Incompressible Flow Solver by Means of Pseudo-Compressibility Method

Árpád Veress
Department of Aircraft and Ships
Budapest University of Technology and Economics
H–1111 Budapest, Sztoczek u. 6. J. ép., Hungary

HU ISSN 1418-7108: HEJ Manuscript no.: ANM-030110-A

Abstract

In this project a new computational code is developed using cell centered finite volume method with constant space-wise approximation over a control volume to solve the Euler equations, which is modified by means of artificial compressibility method. The code is based on a two dimensional second order central discretization using an adoption of Jamenson's artificial viscosity for stability [2]. Explicit 4^{th} order Runge-Kutta time integration is applied to evaluate steady state conditions. At the outer surface of the computational domain, an extrapolation type boundary condition is used. For validation, Rizzi A. W. test case is implemented [5]. As a future plan, a solid wall technique is going to be developed for boundary condition remains to decrease the computational time. On the other hand, this code is intent to be the basic of later improvements, namely an extension to 3D and the construction of Navier-Stokes solver to analyse and optimize any kind of fluid machinery.

Keywords: Euler solver, artificial viscosity, pseudo-compressibility method, finite volume method

1 Introduction

The discretisation of the incompressible Euler or Navier-Stokes equations requires particular consideration since the time derivative of the density does not appear. Hence, the time-dependent methods suitable for the compressible equations cannot be applied without adaption.

There are two main approach to realize adaption mentioned above; vorticity-stream function approach and primitive-variable approach.

In the vorticity-stream function approach, the velocity components are replaced by the vorticity (ω) and the stream function (ψ) in the fluid dynamics formulation. The solution of vorticity and stream function is obtained to describe the flow field. The velocity field can be calculated back according to definition of stream function. The pressure term is not explicitly in the formulation. Hence, a new equation is necessary to introduce. By the Poisson equation for pressure, which can be derived from momentum equations and in which the pressure is in the function of velocity components or vorticity-stream function, pressure can be determined.

The extension of this method to 3D problems is not straightforward, since a stream function does not exist for a truly 3D flow. Several algorithms have been developed to cure this limitation: 1. Vorticity-potential method: the formulation can be generalized to 3D by making use of a vector potential. This method may require more computational effort than the primitive-variable approach. 2. Dual-potential method: the velocity is decomposed as a scalar potential and a vector potential. The solution is solved for these two potentials. Treatment of boundary conditions of this method is rather complex (Morino, 1986; Gegg et al., 1989). 3. Vorticity-velocity method: the solution is obtained by solving the vorticity and velocity together. Several applications of this method can be found (Agarwal, 1981; Gastski, et al. 1982; Gui, Stella, 1988) [7].

The extension of formulation of above methods is not a general approach for solving 3D flow, since their numerical schemes and specification of boundary conditions are complicated. On the other hand the primitive-variable approach does not have such complicated formulation when it being applied to 3D flow.

In the primitive-variable approach the incompressible Euler and Navier-Stokes equations are most often solved in their primitive variable form (u, v, w, p) for 3D problems. Even for 2D problems, the use of primitive variables is quite common. Two broad categories of the numerical methods in primitive-variable approach are next: the pressure correction approach and the coupled approach.

The methods falling in the class of pressure correction approach can be applied to the stationary as well as to the time dependent incompressible flow equations. They consist of a basic iterative procedure between the velocity and pressure field. For an initial approximation of the pressure, the momentum equation can be solved to determine the velocity field. The obtained velocity field does not satisfy the divergence-free continuity equation and has therefore to be corrected. Since this correction has an impact on the pressure field, a related pressure correction is defined, obtained by showing, that the corrected velocity satisfies the continuity equation. This leads the most often to a Poisson equation for the pressure correction. The three most widespread technique are MAC (Marker and Cell) method (Harlow, Welch, 1965), SIMPLE (Semi Implicit Method for Pressure Linked Equations) family methods (SIMPLE, SIMPLEC, SIMPLER) (Caretto at al., 1972; Patankar, 1980) and PISO (Primitive-Variable) Implicit Separator method (Issa, 1986) [4].

