29th Congress of the International Council of the Aeronautical Sciences
St. Petersburg, Russia; September 7-12, 2014

A DEVELOPMENT AND VERIFICATION OF DENSITY BASED SOLVER USING LU-SGS ALGORITHM IN OPENFOAM

Junghyun Kim*, Kyuhong Kim**
***Korea Aerospace Research Institute(KARI), **Seoul National University**

Abstract

A development and verification of density based solver using LU-SGS(Lower Upper Symmetric Gauss Seidel) Algorithm in OpenFOAM(Open Field Operation And Manipulation) was performed. A pressure based solver in OpenFOAM for solving incompressible flow was modified to density based solver for dealing with compressible flows. It was not only developed implicit LU-SGS algorithm instead of an explicit time integration in OpenFOAM but implemented Riemann boundary condition which has not been developed in OpenFOAM. In addition, libraries such as wall shear stress dictionary in OpenFOAM were modified to solve and handle compressible problems. To validate the developed code, some validation models which are widely used were analyzed. Preliminary results showing the comparison between an experiment and computation data indicated that our setup in OpenFOAM was correct.

1 Introduction

The introduction part of this paper has the following sections: Research background, Introduction to OpenFOAM, and Research objectives.

1.1 Research background

It is well known there are three approaches to try to solve the phenomenon of fluid dynamics which is especially called as aerodynamics at aircraft design: 1) Theoretical approach, 2) Experimental method, and 3) Computational fluid dynamics(CFD). However, a solution of the analytical equations that govern the flow still remains a challenging task due to the characteristic of non-linear. An experimental method has also a cost problem. [1]

For these reasons, it is considered that CFD has become a very popular tool for research work in different fields and particularly in fluid dynamics. A CFD can be generally used by both commercial programs such as FLUENT and in-house codes.

For commercial tools, it is so easy that people can handle it with less knowledge of programs. In addition, there is unnecessary to develop the code since many options and solvers have already been installed in the program. A commercial code, however, can be costly due to the license fee and can have many limitations if users want to specify some problems they have to solve.

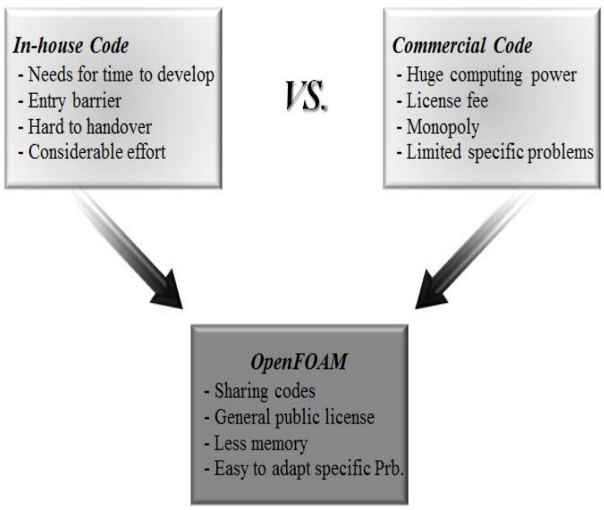

Fig. 1. Why we need to use open source code

For in-house codes, it is considered as powerful tools for people who would like to have conducted their research in specified problems such as hypersonic flow on re-entry vehicles since users can handle the code with perfect freedom. Although it has the highest accuracy in the problem, it requires a lot of times and efforts to develop the code.

In order to make up for the weak points of both a commercial tool and in-house code, an open source code such as OpenFOAM(Open Field Operation And Manipulation) has been used for researcher who are interested in fluid dynamics and aerodynamics since 1990.

1.2 What is OpenFOAM?

OpenFOAM is an open source numerical simulation software with extensive CFD and multi-physics capabilities. [6] It is first and foremost a C++ library, used primarily to create things, known as applications in the program. [7] The applications fall into two categories: solvers, that are each designed to solve a specific problem in continuum mechanics, and utilities, which are designed to perform tasks that involve data manipulation. [15] The OpenFOAM distribution contains numerous solvers and utilities covering a wide range of problems. One of the strength of OpenFOAM is that new solvers and utilities can be created by its users with some pre-requisite knowledge of the underlying method, physics, and programming techniques involved.

A central theme of the OpenFOAM design is that the solver applications have a syntax that closely resembles the partial differential equation being solved. [7] For instance, an equation 1 is represented by the code shown in below.

$$\frac{\partial \rho U}{\partial t} + \nabla \cdot \emptyset U - \nabla \cdot \mu \nabla U = -\nabla p \qquad (1)$$

By representing the equation 1 to the code, we have following types of language.

solve(fvm::ddt(rho,U)+fvm::div(phi,U)-fvm:laplacian(mu,U)==-fvc::grad(p));

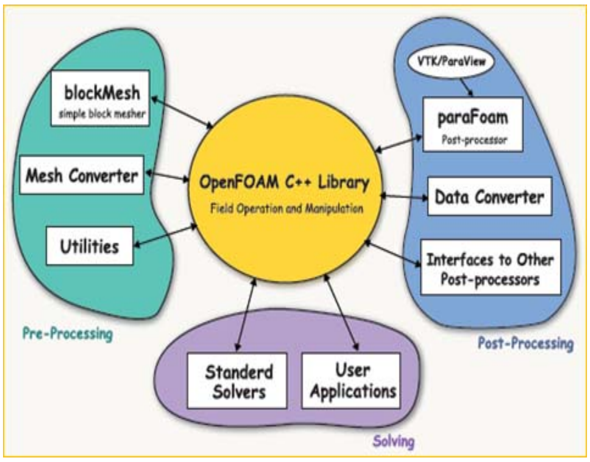

Fig. 2. Overview of OpenFOAM structure

1.3 Research objectives

In general, OpenFOAM has been used for following reasons that are quite similar with most open source code in the world: 1)Requires less memory and can be used in massively parallel computers, 2)Use the General Public License(GPL), 3)Share in-house code which means users can use it as reference as well as develop the code freely, 4)Easy to adapt to specific problems. [16]

However, OpenFOAM does not have proper manual for users as well as not provide the user interface which is well prepared in the case of commercial programs so that there are actually having little users in the community.

