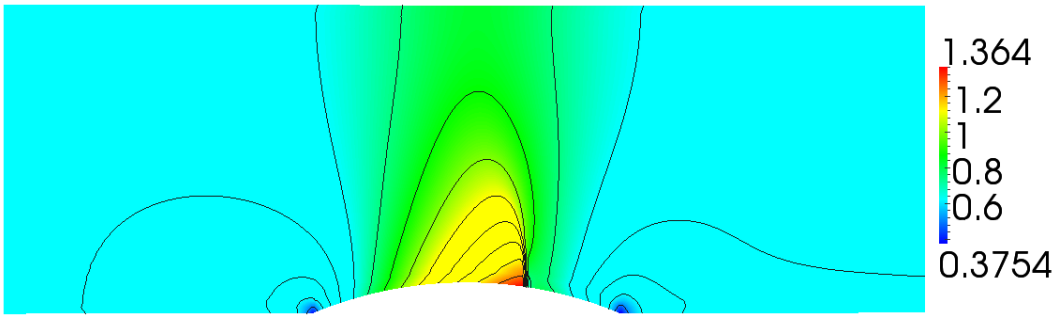


***Fig. 6.*** *Mach number distribution simulated by ImplicitAUSMFoam.*

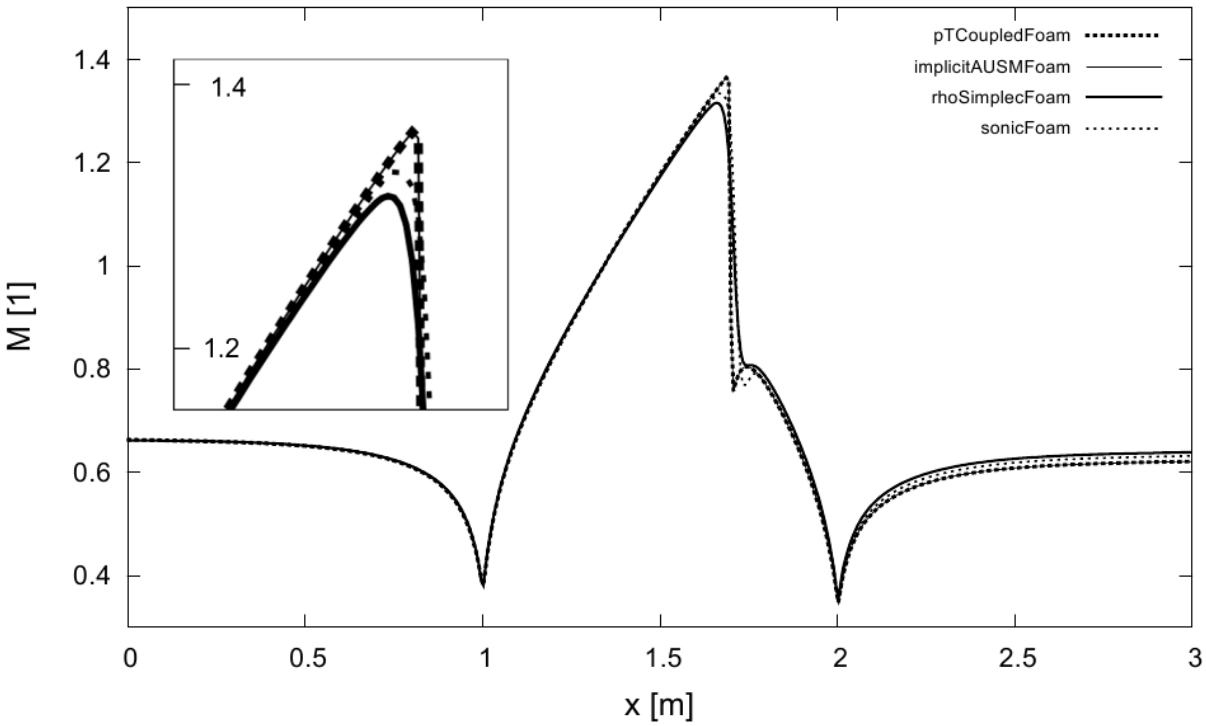


***Fig. 7.*** *Mach number distribution along a lower wall of GAMM channel.*

## 6. Conclusions

We have implemented new C++ codes implemented into OpenFOAM 1.6-ext. These codes perform the pressure correction and the coupled pressure and temperature correction algorithms. The important feature is that the highest allowable time step increased from (5) to (6). In contrary to implicitAUSMFoam the pTCoupledFoam is more complicated due block matrix operations. On the other hand pTCoupledFoam has no restriction on simulations of perfect gas without heat transfer. First numerical results for compressible flow test cases confirm that implementations are well suited for calculations of high-speed flows, although both solvers are still in development.

***List of symbols***

| | | |
|---|---|---|
| $A$ | *linear equation system coefficient* | (-) |
| $B$ | *linear equation system coefficient* | (-) |
| $b$ | *chord length* | (m) |
| $C$ | *linear equation system coefficient* | (-) |
| $c$ | *speed of sound* | $(\mathrm{m}\cdot \mathrm{s}^{-1})$ |
| $CFL_{max}$ | *maximal Courant number* | (1) |
| $e$ | *specific internal energy* | $(\mathrm{J}\cdot \mathrm{kg}^{-1})$ |
| $G$ | *linear equation system coefficient* | (-) |
| $J$ | *linear equation system coefficient* | (-) |
| $K$ | *linear equation system coefficient* | (-) |
| $M$ | *Mach number* | 1 |
| $M_{2is}$ | *outlet isentropic Mach number* | 1 |
| $p$ | *pressure* | (Pa) |
| $p_0$ | *total pressure* | (Pa) |
| $p_2$ | *outlet static pressure* | (Pa) |