In the coupled type approach the pressure is adopted in the fluid dynamics equations, so there is no need to be calculated separately. For stationary flows, a structure similar to the to the compressible equations can be recovered by adding an artificial compressibility term under the form of the time derivative of the pressure added to the continuity equation. When steady state is reached, this term vanishes. This type methods are also applicable to describe transient phenomena by using dual time stepping. This method is the pseudo-compressibility method, which was originally introduced by Chorin (1967) for finite difference approximation, and this idea is used in this report [1].

All the numerical computations were performed at the Center of Information Systems (CIS) of the Budapest University of Technology and Economics on a supercomputer. This serverfarm contains four Compaq 4100 nodes: 16 x EV5.6 (21164A, 600 Mhz, 8 MByte cache) Alpha CPU, 32 GByte memory, 0.62 TByte hard-disc. The nodes communicate by Memory Channel (Full/100 Mb).

2 Euler Solver

2.1 Governing Equations

The 2 dimensional incompressible Euler equations with the extension of the pseudo-incompressible term in dimensional form are appeared:

$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} + \frac{\partial G(U)}{\partial y} = 0 \tag{1}$$

with

$$U = [P/\beta^2,\ u,\ v]^t$$
$$F = [u,\ u^2 + P,\ uv]^t$$
$$G = [v,\ uv,\ v^2 + P]^t$$

Where $P = p/\rho$, p is the static pressure, ρ is the density, u and v are Cartesian components of velocity.

In order to satisfy the consistency property, when the steady state condition is reached, the first term of the continuity equation is vanished, which means that the original form of the incompressible Euler equations in dimensional form are appeared.

System (1) can be written in compact form with the introduction of total flux vector, $\vec{H}$:

$$\vec{H} = F\vec{e}_x + G\vec{e}_y \tag{2}$$

$$\frac{\partial U}{\partial t} + \vec{\nabla}\vec{H}(U) = 0 \tag{3}$$

Integrating system (3) over a control volume Ω, which is bounded by interface Γ, and applying the Gauss divergence theorem, one gets:

$$\frac{\partial}{\partial t}\int\int_{\Omega} U d\Omega + \int_{\Gamma} \vec{H}\vec{n} d\Gamma = 0 \tag{4}$$

where $\vec{n} = (n_x, n_y)$ is the local outward pointing unit vector normal to the boundary surface, as it is shown in figure (1). Equation (4) simply states that, the rate of change of the conservative variables in a volume Ω is balanced by the net flux $\vec{H}$ passing through the boundary Γ of Ω volume [3].

In the finite volume approach, first, one should evaluate the contour integral of the total flux vector $\vec{H}$ in equation (4). It is convenient to define total flux normal to the surface of the elementary control volumes Ω, rather than making use of the individual cartesian components. Hence, for the Euler equations the total flux in terms of normal component of the velocity V_n ($V_n = \vec{V}\vec{n} = (u\vec{e}_x + v\vec{e}_y)(n_x\vec{e}_x + n_y\vec{e}_y) = un_x + vn_y$) yields:

$$H_n = \vec{H}\vec{n} = \begin{pmatrix} V_n\beta^2 \\ uV_n + Pn_x \\ vV_n + Pn_y \end{pmatrix} \tag{5}$$

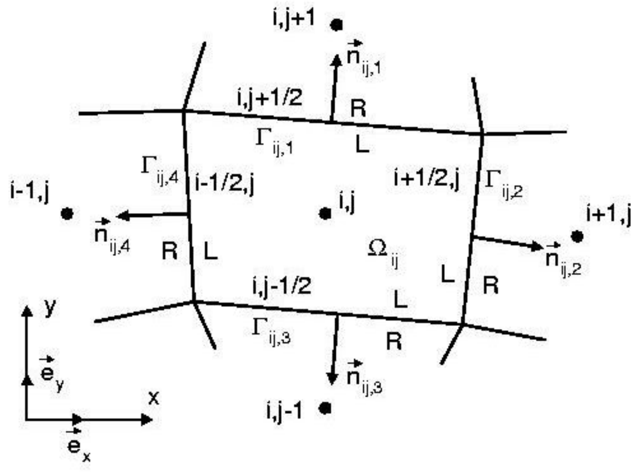

Figure 1: Cell-centered finite volume formulation

with

$$U = [P,\ u,\ v]^t$$

equation (4) can be written as:

$$\frac{\partial}{\partial t}\int\int_{\Omega} U d\Omega + \int_{\Gamma} H_n d\Gamma = 0 \tag{6}$$

According to numerical experiments, the appropriate choice of pseudo coefficient; β has an indispensable effect for the scheme to be accurate and convergent. In this case β can vary from 3 to 7 and the results are shown at $\beta = 3$.