In addition, OpenFOAM cannot handle compressible problems since it had been especially developed for solving incompressible problems. Of course, it is well known that OpenFOAM has already some solvers such as sonicFoam which will be able to deal with compressible flows but it has not only unstable but inaccurate in the simulation since it might be still governed by pressure based code even though flow has compressible effect.

Hence, this study focused on a development and verification of density based solver using LU-SGS(Lower Upper Symmetric Gauss Seidel) algorithm in OpenFOAM. A pressure based solver in OpenFOAM for solving incompressible flow was modified to density based solver which can be applied to compressible flows. Secondly, it was not only developed implicit LU-SGS scheme instead of an explicit time integration method but implemented Riemann boundary conditions called as characteristic boundary condition which has not been developed in OpenFOAM. In addition, libraries such as wall shear stress dictionary in OpenFOAM were modified to solve and handle compressible problems. Finally, some validation models which are widely used were analyzed to validate the developed code.

2 OpenFOAM standard solvers

To begin with a development and verification of compressible codes in OpenFOAM, the authors have conducted research for validation and verification of algorithms, basic solvers, and utilities given from OpenFOAM open code.

2.1 Incompressible flow

To validate the basic solvers such as icoFoam and pimpleFoam provided by OpenFOAM for incompressible problems, a computational study of the flow fast a circular cylinder at low Reynolds number is performed numerically by solving the Navier-Stokes equations in two-dimensions.

2.1.1 Computational setup

The computational domain is assumed to be two dimensional with no variation into the page. The flow around the cylinder is discretized using a grid of 47,000 cells which is finally chosen using a grid resolution study as shown in Fig.3.

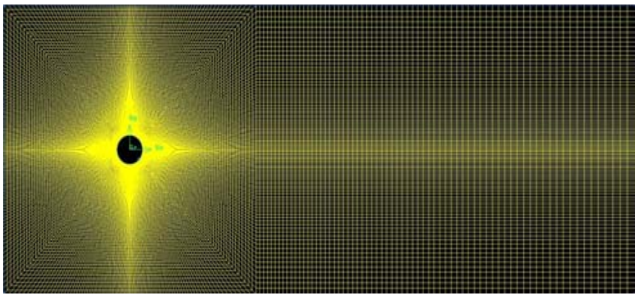

Fig. 3. Grid system for cylinder

A schematic of the flow geometry including relevant dimensions and boundaries is shown in Fig.4.

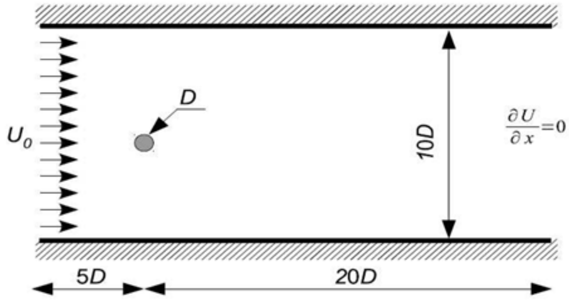

Fig. 4. Flow schematic with flow geometry

2.1.2 Flow and boundary conditions

The relevant fluid properties and boundary conditions which are used in the simulations are tabulated in Table 1 and 2. [2]

Table 1. Flow conditions for cylinder

Flow Conditions	
Viscosity	1.824E-5 Ns/m^2
Density	1.19 kg/m^3
Reynolds Number	100
Velocity	0.15 m/s
D(Characteristic Length)	0.01 m
Fluid	Air

Table 2. Numerical schemes

Numerical schemes	
Time Discretization	Implicit backward 2nd
Convective Discretization	Total Variation Diminishing
Diffusion Discretization	Central differencing scheme
Convergence criterion	1e-06

The overall accuracy of the numerical method is second order. Since a time transient analysis is performed, an initial time step of 0.00025s is chosen. The time step is validated using a time resolution study and is then used the simulation.

The total simulation time is set to 10s and the laminar model is chosen as the turbulence model.

2.1.3 Computational results

The accuracy of the numerical method is validated for natural vortex shedding case by comparing the Strouhal number from the simulation to the experimental value by Williamson. [2] The result with selected grid resolution, time resolution, and domain size parameters shows in excellent agreement with the experimental data as shown in Table 3.

Table 3. Validation on experiment

(Re = 100)	Strouhal number
Experiment	0.1643
Computation	0.1647

Fig.5. illustrates the development of the Von Karman vortex street behind a cylinder.

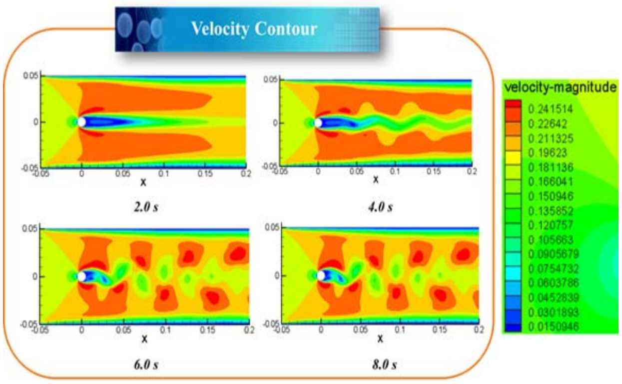

Fig. 5. Computational results of Von Karman vortex shedding

2.2 Compressible solver

To check whether standard solvers such as rhoSimpleFoam and rhoPimpleFoam provided by OpenFOAM for compressible problems are suitable for solving the compressible flow or not, the simple wedge problem with supersonic is considered.

