ParCFD'2020
32^{nd} International Conference on Parallel Computational Fluid Dynamics
May-17-19 2021, Nice, France

A VALIDATION FOR THE OPENFOAM-HISA SOLVER FOR DRAG PREDICTION

Saleh ABUHANIEH[†], Barış BİÇER[*] and Mehmet SAHIN[**]

[†]Atilim University
Department of Mechanical Engineering Ph.D. Program
06830 Ankara, TURKEY
e-mail: abuhanieh.sksaleh@student.atilim.edu.tr, Web page: https://www.atilim.edu.tr/en

[*] Turkish Aerospace
OpenSource CFD Research Center
Ankara, TURKEY
e-mail: baris.bicer@tai.com.tr, Web page: http://www.tusas.com

[**] Istanbul Technical University
Astronautical Engineering Department
Istanbul, TURKEY
e-mail: msahin@itu.edu.tr, Web page: https://www.itu.edu.tr/en

Key words: OpenFOAM, HiSA, DLR-F6, Drag prediction, Aerodynamics, Density-based solvers.

Abstract. Despite the popularity of OpenFOAM in many CFD applications as an open-source FVM library, it has been rarely used in the aerospace field. The same can be attributed to the absence of a coupled density-based solver which is the norm in solving such kind of problems. HiSA, is a fully-coupled density-based solver which was written mainly using the OpenFOAM libraries and can be considered as a remarkable contribution to it. In this paper, the HiSA solver's accuracy for predicting the drag is evaluated using the DLR-F6 benchmark case. This study shows that HiSA's results are good and comparable to the common solvers which are used for this application.

1 INTRODUCTION

OpenFOAM[1] can be considered as one of the most successful open-source CFD codes all over the world in both academia and industry. Except one segregated density-based solver (rhoCentralFoam), all the standard OpenFOAM solvers are segregated and pressure-based. In the foam-extend[2] branch of OpenFOAM, there are pressure-based coupled solvers, however, until now they are all incompressible solvers. In the applications where high Mach number regimes exist, the coupled density-based solvers are very common due to their capabilities in resolving the flow [1]. There were few trials like in [2] and [3] to develop a coupled density-based solvers in OpenFOAM, however, they weren't added to

[1]www.openfoam.com
[2]sourceforge.net/projects/foam-extend

the standard OpenFOAM solvers, and without proper maintenance, finally they lost their compatibility with the recent OpenFOAM versions. A serious work has been exerted in the Council for Scientific and Industrial Research (CSIR [3]) in South Africa where a new coupled density-solver has been developed. The HiSA solver [4] which stands for High Speed Aerodynamic, uses the OpenFOAM libraries and needs OpenFOAM installation to be compiled. The purpose of this work is to participate in validating this coupled solver using the well-known DLR-F6 test case with different meshes. In section 2, a brief introduction about the HiSA solver is presented. In section 3, the benchmark test case along with the used meshes are provided. The results and the conclusion are outlined in section 4 and 5 respectively.

2 HiSA

2.1 Mathematical Formulation

The conservation of mass, momentum and energy equations can be written in vector format as:

$$\frac{\partial \vec{W}}{\partial t} + \vec{\nabla} \cdot \vec{F}(\vec{W}) = \vec{Q}(\vec{W}) \tag{1}$$

Where $\vec{W}$ is the conservative variables vector which contains five components in 3D cases.

$$\vec{W} = [\rho \;\; \rho u \;\; \rho v \;\; \rho w \;\; \rho E]^T \tag{2}$$

ρ is the density. u, v and w are the fluid velocity components. E is the total energy.

The flux vector $\vec{F}(\vec{W})$ is composed of the convective flux $\vec{F_c}(\vec{W})$ and the viscous flux $\vec{F_v}(\vec{W})$ as per Eq. (3).

$$\vec{F}(\vec{W}) = \vec{F_c}(\vec{W}) - \vec{F_v}(\vec{W}) \tag{3}$$

Finally, $\vec{Q}$ is the source terms vector.

After integrating over a control volume Ω with volume V_Ω, using the divergence theorem to convert the volume integral to surface integral, and discretizing the time derivative term using the first order backward Euler (for demonstration), Eq. (1) can be written as:

$$\frac{\vec{W}_\Omega^{(n+1)} - \vec{W}_\Omega^{(n)}}{\Delta t} V_\Omega = -\vec{R}(\vec{W}^{(n+1)})_\Omega \tag{4}$$

Where $\vec{R}$ is the residual vector and it contains nonlinear terms.

$$\vec{R}(\vec{W}^{(n+1)})_\Omega = \sum_f \vec{F}(\vec{W^{(n+1)}})_f \cdot \vec{S_f} - \int_\Omega \vec{Q} d\Omega \tag{5}$$

[3] www.csir.co.za

By linearizing $\vec{R}$ and re-arranging the terms, the final format of the equation can be written as:

$$[\frac{V_\Omega \overline{I}}{\Delta t} + \frac{\partial \vec{R}(\vec{W})_\Omega}{\partial \vec{W}}|_n]\Delta \vec{W}_\Omega = -\vec{R}(\vec{W}^{(n)})_\Omega \tag{6}$$

$\overline{I}$ is the identity matrix (5x5 for each cell), and the differential term is called the Jacobian matrix. The relation in Eq. (6) represents the standard Ax = b system of equations which needs to be solved. What remains is to evaluate the right hand side (RHS) vector and the Jacobian matrix.