| $r$ | *specific gas constant* | $(\mathrm{J}\cdot \mathrm{kg}^{-1}\cdot \mathrm{K}^{-1})$ |
| $T$ | *temperature* | (K) |
| $T_0$ | *total temperature* | (K) |
| $t$ | *time* | (s) |
| $u$ | *velocity* | $(\mathrm{m}\cdot \mathrm{s}^{-1})$ |
| $x$ | *length* | (m) |
| $y$ | *length* | (m) |
| $\alpha$ | *inlet angle* | (°) |
| $\Delta x$ | *spatial step* | (m) |
| $\Delta t$ | *time step* | (s) |
| $\lambda$ | *thermal conductivity* | $(\mathrm{W}\cdot \mathrm{m}^{-1}\cdot \mathrm{K}^{-1})$ |
| $\rho$ | *density* | $(\mathrm{kg}\cdot \mathrm{m}^{-3})$ |
| $\rho E$ | *total energy* | $(\mathrm{J}\cdot \mathrm{m}^{-3})$ |
| $\rho u$ | *momentum* | $(\mathrm{kg}\cdot \mathrm{m}^{-2}\cdot \mathrm{s}^{-1})$ |
| $\bar{\bar{\tau}}$ | *shear stress tensor* | (Pa) |
| $i$ | *cell index* | (-) |
| $n$ | *time level index* | (-) |
| $*$ | *predictor* | (-) |
| $'$ | *corrector* | (-) |

***References***


[1] Kožíšek Martin, Fürst Jiří: Implementation of Explicit Advection Upstream Splitting Method into OpenFoam. In Conference Topical Problems of Fluid Mechanics 2012. Prague: Institute of Thermomechanics AS CR, v. v. i., p. 65-66. ISBN 9788087012406.

[2] Krista Nerinckx, Jan Vierendeels, Erik Dick: Mach-uniformity through the coupled pressure and temperature correction algorithm, JCP 206 (2005).

[3] http://uriah.dedi.melbourne.co.uk/w/index.php/AC_6-12_Test_Data



**Acknowledgements:** The work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS 13/174/OHK2/3T/12.

The authors would like to express their thanks to the Technology Agency of the Czech Republic, which supported this research under grant No. TA03020277 and institutional support RVO: 61388998 are also gratefully acknowledged.