2.2.1 Computational setup

The schematic of flow geometry of the wedge is shown in Fig.6. The computational domain is assumed to be three dimensional but z direction is almost considered as empty (OpenFOAM is basically based on only three dimensional unstructured code system.) The mesh shown in Fig.7 is used in the simulations.

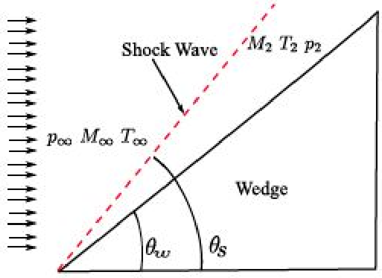

Fig. 6. The schematic with flow geometry used for wedge

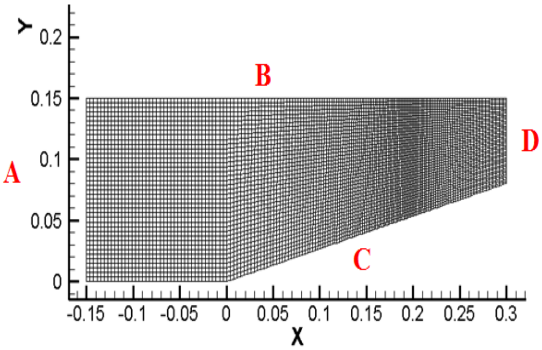

Fig. 7. Grid system for wedge

2.2.2 Flow and boundary conditions

The relevant fluid properties and boundary conditions which are used in the simulations are tabulated in Table 4 and 5.

Table 4. Boundary conditions for wedge

Boundary Conditions	
A	Fixed value (P,T,U)
B	1st order extrapolation
C	Slip wall
D	1st order extrapolation

Table 5. Flow conditions for wedge

Flow Conditions	
Viscosity	1.784E-05 Ns/m^2
Temperature	288.88 K
Pressure	101325.58 Pa
Mach number	2.5
Half angle of wedge	15 degree
Fluid	Air

2.2.3 Computational results

The simulation had been conducted in the conditions of inviscid and compressible to compare with analytical solution. The result showing pressure distribution before and after shock is shown in Fig.8.

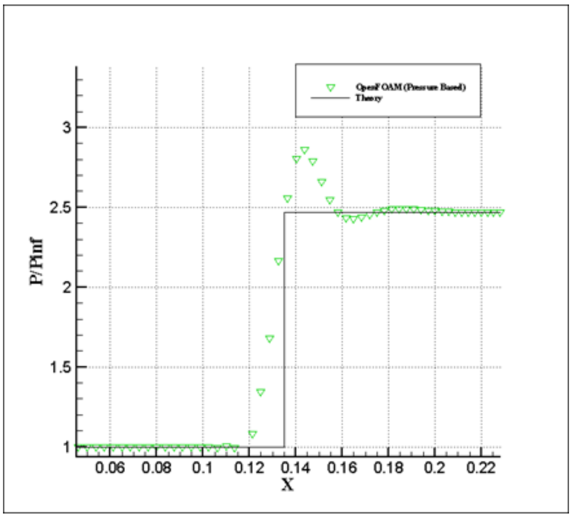

Fig. 8. Pressure distribution along the wall

As shown in Fig.8, values showing pressure distribution along the wall of the wedge after oblique shock are in a poor agreement such as oscillations when compared with analytical predictions. These results let us recognize the limitations of the solver provided by OpenFOAM for solving compressible flow. The reason why the code cannot handle compressible problems is to be developed by pressure based code even though the flow has compressible effect.

In other words, solvers that have been developed in OpenFOAM for compressible flow cannot handle compressible problems. It means it is required to develop compressible codes in OpenFOAM. That is one of the most important objectives in this paper as well.

3 Code development and validation

To overcome the limitation mentioned above section, a development and validation of density based solver was performed based on explicit solvers of Oliver Borm. [4] An outline for the development of compressible solver in OpenFOAM is shown is Fig.9.

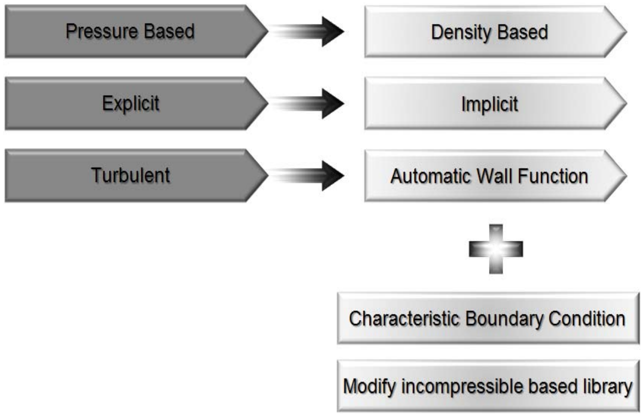

Fig. 9. Overview of code development process

3.1 Density based solver

3.1.1 Space discretization

The authors have tried to change pressure based code into density based code with reference of Oliver Borm, particularly focused on steady state. [9]

In order to handle different Riemann solvers, generic Godunov flux type class is added into OpenFOAM. For instance, two different Riemann solvers, namely the Roe with fixed entropy [10] and the AUSM+ scheme, are implemented in OpenFOAM up to now. In addition, one dimensional flux limiters like Minmod and VanLeer were rewritten as slope limiter as well as multi-dimensional Venkatakrishnan slope limiters were referred.

As a final note, the local time stepping was implemented for a fast steady state convergence.

3.1.2 Oblique shock on a 2D wedge at Mach 2.5

A computational study of oblique shock on a two dimensional wedge with supersonic was performed to validate the developed code compared with both original compressible code in OpenFOAM and analytical solutions. [12]

The grids, boundary and flow conditions are same as section from 2.2.1 to 2.2.2.

The computational results will be tabulated and shown next page in Table 6 and Fig.10. As shown in the table and figure, values with solving original compressible code are in a poor agreement but modified OpenFOAM code shows quite correct with analytic solutions.