Ignoring the source term vector and the viscous flux, the RHS is expressed as:

$$-\vec{R}(\vec{W}^{(n)})_\Omega = -\sum_f \vec{F_c}(\vec{W^{(n)}})_f \cdot \vec{S_f} \tag{7}$$

The only remaining unknown is the convective flux at each internal face. In HiSA, two schemes are implemented to evaluate this value: HLLC [5] and AUSMPlusUp [6].

For evaluating the Jacobian matrix, HiSA uses the Lax-Friedrich approximation as explained by [7].

It's worth mentioning that, in it's initial release, HiSA was a matrix-free solver. However, in the latest releases, it's fully implicit as explained here. The same has been confirmed to the authors by the HiSA's main developers.

2.2 Code

HiSA is a library more than just a solver since it contains many new data structures and not only similar implementations for the existing ones (like implementing new boundary condition), and that because these new data structures were not needed before in the standard OpenFOAM. The "fvjMatrix" class is an example about the newly developed data structures. This class holds the assembled coupled matrix in Eq. (6), and this sort of block matrix class is not available in the standard OpenFOAM release. However, since HiSA is a coupled solver, this class is required now. Another example can be the "fluxSchemes" class which is a base class for the flux calculation methods like HLLC and AUSMPlusUp. The dual time stepping scheme is also implemented with the option of using local or global pseudo time.

3 Benchmark Test Case

For the purpose of this study, the well-known DLR-F6 case (the wing-body-nacelle-pylon model) from the 2^{nd} AIAA CFD Drag Prediction Workshop[4] has been chosen. This case is an ideal candidate since it has beside the wind tunnel results many available numerical solutions. Furthermore, the geometry is quite complex and can be considered as industry-scale case. The geometry considering half of the airplane is shown in Fig. 1. The case is transonic with Mach number of 0.75 and Reynolds number equals to 3E+6.

[4] aiaa-dpw.larc.nasa.gov/Workshop2

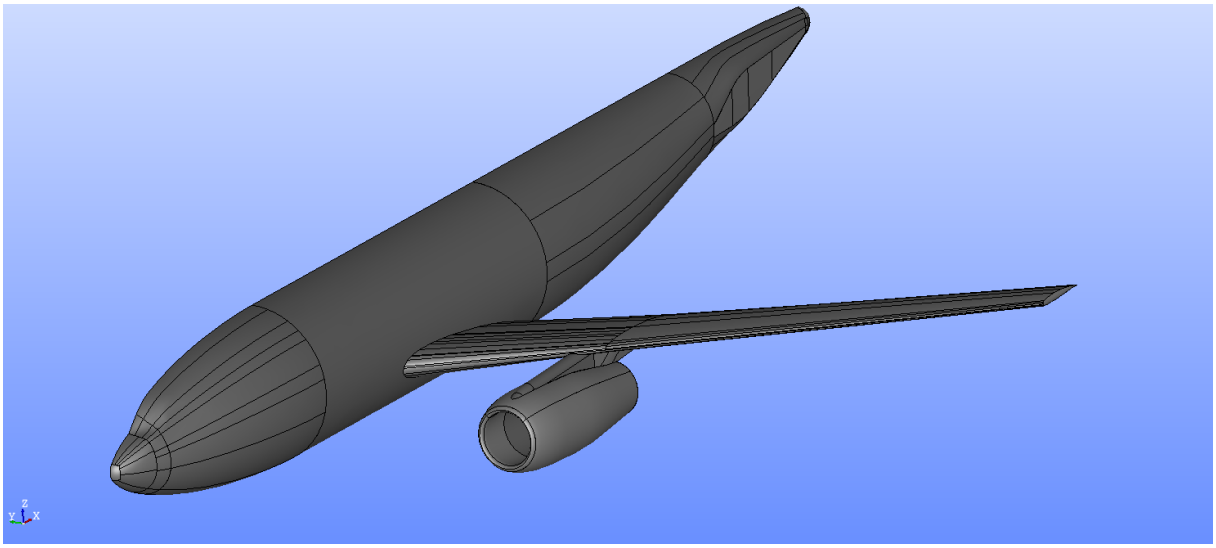


Figure 1: DLR-F6 geometry (wing-body-nacelle-pylon).

In this work, four meshes have been used/generated to produce and evaluate the results. A summary about the meshes are provided in Table. 1.