2.2 Cell Centered Finite Volume Spatial Discretization

The spatial discretization starts with setting up a mesh or grid, by which the flow domain is replaced by a finite number of points, in which the numerical values of the variables will be determined. In this work an H-type mesh is used, which symbolize that the physical domain is divided into the set of grids consisting of pitch-wise lines in y direction and quasi-streamlines.

Concerning the system (6), in order to pass from continuous to a discrete form, a choice about the type of representation of the solution vector over the finite volume has to be made. Having partitioned the computational domain, in a finite number of volumes, the discrete unknown U_j of the generic point j, is defined as:

$$U_j \equiv \frac{1}{\Omega_j}\int\int_{\Omega_j} U d\Omega \tag{7}$$

This is consistent with the so called cell center approach. Hence, the unknown vector U has to be interpreted as a mean value over the control volume rather than a nodal value as in the cell vertex formulation. By substituting (7) into the first integral in equation (6), and replacing the second integral by a summation over the number of faces N_f of the chosen control volume Ω_j, equation (6) can be written in the following semi-discrete form for the point j:

$$\frac{\partial}{\partial t}U_j = -\frac{1}{\Omega_j}\sum_{k=1}^{N_f}[H_n]_{j,k}\Gamma_{j,k} \tag{8}$$

where $[H_n]_{j,k}$ is the total inviscid flux normal to the cell interface with the length (surface in 3D) of $\Gamma_{j,k}$ cell boundary exchanged between points j and k. The quantity $[H_n]_{j,k}$ will be referred to, in the following, as the numerical flux function $\widetilde{H}_n$. As the lowest level the flux function is constructed from a piecewise constant data base, that is:

$$\widetilde{H}_n = \widetilde{H}_n(U^L, U^R) = \frac{\widetilde{H}_n(U^L) + \widetilde{H}_n(U^R)}{2} \tag{9}$$

where $U^L = U_j$ and $U^R = U_k$ are the state values on both side of the surface under consideration. Equation (9) corresponds to a first order accurate spatial discretization [3].

2.3 Artificial Dissipation

All second-order central-discretization schemes, even with a stable time-step, suffer from some instabilities. The problem is the spurious mode can be defined as an unphysical set of discrete values satisfying the scheme at interior points and vanishing at the nodes where a boundary condition is imposed. By this phenomena the odd and even mesh points are decoupled and it is caused by the scheme itself. In order to make the computation stable and cure the numerical artefact described above, some explicitly added higher order dissipation; artificial viscosity may be introduced. In the present work Jamenson's [2] artificial viscosity is used, but of course in this incompressible case the second order part of this numerical dissipation is negligible because of subsonic flow. So, the equation (8) becomes

$$\frac{\partial}{\partial t}U_j = -\frac{1}{\Omega_j}\sum_{k=1}^{N_f}[H_n]_{j,k}\Gamma_{j,k} + \frac{1}{\Omega_j}D \tag{10}$$

where

$$D = D_x + D_y \tag{11}$$

and

$$D_x = d_{i+\frac{1}{2},j} - d_{i-\frac{1}{2},j} \tag{12}$$

$$D_y = d_{i,j+\frac{1}{2}} - d_{i,j-\frac{1}{2}} \tag{13}$$

The all terms on right side have a similar form, for example across the cell interface between the nodal values of (i,j) and $(i+1,j)$:

$$d_{i+\frac{1}{2},j} = -\varepsilon^{(4)}_{i+\frac{1}{2},j}(U_{i+2,j} - 3U_{i+1,j} + 3U_{i,j} - U_{i-1,j}) \tag{14}$$

Where the typical value of coefficients $\varepsilon^{(4)}$ is $\frac{1}{128}$.