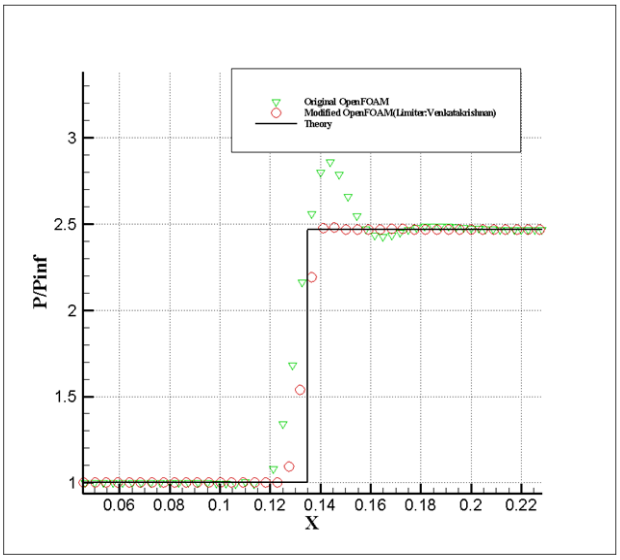

Fig. 10. Pressure distribution (Original vs. Modified code)

Table 6. Comparison between theory and computation (Modified OpenFOAM code)

Inviscid and Compressible flow over a wedge			
	Theory	OpenFOAM	Error (%)
Density ratio	1.8665	1.8641	0.12
Temp. ratio	1.3219	1.3236	0.13
Pressure ratio	2.4675	2.4655	0.08

3.2 Implicit time integration (LU-SGS)

3.2.1 Time integration

Generally speaking, the way to deal with time discretization is categorized into explicit and implicit methods at computational fluid dynamics. The explicit method has advantages of simple coding but unstable due to the restriction of Courant number. In contrast, the implicit code has drawbacks of spending much time to one iteration but very stable and efficient in convergence as well as no limitation of Courant number.

For those reasons, it is recommended for people who are getting involved in working of CFD to use implicit method as time integration if the problem is described as steady state. [23]

3.2.2 Implicit LU-SGS

In this study, implicit method of time integration called as LU-SGS scheme was used for discretization. The scheme is very popular among people who are conducting research in aerospace engineering since it is so efficient to analyze external flow with both no needs of calculation of inverse vectors and less memory.

The Thin Shear Layer(TSL) assumption is applied for LU-SGS algorithm in OpenFOAM. [5]

3.2.3 Transonic flow over a Bump in a channel

The problem called as transonic flow over a Bump in a channel was analyzed to validate the modified code with LU-SGS scheme in OpenFOAM compared with Runge-Kutta explicit method developed by Oliver Borm in OpenFOAM. [4]

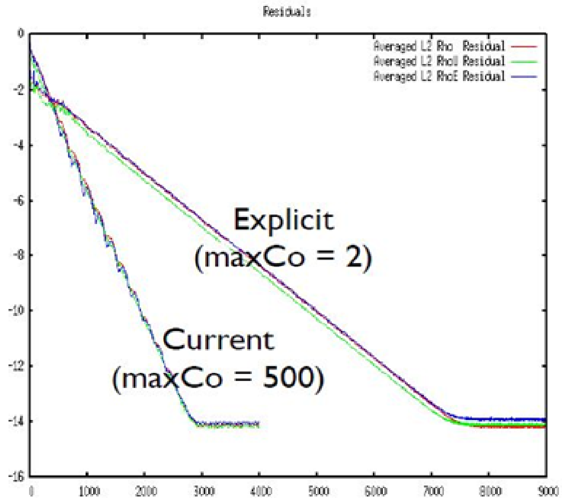

Fig. 11. Convergence history (Current = Implicit)

The simulation was performed with steady state and the results let us know the difference between explicit and implicit method in convergence history. As mentioned above of advantages of implicit method, [24] LU-SGS scheme is much faster than the scheme of explicit, namely Runge-Kutta method. It can be also regarded as having high accuracy in the simulation. [18]

3.3 Characteristic boundary condition

In general, a set of supplementary conditions such as initial and boundary condition must be provided in order to obtain a solution of numerical analysis. [23] For OpenFOAM, two types of boundary conditions were used to get a solution. [6] One is the Dirichet boundary condition, which is used when the dependent variable along the boundary is prescribed. The other is the Neumann boundary condition which is used when the normal gradient of the dependent variable along the boundary is specified. However, there are no characteristic boundaries in OpenFOM which will be useful for external aerodynamics.

Hence, the characteristic boundary condition, namely Riemann invariant, was implemented in the modified OpenFOAM code.

3.3.1 RAE 2822 airfoil

To validate modified code with Riemann invariant, aerodynamic characteristics of RAE 2822 airfoil which are widely used was analyzed with comparison of original boundary conditions such as fixed-value or free-stream condition provided by OpenFOAM. The grid system, boundary and flow conditions is tabulated and shown below.

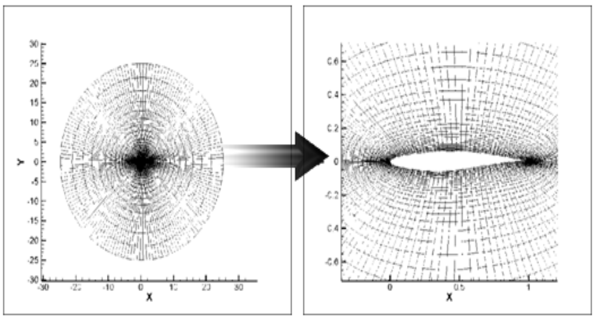

Fig. 12. Grid system for RAE-2822

Table 7. Flow condtions for RAE-2822

Flow Conditions	
Temperature	255.5 K
Pressure	108987.393 Pa
Mach number	0.729
Angle of Attack	2.31 degree
Characteristic length	1 ft

In Fig.13 and 14, it is indicated that there seems to be errors as free-stream boundary conditions is used in the computational domain reduced from 25 times of chord length to 10 times. On the other hand, there is little difference between two computational domains that used in the same way with free-stream case even though the domain is becoming small place.