Table 1: The details of the four used meshes

Mesh Details	**Mesh A**	**Mesh B**	**Mesh C**	**Mesh D**
Mesh Type	Structured	Hybrid	Unstructured	Hybrid
Surface Mesh Size [K]	47.66	227.7	1299.5	377.63
Volume Mesh Size [M]	4.78	10.18	13.91	27.5
First Layer Thickness [μm]	1	1	100	1
Num. of Boundary Layers	25	25	5	32
Max. Non-orthogonality [°]	88.97	89.84	84.97	84.12
Max. Skewness	6.0	6.0	7.9	10.1
Max. Aspect Ratio	14694	5066	37	3620
Min. Cell Volume [m^3]	1.5E-17	1.8E-26	1.6E-13	8.4E-15

Mesh A and B are available at the workshop website as a participation by ICEM and DLR respectively. Mesh C was generated by the authors using snappyHexMesh which is a standard mesh generation tool in OpenFOAM which can produce a hex-dominant unstructured meshes. Finally, Mesh D was generated by the authors using the pentagrow tool in SUMO mesh generation toolkit which is an open-source tool [8]. The pnetagrow in SUMO extrudes the original solid body surface and create prisms/pentahedrals in the extruded region, then fills the region between the extrusion and the far-field by tetrahe-

dral cells using the TetGen library [9]. SUMO can't handle the symmetry plane, thus, the full geometry needs to be meshed.

It's worth mentioning that, the quality metrics can be calculated differently in different codes. The metrics provided in TABLE. 1 is according to OpenFOAM which considers the face values. Another point to avoid confusion is that OpenFOAM is a cell-center based finite volume code, thus, the number of computational nodes for the same mesh is different if a vertex based code to be considered.

To have more clear idea about the differences between the meshes, a slice/cut has been taken at y/b = 0.331, and the same is shown in Fig. 2.

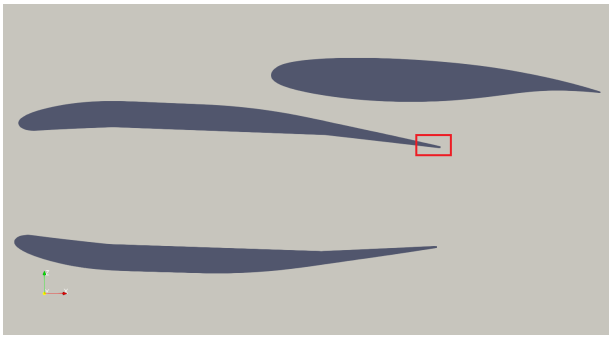


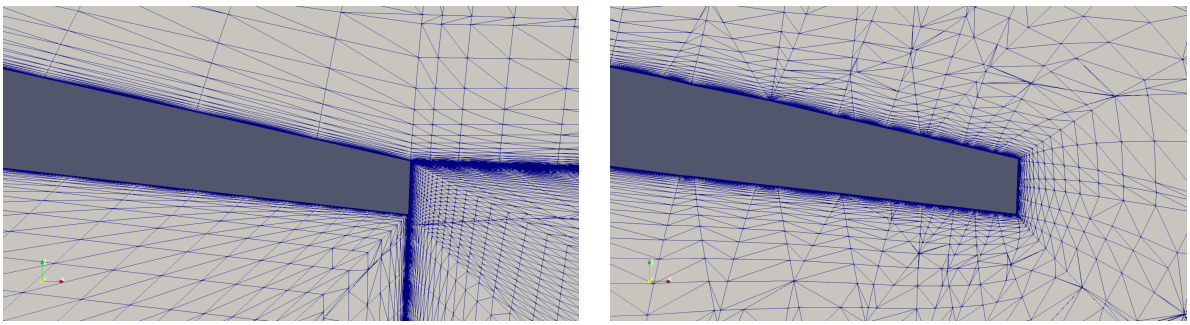
Figure 2: A slice from the DLR-F6 geometry at y/b = 0.331.

A zoomed view over the red box of Fig. 2 has been presented for the four meshes in Fig. 3 (a-d). This region has been chosen due to the difficulties associated with meshing it and specially the boundary layer adding. Thus, it's a proper region to test the capability of the used meshing tool.

As mentioned before Mesh A and Mesh B are from the workshop website, both of them have been generated using a well-known commercial codes. Thus, resolving this part of the geometry well with all the required boundary layers added isn't surprising. snappyHexMesh has a known problem of adding the boundary layers, and this is due to the fact that, snappyHexMesh doesn't start from the surface mesh (bottom-up) and doesn't preserve it. The produced SUMO mesh is a good quality mesh, and it's comparable to the commercial codes meshes. A complete 35 boundary layers were added everywhere over the aircraft. However, it can be noticed that the boundary layers extrusion direction deviates from the orthogonality more than the other meshes, and that may explain the high skewness value for this mesh in TABLE. 1.

4 Results

For the four used meshes, the characteristic boundary conditions have been used. These boundary conditions come with the HiSA library and not available (at least with the same name) in the OpenFOAM standard release. The kOmegaSST turbulence model [10] has



(a) Mesh A

(b) Mesh B

(c) Mesh C

(d) Mesh D

Figure 3: Zoomed views for the different meshes.

been used due to it's higher stability (with the OpenFOAM solvers according to the authors' experience) comparing to SpalartAllmaras model [11] . The cases have been ran as steady-state using the dual time stepping method to accelerate the convergence. Within 10K iteration, all the meshes were able to converge.