2.4 Boundary Treatment

The appropriate choice of boundary condition is indispensable for the code to be convergent and accurate and it has an effect for the speed of convergence. The number of physical and numerical parameters, must be imposed and computed, are determined by theory of characteristic, which is particular for such as hyperbolic type systems where the information propagation is dominant. The most widespread boundary conditions to compute the numerical variables are based on the theory of characteristic, extrapolation techniques and compatibility relations. In order to avoid the perturbation reflection at the boundary and so to make the convergence faster, although, the characteristic type boundary condition is preferred, in which the Riemann invariant is kept constant between two states, but in this case, as a first approach an extrapolation technique is used to determine numerical boundary conditions. In order to make all unknowns determinable at the boundary, a set of variables are imposed as physical boundary conditions.

2.4.1 Inlet Boundary Conditions

In case of characteristic variables, two eigenvalues of the Jacobian are positive: $V_n,\ V_n+c_p$ and one is negative V_n-c_p with speudo-sonic speed: $c_p=\sqrt{V_n^2+\beta^2}$, suppose, for incompressible flow that the pseudo-Mach number is less then 1 $(M_p=\frac{V_n}{c_p})$. The examination of eigenvalues says that at the inlet boundary interface two characteristic lines come from the outside of the computational domain, so two physical boundary condition must be imposed, while one characteristic line goes out from the domain of interested, so one numerical boundary condition must be known. In this case the next imposed set of variables are chosen: the total pressure p_0 and the angle of attack α. At the extrapolation type boundary condition, a static pressure p is a primitive variables, which is the function of the interior unknowns, which should be determined by the information propagates from the interior towards the outside of the computational domain. This has been done by a simple first order extrapolation technique. Now, the static boundary pressure is known, hence the velocity V can be computed by:

$$p_{to}=p_{st}+\frac{V^2}{2}\rho \tag{15}$$

Finally the u and v velocity components are possible to determine in the function of inlet flow angle α and V velocity.

2.4.2 Outlet Boundary Conditions

At the outlet, there are one ingoing; V_n-c_p and two outgoing; $V_n,\ V_n+c_p$ characteristics, so one physical and two numerical boundary conditions have to be imposed. It is common practice to fix the static pressure, while the velocity component u and v are extrapolated from the interior.

2.4.3 Solid Wall Boundary Conditions

For a solid boundary condition, a mirror type wall model is adopted, see figure (2). There is only one characteristic enters to the flow domain and only a single physical boundary condition is to be imposed: $\overrightarrow{V}_n=0$. Hence, the normal velocity component in ghost cell

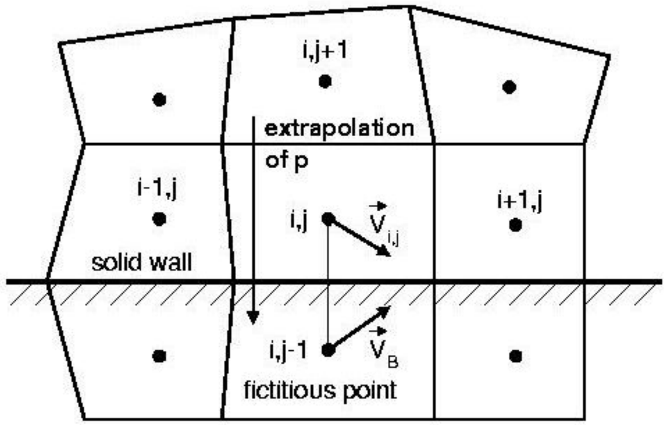

Figure 2: Mirror-type solid wall model

is equal to the inside ones, but with opposite direction, while the parallel component are the same. As a consequence, all convective flux components through the solid boundary will vanish, as soon as the steady condition is reached. Only one parameter (p) remained to extrapolate from the interior by zero ($p_i = p_B$) or first order type.

2.5 Convergence

In this part the L_2 norm of the normalized pressure residuals are computed by:

$$\|\frac{\Delta p}{p}\| = log_{10}\sqrt{\frac{1}{N_p}\sum_{i=1}^{N_p}(\frac{\Delta p_i}{p_i})^2} \tag{16}$$

The figure (3) shows convergence for the extrapolation type boundary condition. This slow convergence is caused, the most probably, by the perturbation reflection at the boundary, which is not dumped. The computational time (in the function of the initial data) was approximately 15 minutes.