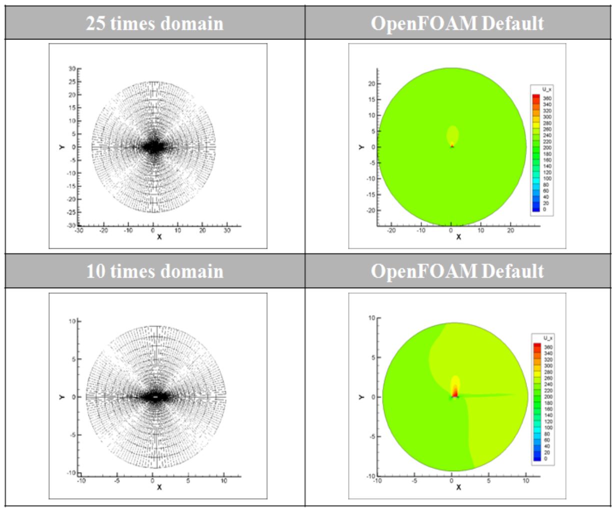

Fig. 13. OpenFOAM free-stream boundary condition

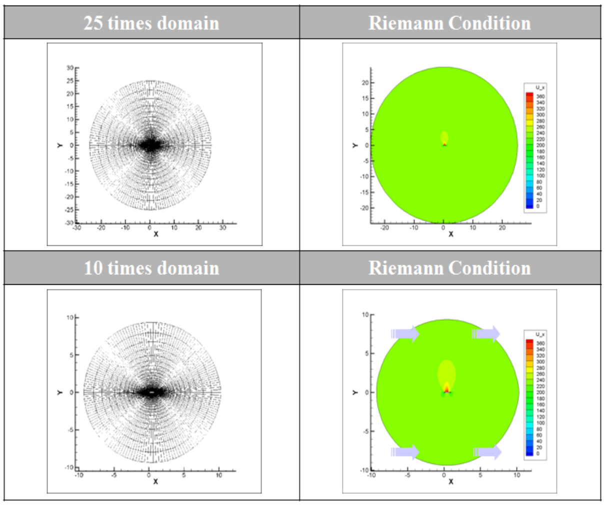

Fig. 14. Riemann boundary condition in OpenFOAM

Preliminary results showing the difference between using original boundary condition and Riemann boundary condition indicate that our setup of the characteristic boundary condition in

OpenFOAM for external aerodynamics was available.

3.4 Automatic wall function

3.4.1 Wall treatment for a turbulence model

If you have to deal with the wall which assumed to be governed by high viscous effect, it is no doubt that the grid size must be very small on the near of the wall. However, it is difficult to arrange small grids size if the configuration of models is complex so that law of the wall suggested by Von Karman is sometimes used for solving the problem.

The law of the wall is generally categorized into three parts, viscous sub-layer, log layer, and buffer layer.

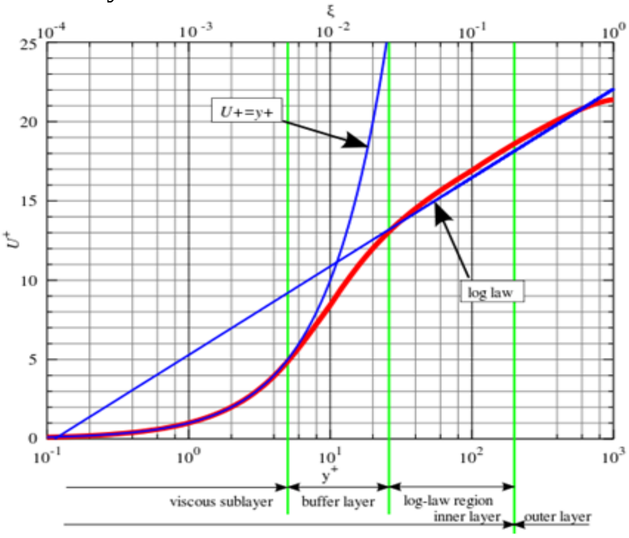

Fig. 15. Law of the wall

3.4.2 Flat plate turbulent boundary layer

Of course, OpenFOAM has already provided the option of law of the wall with turbulence models but the authors have conducted research and tried to change the code in K-Omega SST turbulence model. To validate it, a problem of flat plate with incompressible flow was performed as y+ changed.

Table 8. Grids information

Grids Information			
GRID2	y+1	GRID5	y+20
GRID3	y+5	GRID6	y+30
GRID4	y+10	GRID8	y+70
GRID9	y+90		

In figure 16 and 17, it is indicated that there is not good in accuracy in the region of buffer layer when using OpenFOAM default wall treatment. It is caused by the reason that OpenFOAM can distinguish between log layer and viscous layer but cannot handle buffer layer which is located between log and viscous layer.

On the other hand, OpenFOAM modified wall treatment, which was applying Spalding's universal equation, can have a good agreement in buffer layer as well as log and viscous layer.