The obtained drag and lift coefficients along with the relative error percentages at Angle of Attack AoA = 1° for the four meshes are shown in Table. 2. It can be seen clearly that the results of Mesh A are better than the rest. Thus, from now on, only Mesh A will be considered.

It's worth mentioning that in the snappyHexMesh case (Mesh C), HiSA was able to predict the drag quite well although the mesh has only 5 boundary layers, however, the lift prediction was poor.

Table 2: C_d and C_l comparison with the experiment at AoA = 1°

Results	**Exp.**	**Mesh A**	**Mesh B**	**Mesh C**	**Mesh D**
C_d	0.0338	0.0335	0.0386	0.0356	0.0505
C_d Rel. Err.[%]	0.0	-0.89	14.24	5.21	49.50
C_l	0.5	0.5012	0.5191	0.6237	0.5804
C_l Rel. Err.[%]	0.0	0.23	3.81	24.74	16.07

The C_d, C_l and density residuals are presented in Fig. 4-6. It can be noticed that the lift coefficient takes more time to converge than the drag. A general observation about the residuals in HiSA is that, it's hard to go below the third order of magnitude. The residuals show convergence despite their relatively high value.

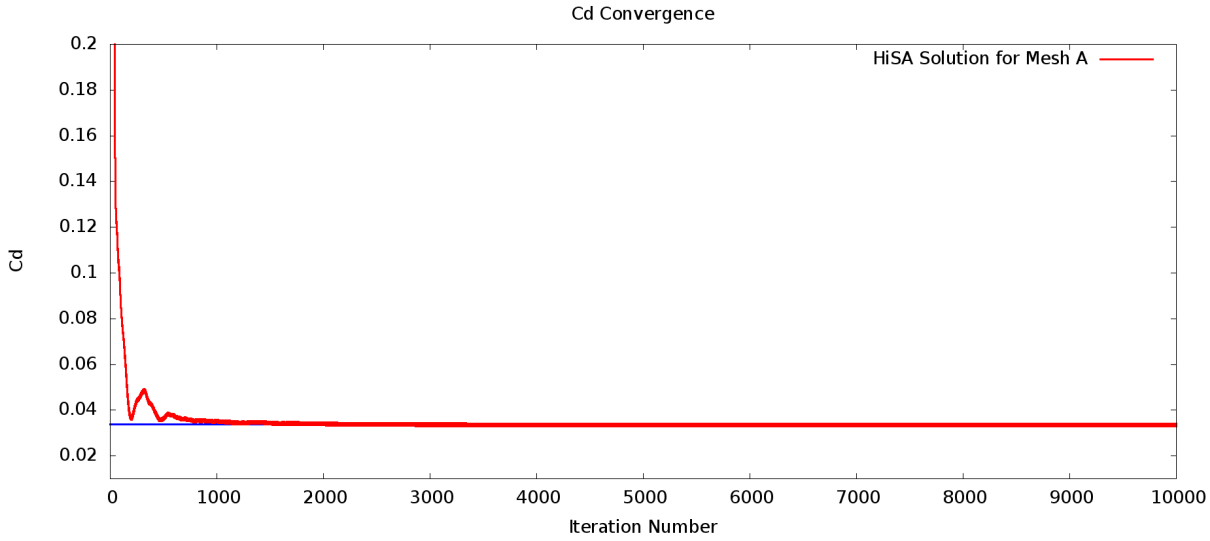


Figure 4: C_d Residual (Mesh A), the blue line represents the experimental result.

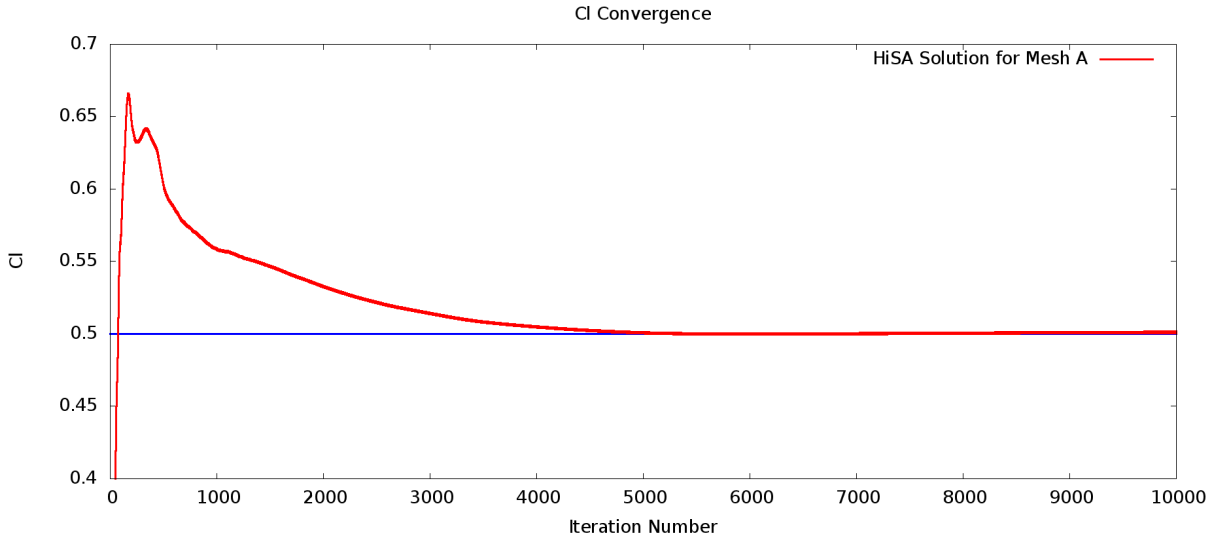


Figure 5: C_l Residual (Mesh A), the blue line represents the experimental result.