2.6 4^{th} Order Runge-Kutta Method for Time Stepping

A couple of stable time stepping method, with the appropriate constraints, are possible to apply to solve the ordinary differential equation given by (10):

$$\frac{\partial}{\partial t}U_j = -\frac{1}{\Omega_j}\sum_{k=1}^{N_f}[H_n]_{j,k}\Gamma_{j,k} + \frac{1}{\Omega_j}D = \Re(U_j, U_k) \tag{17}$$

For the minimum computational storage and the large stability range with the optimal choice of α_k, the effective 4^{th} Order Runge-Kutta method is used [3]. Omitting the the subscript denoting the cell index, a general family of Runge-Kutta schemes is given by:

$$\begin{array}{rcl} U^0 = & U^n & \\ U^k = & U^0 + \alpha_k \Delta t \Re(U^{(k-1)}) & k = 1, m \\ U^{n+1} = & U^m & \end{array} \tag{18}$$

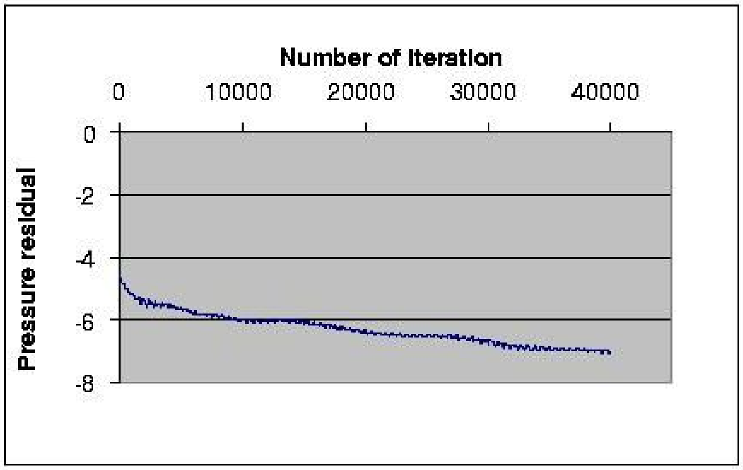

Figure 3: Convergence history of the scheme

where the parameters α_k are such that:

$$0 \ < \ \alpha_k \ \leq 1 \qquad\qquad \alpha_m = 1 \tag{19}$$

Generally, in case of fourth order accuracy in time in the linear case, as mentioned before, let $k = 4$ be and for α_k:

$$\alpha_k = \frac{1}{(4 - k + 1)} \tag{20}$$

For the optimal stability region, determine the coefficients α_k, which maximize the CFL number. So, at the second order space diskretization α_k is given by [3]:

$$\alpha_1 = 1/4 \qquad \alpha_2 = 1/3 \qquad \alpha_3 = 1/2 \qquad \alpha_4 = 1 \tag{21}$$

3 Validation

In order to make sure in the correctness of any calculation, the validation is always necessary to deal with it. Generally, the experiments are the best test to certify our computation, but, because of possibilities, they are not always available. Here, another validation techniques is used: comparison to any commercial or other own code, for example, on the same test cases, such as channel flow over a circular arc bump [2]. Although, this bench mark problem was originally developed for compressible flow, it could be adopted to incompressible flow also.

3.1 Channel flow Over a Circular Arc Bump

This standard test case proposed by Rizzi [5] has been realized. The flow is examined on a relatively coarse grid in the rectangular channel over the circular arc has a maximum thickness of 4.2% in figure (4). The computational domain is divided three parts in order to gain more information about the phenomena is going to be over the circular arc. In the first part, x ∈ [3, 4.35], the grid stretched towards the bump. In the central part, x