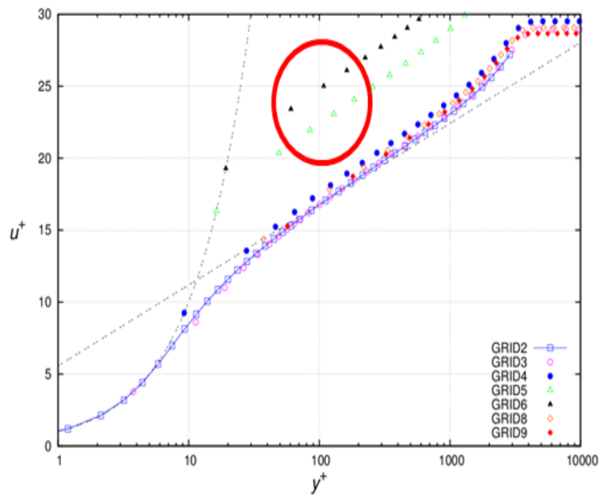

Fig. 16. OpenFOAM default wall treatment

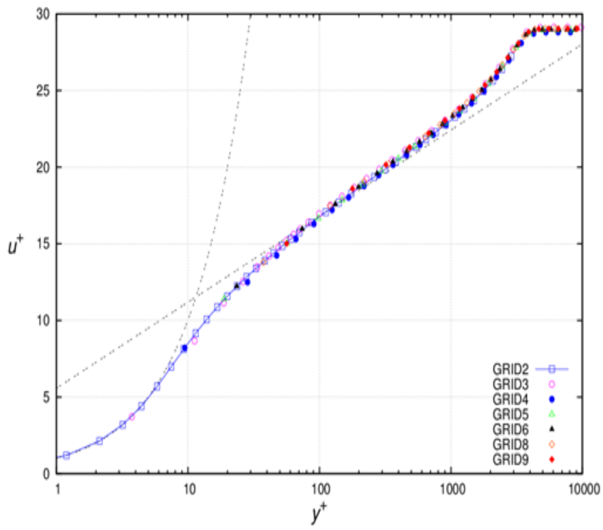

Fig. 17. OpenFOAM modified wall treatment

4 Applications

To validate the developed code, some validation models which are widely used were analyzed. The application part of this paper has the following problems: RAE-2822 transonic airfoil, three dimensional flat plate with hypersonic flow, and Shock boundary layer interaction in supersonic flow.

4.1 RAE-2822 airfoil

For aircraft speeds which are very near the speed of sound, the aircraft is called as transonic [20] and typical speeds for transonic aircraft are nearly equal to Mach one. While the aircraft itself may be traveling less than the speed of sound, the air going around the aircraft exceeds the speed of sound at some locations on the aircraft. Even though modern airliners typically fly at about M=0.85, the flow over the wing is transonic or supersonic so that it is considered as big problem in efficient operation of the aircraft.

In this study, the authors have conducted research with RAE-2822 airfoil, which is famous as transonic airfoil, to validate the developed code in transonic flow. The grids, boundary and flow conditions are same as section 3.3.1.

4.1.1 Computational results

The computational result was compared with experimental data provided by NASA. [11] Preliminary results showing between experiment and computation indicated that our setup for transonic flow is quite correct.

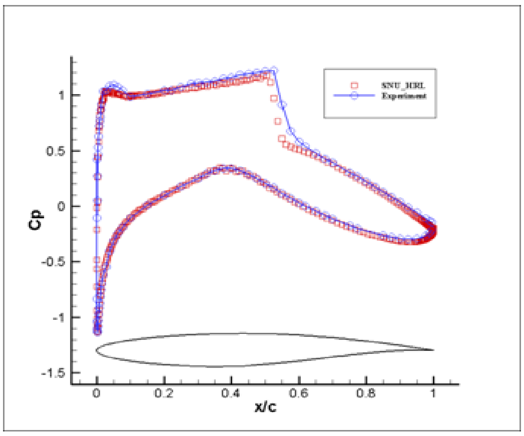

Fig. 18. Pressure coefficient on RAE-2822

4.2 Shock boundary layer interaction

A shock wave boundary layer interaction is one of the famous problems to validate supersonic flow that oblique shock has an incidence angle to the viscous layer of the wall. It is usually accompanied the phenomenon of flow separation which is due to both heat transfer increase and adverse pressure gradient. [19] In particular for hypersonic flow, it must be analyzed the phenomenon of a shock boundary layer interaction problem in case of both aerodynamic heating and fatigue of the structure from heat transfer.

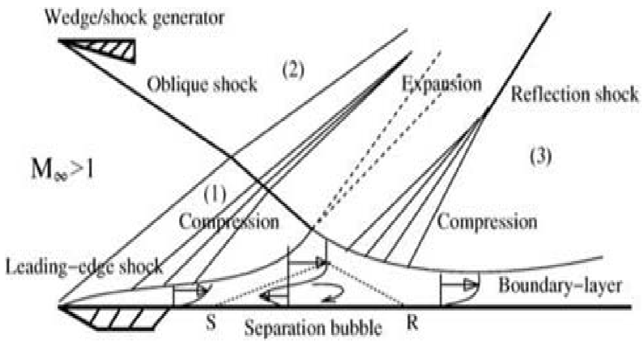

Fig. 19. Schematic of Shock B.L. interaction

4.2.1 Flow and boundary conditions

Flow conditions are same as Hakkienen's experimental conditions. The condition is tabulated in Table 9. [21]

Table 9. Flow conditions used of SWBLI

Flow Conditions	
Viscosity	1.78E-05 Ns/m^2
Temperature	288.815 K
Pressure	101325 Pa
Mach number	2
Speed of sound	340.28 m/s
Characteristic length	2
Reynolds Number	2.96x10^5
Prandtl Number	0.72
Impinging shock angle	32.585 degree
Gas	Calorically perfect gas

Grids system used in the simulation and boundary conditions are shown below.

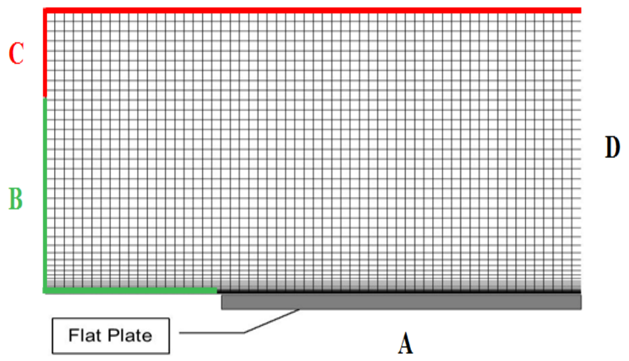

Fig. 20. Grid system used for SWBLI

Table 10. Boundary conditions used for SWBLI

Boundary Condition	
A	No slip wall
B	Inlet (1 condition)
C	Inlet (2 condition)
D	1st order extrapolation

An analytical approach [12] is used to reproduce impinging shock angle 32.585 degrees in computational domain. It should follow the fact that inlet boundary conditions have to implement differently each other.