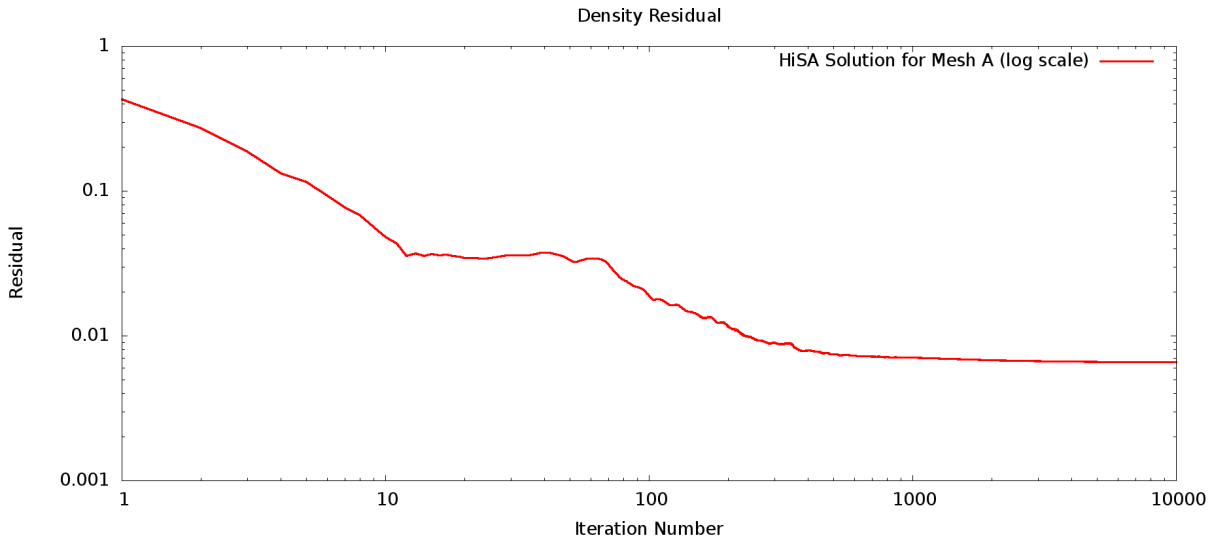


Figure 6: Density Residual (Mesh A).

To ensure the reliability of the obtained results using Mesh A, a finer version of the same mesh (the medium mesh) has been tried to test the mesh convergence. The medium mesh contains 8.29M cells and that's why it takes more number of iteration to converge (around 15K iteration). The C_d and C_l convergence comparisons between the coarse and medium versions of Mesh A are plotted in Fig. 7 and Fig. 8 respectively. Knowing that for this geometry (wing-body-nacelle-pylon) the mesh convergence was not always achieved by the participants of the mentioned drag workshop.

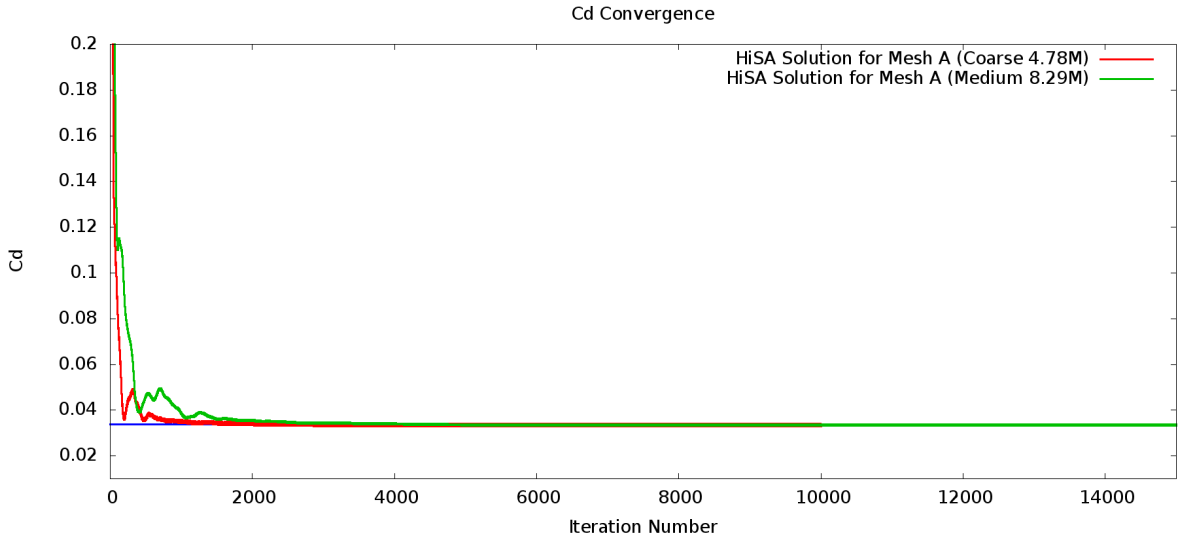


Figure 7: Mesh convergence test using C_d (Mesh A), the blue line represents the experimental result.