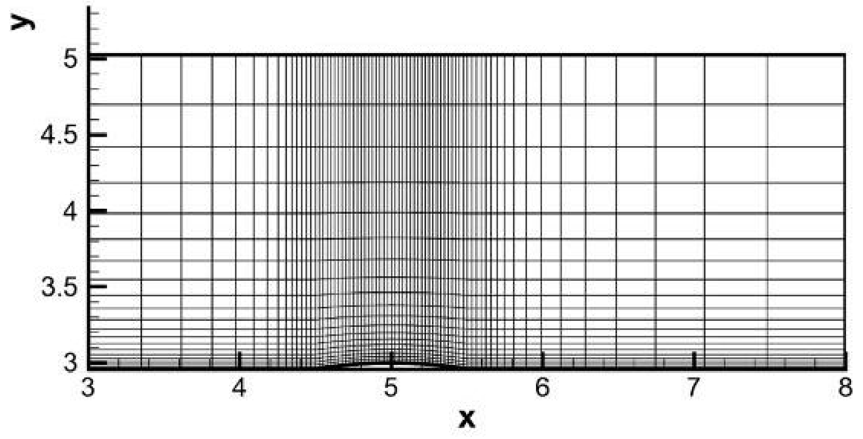

Figure 4: Channel and a circular arc with H-type grid; 72×21 points

∈ [4.35, 5.65], a constant grid spacing is used. Finally in the last part of the channel, x ∈ [5.65, 8], the grid is stretched starting from the bump towards the outlet plane. For the stretching function, which is applied in the y direction also, a simple geometric series type is considered:

$$L = \sum_{i=1}^{N-1} \Delta\xi_i \tag{22}$$

where $\Delta\xi_i = \xi_{i+1} - \xi_i$ is constant and predefined, L is the length to be stretched and N is the number of points. Let R is the ration of two consecutive intervals:

$$R = \frac{\Delta\xi_{i+1}}{\Delta\xi_i} \tag{23}$$

The sum of geometric series is given by:

$$L = \Delta\xi_1 \left(\frac{1 - R^N}{1 - R}\right) \tag{24}$$

which, for a given N, L, is a nonlinear function to be solved with a Newton-Raphson method or Fixed Point Iteration for R, if $\Delta\xi_1$ is given (and greatest value of $\Delta\xi_i$) and vice versa. The inlet total pressure is 100000 Pa, inlet flow angle is 0 and outlet static pressure is 98000 Pa. Density is 1000 $\frac{kg}{m^3}$. The results of this flow field can be found in figure (5) and (6) with two pressure iso-lines are shown on grid size of 144×42.

4 Future Plan

In order to decrease computational time a new soft solid wall boundary condition is going to be developed, which was originally established in the Karim Mazahery Body's Ph. D. thesis in 1992 for compressible flow.

On the other hand this code has an excellent feature to extent to all viscous terms to obtain full Navier-Stokes equation. Hence, DNS (Direct Numerical Simulation) computation or insertion of any turbulence model can be also applied for the governing equation to describe the real flow field in any kind of geometry or fluid machinery.

The 3D extension of the code is also possible without any difficulties.

References

[1] Chorin, A. J. (1967). A Numerical Method for Solving Incompressible Viscous Flow Problems. *Journal of Computational Physics*, 2, 12-26.

[2] Veress, Á., Sánta, I. (2002). A 2D Mathematical Model on Transonic Axial Compressor Rotor Flow. *Periodika Politechnika*, under edition.

[3] Manna, M. (1992). A Three Dimensional High Resolution Compressible Flow Solver. *Ph.D. Thesis in Catholic University of Louvain.*

[4] Hirsch, C. (1989). Numerical Computation of Internal and External Flow. volume 1, 2 of *Series in Numerical Methods in Engineering.* Wiley-Interscience.

[5] Rizzi, A. W. (1981). Numerical Methods for the Computations of Transonic Flows with Shock Waves. *Notes on Numerical Fluid Mechanics*, Vieweg, Braunschweig 3.

[6] Veress, Á. (2001). Inverse Design on Return Flow Channel for Multistage Radial Compressor. *VKI Diploma Course Project Report 2001-27.*

[7] Lecture Notes at the National Central University, Taiwan. Numerical Methods for the Incompressible Navier-Stokes Equations. *Downloaded from the internet by w3new.ncu.edu.tw/ junwu/me681/chap6.doc.*

[8] Veress, Á. (2002). Bevezetés az áramlástan numerikus módszereibe. *Oktatási segédanyag, BME, Repülőgépek és Hajók Tanszék, Budapest.*

[9] Veress, Á. (2000). Simplified Method on Mathematical Model of Transonic Axial Compressors. *22nd ICAS Proceedings: ICA0775P, Harrogate, UK.*