4.2.2 Computational results

A pressure distribution along the wall is shown in Fig.21. In-house code marked in the graph is structured code developed by hypersonic and rarefied laboratory in Seoul National University and is already validated with experimental data.

As shown in the figure, it is quite good agreement between in-house code and OpenFOAM developed code but there seems to be less accurate in the region of separation point. The reason why it has less accuracy is due to the accuracy on space discretization. To be more specific, it is basically well known that in-house code based on structured code has more three order accuracy but OpenFOAM has second order accuracy at most because of the characteristic of the unstructured code.

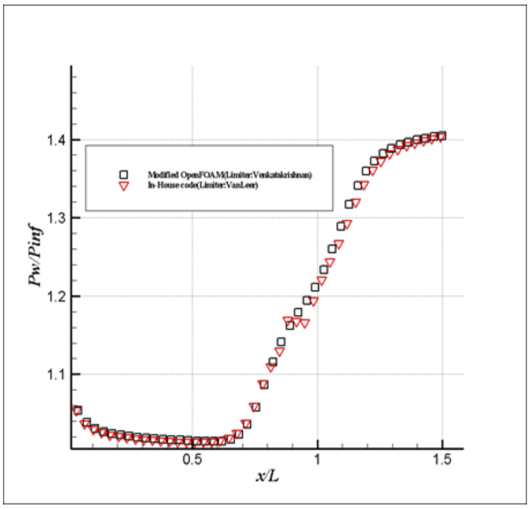

Fig. 21. In-house vs. Modified OpenFOAM (Cp)

4.3 Three-dimensional flat plate

To ensure the developed code, a computational study on three dimensional flat plate had conducted. All flow solutions were initialized by applying the free-stream conditions at altitude 15km over the entire computational domain. The schematic of flow geometry is shown in Fig.22. [13]

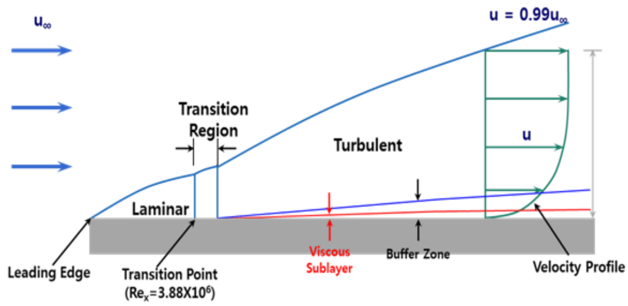

Fig. 22. Schematic of the flat plate

4.3.1 Flow and boundary conditions

The relevant fluid properties which are used in the simulations are tabulated in Table 11. Also, a grid system used in the simulation is 80x160x3 and it is simulated with x, y, and z direction for accuracy in the code.

Table 11. Flow conditions used for flat plate

Flow Conditions	
Viscosity	1.4216E-05 Ns/m^2
Temperature	216.65 K
Pressure	12111.4 Pa
Mach number	8
Speed of sound	295.07 m/s
Density	0.193919 kg/m^3

4.3.2 Computational results

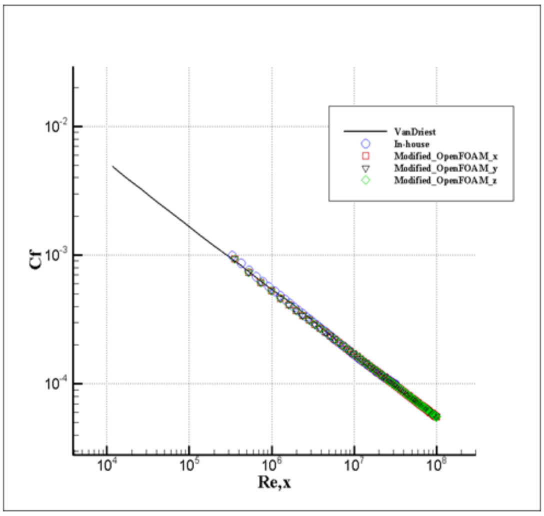

Fig. 23. Skin friction coefficient (Laminar flow, x/y/z direction)

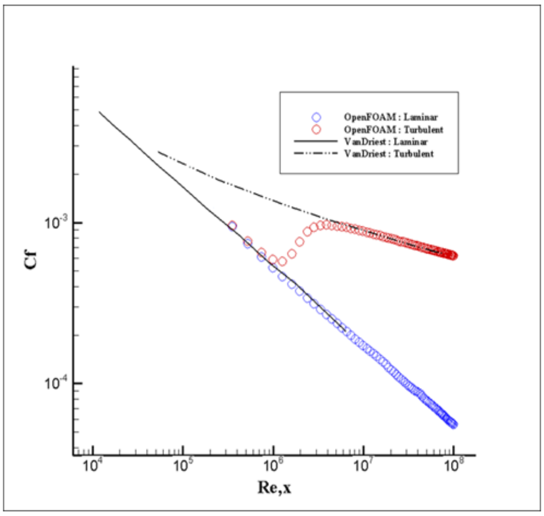

Fig. 24. Skin friction coefficient (Turbulent flow, x direction)

As shown in Fig.23 and 24, preliminary results showing skin friction coefficient along the wall between in-house code and OpenFOAM developed code are quite correct regardless of the flow is laminar or turbulent as well as even x/y/z flow direction. (Please note that the gas is assumed calorically perfect gas even though the flow is hypersonic.)