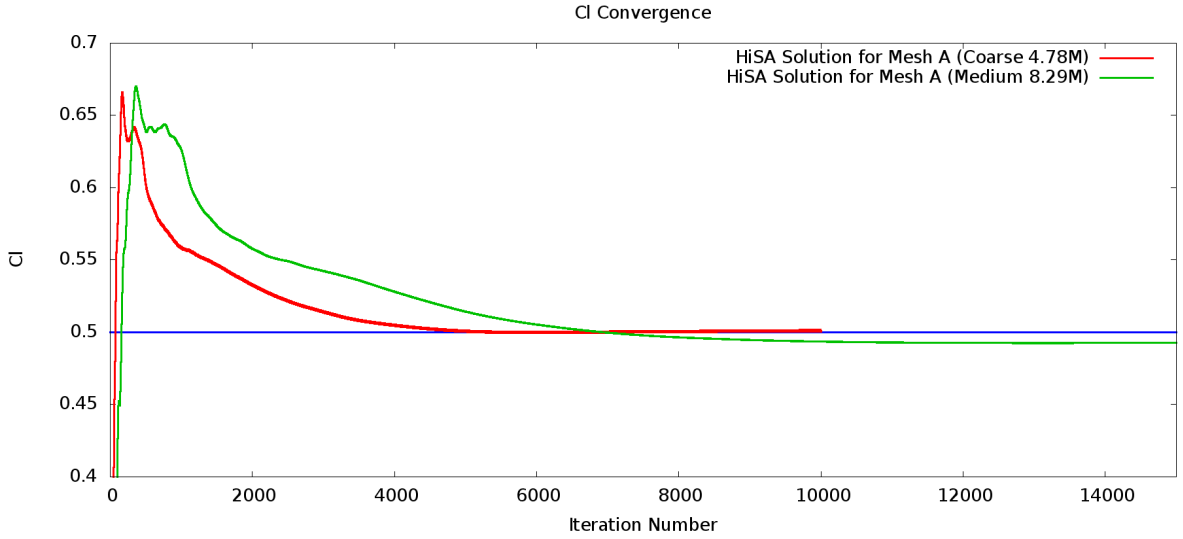


Figure 8: Mesh convergence test using C_l (Mesh A), the blue line represents the experimental result.

The pressure coefficients (C_p) at different y/b sections are shown Fig. 9-11. The AIRBUS results using the elsA code have been added to the comparison since they used the same structured mesh, however, they used the fine version of Mesh A to produce their results. Furthermore, these results from AIRBUS are the only C_p results provided by the participants of this workshop using a structured mesh.

At the first cross section (y/b = 0.15) in Fig. 9 which is the closest to the fuselage, the AIRBUS results are closer to the experiment at the trailing edge and at the middle of the suction side of the wing. The same can be attributed to the finer mesh used. At the other points, HiSA solution is either better or comparable.

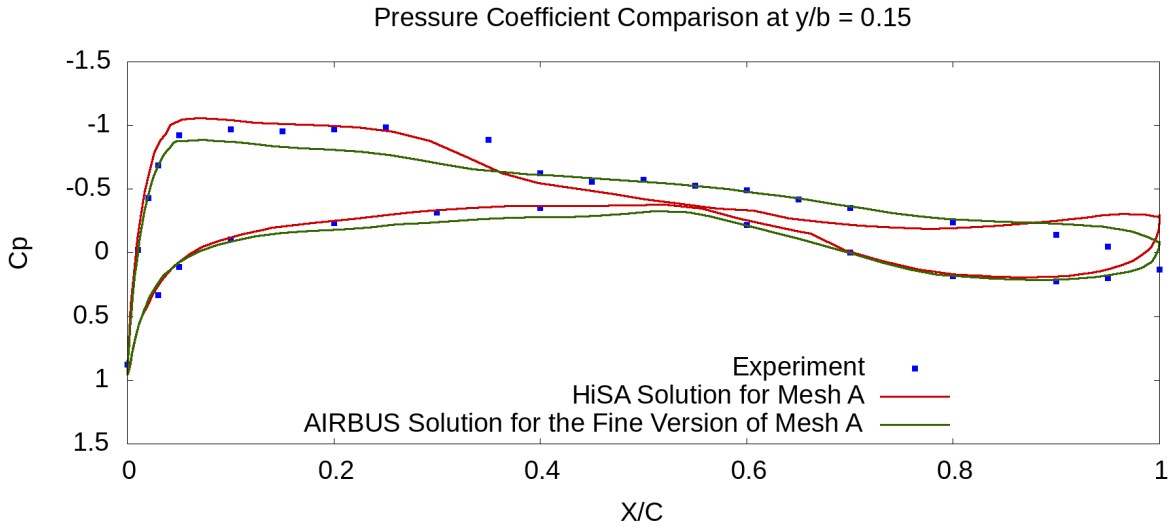


Figure 9: C_p plot at y/b = 0.15.