List of Symbols

	Variables	w	z comp. of velocity vector $[m/s]$
c_p	Pseudo-sound speed $[m/s]$	$\overrightarrow{V}$	Velocity vector of comp. $(u,\ v)$
$d\Gamma$	Surface element $[m]$	α	Flow angle
$\overrightarrow{e}$	Unit vector	ρ	Density $[kg/m^3]$
F	Convective flux vector (x component)	ψ	Stream-function
G	Convective flux vector (y component)	ω	Vorticity
$\overrightarrow{H}$	Total flux vector	Ω	Cell volume (area) $[m^2]$
M_p	Pseudo-Mach number		*Indices*
$\overrightarrow{n}$	Outward pointing unit normal	$_{to}$	Total or stagnation condition
p	Pressure [Pa]	$_{st}$	Static condition
t	Time $[s]$	^{n}	Time level
u	x component of velocity vector $[m/s]$	$_{n}$	Normal to boundary
U	Vector of conservative variables	$_{x,\ y}$	Refer to space variables
$x,\ y$	Space variables	$_{i,j}$	Cell or spatial indices
v	y comp. of velocity vector $[m/s]$		

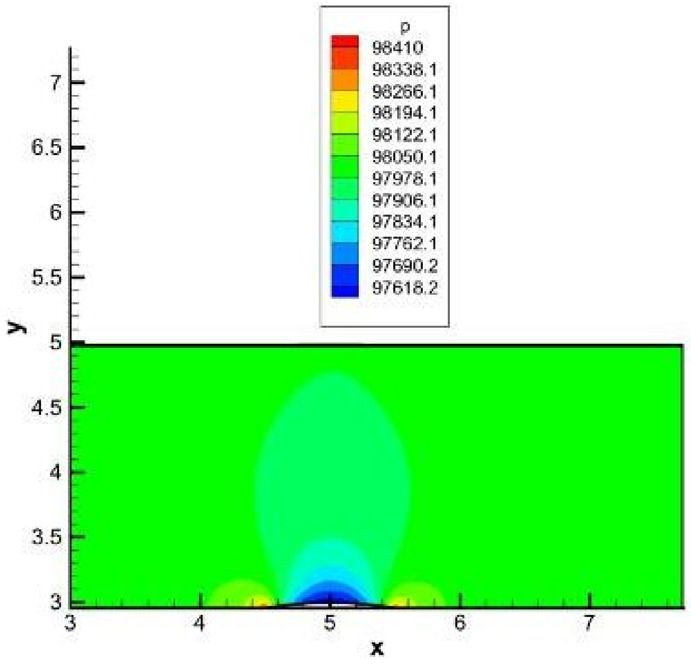

Figure 5: Pressure distribution over the circular arc in the channel with the mesh size of 144×42

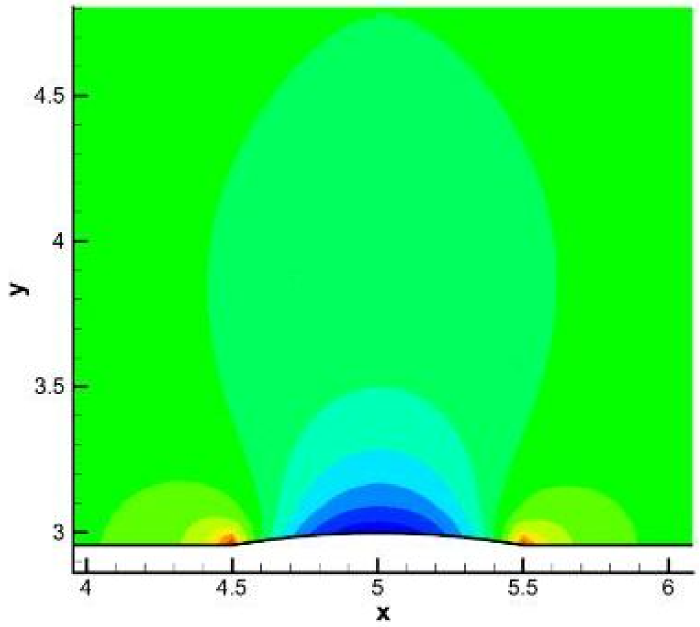

Figure 6: Zoom to pressure distribution over the circular arc in the channel with the mesh size of 144×42