5 Conclusion

In this study, OpenFOAM which is an open source code was suggested to make up drawbacks of both in-house code and commercial program. Pressure based equations in OpenFOAM has been modified to density based equations in order to handle compressible problems. In addition, LU-SGS scheme has been implemented into the code instead of Runge-Kutta time integration which is explicit time integration method and is installed by Oliver Borm in OpenFOAM and is known as explicit time integration. Contents of the development and validation of the code have the followings:

5.1 Density based solver (Conservative form of Navier-Stokes equation)

- Pressure based governing equations were modified to density based equations in OpenFOAM for dealing with compressible flows.
- A validation of the developed code compared with solvers that has already been implemented in OpenFOAM was performed.

5.2 Implicit time integration (LU-SGS)

- LU-SGS scheme was implemented into OpenFOAM for accuracy and efficiency of the code.
- It is founded that implicit time integration like LU-SGS was much faster than explicit method in the history of convergence from the problem, namely transonic flow over a Bump in a channel.

5.3 Automatic wall function (K-Omega SST Turbulence model)

- A basic wall treatment in OpenFOAM has errors on estimation of u+ as y+ changed in the buffer layer.
- A modified wall treatment which includes Spalding's universal function has a good agreement in the estimation of u+ as y+ changed.

5.4 Characteristic boundary condition (Riemann Invariant)

- A characteristic boundary condition which is very powerful for analysis of external aerodynamics and is not ready in OpenFOAM until now has been implemented.
- RAE-2822 airfoil was chosen to validate the developed code and to compare the Riemann condition with free-stream condition that has already been provided by OpenFOAM. As a result, the characteristic boundary condition is independent on the size of domain in spite of the fact that free-stream boundary condition has errors as the domain is reduced from 25 to 10 times.

5.5 Library modification (Wall shear stress and Etc.)

- As it mentioned, OpenFOAM has basically been developed in taking aim at incompressible flow so that most of libraries are physically based on the characteristics of incompressible. This study focused on modification of original libraries to handle compressible effects such as estimating in wall shear stress using viscosity instead of dynamic viscosity.

Finally, to validate the developed code, some validation models which are widely used were analyzed. Preliminary results showing the comparison between an experiment/reference code and computation data indicated that our setup in OpenFOAM was quite correct.

The authors believe that the flexibility of the open source code will allow researchers and engineers to implement a variety of conditions and to specify the problem they want to analyze with different strategies in fluid dynamics if the developed code would be more improved in accuracy. Furthermore, it is expected that OpenFOAM based the developed code can be substituted for commercial program such as FLUENT which is costly due to the license fee.

References

[1] J.H.Kim, C.F.Lange, C.R.Koch, A computational study of a circular cylinder at low Reynolds number for open loop control of Von Karman vortex shedding, 8th International OpenFOAM conference, 2013.

[2] Williamson, C.H.K., Oblique and parallel modes of vortex shedding in the wake of a cylinder at low Reynolds number, J.Fluid Mech., 1989, p579-627.

[3] Park D.S., Hendricks, Feedback control of Von Karman vortex shedding behind a circular at low Reynolds number, Phys. Fluids, 1994, p2390-2405.

[4] Oliver Borm, Aleksandar Jemcov, Hans-Peter Kau, Density based Navier-Stokes solver for transonic flows, 6th International OpenFOAM conference, 2011.

[5] R.F.Chen, Z.J.Wang, Block lower-upper symmetric Gauss-Seidel scheme for arbitrary grids, AIAA journal, 2000.

[6] OpenFOAM User guide

[7] OpenFOAM Programmer guide

[8] Luis f. Gutierrez Marcantoni, Jose P. Tamagno, Sergio A. Elaskar, High speed flow simulation using OpenFOAM, Argentina journal, 2012.

[9] Oliver Borm, Transonic density based flow solver, 5th International OpenFOAM conference, 2010.

[10] P.L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, Journal of computational physics, 1981, p43:357-372.

[11] Catherine M. Maksymiuk, Thomas H. Pulliam, Viscous transonic airfoil workshop results using ARC2D, AIAA journal, 1987.

[12] John. D.Anderson, Fundamentals of Aerodynamics 4th, McGraw-Hill.

[13] Christopher J. Roy, Frederick G. Blottner, Methodology for turbulence model validation - Application to hypersonic flows, Journal of spacecraft and rockets, 2003.

[14] T.W.Kim, S.J.Oh, K.J.Yee, Verification of the open source code, OpenFOAM to the external flows, KSAS journal, 2011.

[15] Hrvoje Jasak, OpenFOAM: Introduction, capabilities and HPC needs, Cyprus advanced HPC workshop, 2012.
[16] J.H.Kim, K.H.Kim, A development and verification of density based implicit Navier-Stokes solver using LU-SGS algorithm in OpenFOAM, APISAT, 2013.
[17] MacCormack R.W., A numerical method for solving the equation of compressible viscous flow, AIAA journal, 1982, p1275-1281.
[18] Hoffmann, K.A., Computational fluid dynamics for engineers, engineering education system, 1989.
[19] John. D.Anderson, Hypersonic and high temperature gas dynamics, McGraw-Hill, 1989.
[20] White, F.M., Viscous fluid flow, McGraw-Hill, 1991.
[21] Hakkinen, R. J., Greber, I. Trilling, The interaction of an oblique shock wave with a laminar boundary layer, Fluid dynamic research group, M.I.T., 1957.
[22] C.F.Lange, Numerical Predictions of Heat and Momentum Transfer from a cylinder in Crossflow with Implications to Hot-Wire Anemometry, Ph.D.Thesis, 1997.
[23] John D. Anderson, Computational Fluid Dynamics, McGraw-Hill, 1995.
[24] John C. Tannehill, Computational fluid mechanics and heat transfer, Taylor&Francis, 1997.

Contact Author Email Address

mailto:sillykim@kari.re.kr

Copyright Statement