At the second cross section (y/b = 0.331) in Fig. 10, the AIRBUS results are quite better at the lower (high pressure) side of the wing near the trailing edge, whereas HiSA performs better at the suction side.

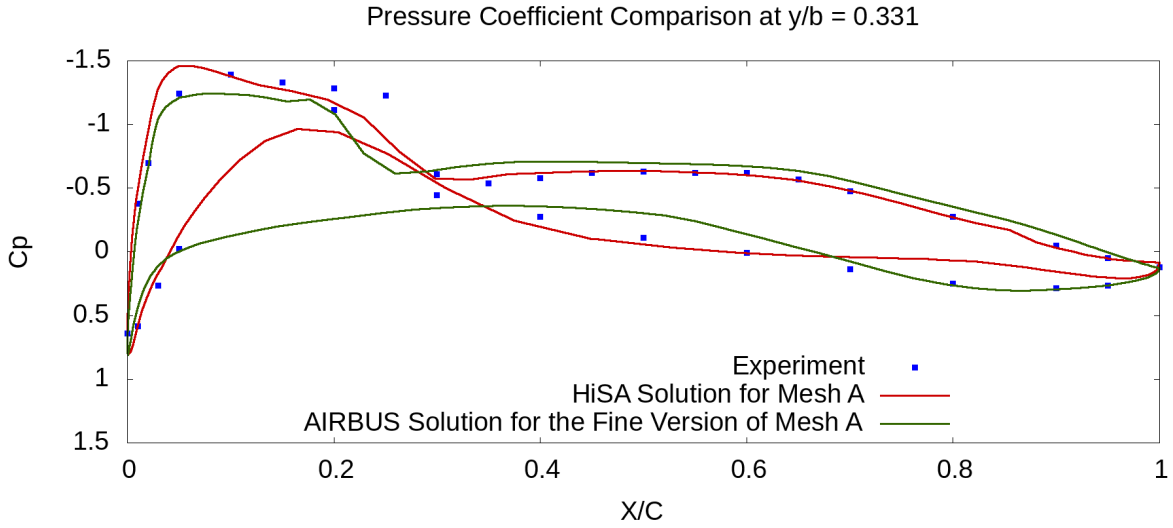


Figure 10: C_p plot at y/b = 0.331.

At the closest section to the tip of the wing (y/b = 0.847) in Fig. 11, HiSA results are quite better near the leading edge and very close to the AIRBUS results at the remaining points.

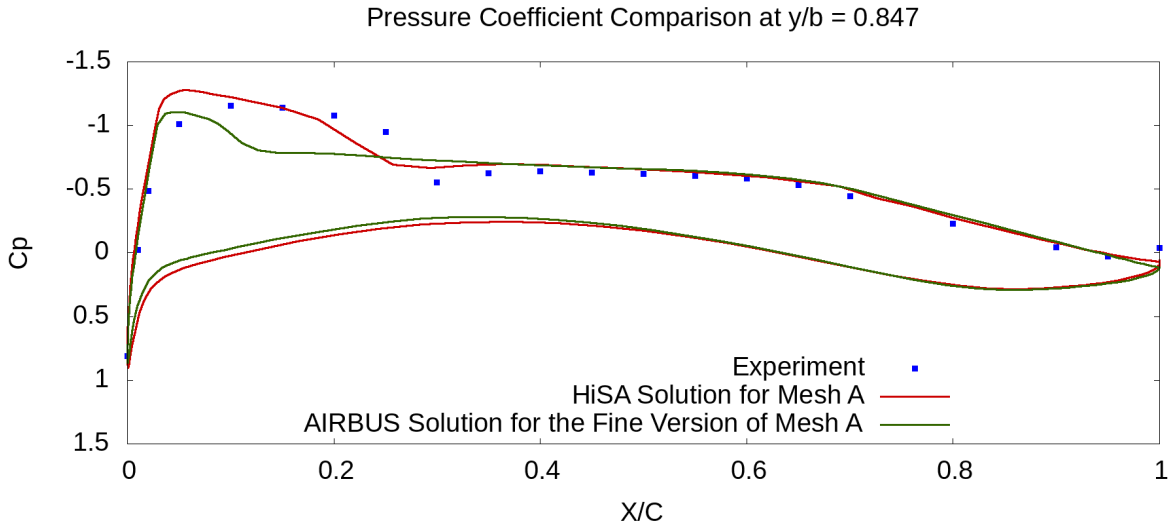


Figure 11: C_p plot at y/b = 0.847.

Finally, the C_l versus C_d at different Angle of Attacks (Drag Polar) have been plotted in Fig. 12. Again the AIRBUS results have been included for comparison, however, this time they used the medium version of mesh A to produce the results.

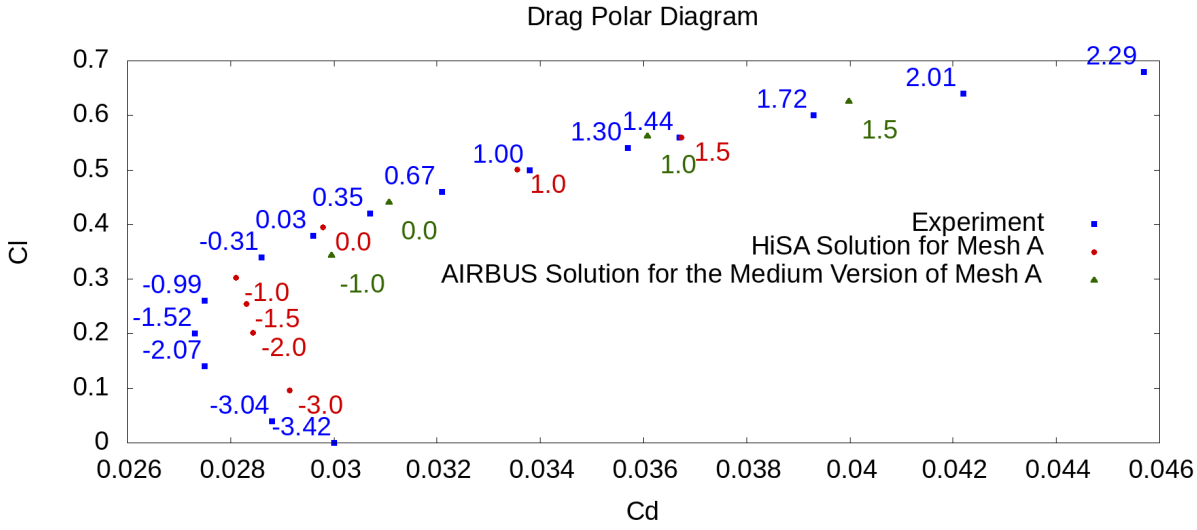


Figure 12: Drag polar diagram comparison.

It can be seen from Fig. 12 that HiSA solver results are better than the provided results by AIRBUS. However, the HiSA results get worse with comparison to the experiment as the AoAs decrease to negative values. The same has been observed by some participants in the workshop, see for example the results obtained by the OVERFLOW code.

5 Conclusion

- The obtained results by HiSA using Mesh A are very good and comparable to the well-known codes used by the other participants of this workshop
- The mesh convergence was achieved which is a good sign about the solver
- With Mesh B, a better results were expected, however, they aren't so far from the experiment
- The results of Mesh C weren't so accurate specially for the lift prediction
- Despite a very good mesh was obtained using SUMO (Mesh D), however, it's results were poor. That can be attributed to the non-orthogonal extrusion direction of the boundary layers starting from the surface mesh.
- This solver is an important contribution towards increasing the usability of OpenFOAM in the high speed flows applications, however, more validations are required

REFERENCES

[1] J. Blazek, *Computational Fluid Dynamics: Principles and Applications*, Elsevier Science, 2 ed., 2015.

[2] C. Shen, F. Sun, and X. Xia, *Implementation of density-based solver for all speeds in the framework of openfoam*, Computer Physics Communications, 185 (2014), pp. 2730–2741.

[3] J. Kim and K. Kim, *A development and verification of density based solver using lu-sgs algorithm in openfoam*, in 29th Congress of the International Council of the Aeronautical Science, St. Petersburg, Russia, September 2014.

[4] J. Heyns, O. Oxtoby, and A. Steenkamp, *Modelling high-speed flow using a matrix-free coupled solver*, in 9th OpenFOAM Workshop, Zagreb, Croatia, June 2014.

[5] P. Batten, M. Leschziner, and U. Goldberg, *Average-state jacobians and implicit methods for compressible viscous and turbulent flows*, Journal of Computational Physics, 137 (1997), pp. 38–78.

[6] M. Liou, *A sequel to ausm, part ii: Ausm+-up for all speeds*, Journal of Computational Physics, 214 (2006), pp. 137–170.

[7] H. Luo, J. D. Baum, and R. Lohner, *A fast, matrix-free implicit method for compressible flows on unstructured grids*, Journal of Computational Physics, 146 (1998), pp. 664–690.

[8] M. Tomac and D. Eller, *Towards automated hybrid-prismatic mesh generation*, Procedia Engineering, 82 (2014), pp. 377–389.

[9] H. Si, *Tetgen, a delaunay-based quality tetrahedral mesh generator*, ACM Transactions on Mathematical Software, 41 (2015).

[10] F. Menter, M. Kuntz, , and R. Langtry, *Ten years of industrial experience with the sst turbulence model*, in In Proceedings of the fourth international symposium on turbulence, heat and mass transfer, Antalya, Turkey, January 2003.

[11] P. Spalart and S. Allmaras, *A one-equation turbulence model for aerodynamic flows*, AIAA, 439 (1992).