Application of Diablo to Three-Dimensional Benchmark Problems for Reynolds-Averaged Navier-Stokes Solvers

Thomas A. Reist* and David W. Zingg†

Institute for Aerospace Studies, University of Toronto, Toronto, Ontario, M3H 5T6, Canada

This paper presents the application of a Newton-Krylov-Schur solver for the Reynolds-averaged Navier-Stokes equations to two turbulent flow benchmark problems. The first is the subsonic flow about a three-dimensional hemispherical-capped cylinder for angles of attack between 0° and 19°. Comparisons are made with both experimental data as well as the results of other CFD codes. For large angles of attack, significant differences exist between the computational and experimental results, particularly in the resolution of the suction peak. However, the computed aerodynamic coefficients agree well with those of other CFD codes. The second benchmark problem, the ONERA M6 wing, is considered at two angles of attack. At an angle of attack of 3.06° the computed pressure distributions agree well with experimental results, with the same discrepancies widely noted in the literature observed. The aerodynamic coefficients agree well with those computed by other codes. At the more challenging angle of attack of 6.06°, an unsteady analysis is performed due to the presence of highly separated flow. Solver performance is characterized by presenting accuracy and CPU time as a function of grid density.

I. Introduction

This paper presents the application of a Newton-Krylov-Schur solver for the Reynolds-averaged Navier-Stokes (RANS) equations, called Diablo, to the two benchmark problems of the AIAA's 2018 Solver Technology Discussion Group. The problems under consideration are the three-dimensional hemispherical cylinder (H3D) and ONERA M6 (OM6) wing, as described on NASA's Turbulence Modelling Resource (TMR) website.[1] In addition, for the OM6 wing, the effect of grid topology is investigated. This builds on the AIAA's 2016 Solver Technology Discussion Group which focused on two-dimensional benchmark problems.[2]

The solution accuracy and solver performance will be assessed. Solution accuracy will be characterized by examining the grid convergence behaviour of functionals, namely lift, drag, and pitching moment, in addition to the resolution of flow characteristics such as shocks, vortices, and separated flow. Where available on the TMR website, comparison with experimental data and the results of other CFD codes will also be made. Solver performance will be characterized using the proposed metrics presented in Brown and Zingg,[3] who studied the convergence behaviour of Diablo for the simulation of the TMR's benchmark NACA0012 airfoil problem, as well as the OM6 and NASA's Common Research Model wing. They showed good agreement between Diablo and other solvers for the NACA0012 problem, and presented solver performance, including functional accuracy and CPU time, as a function of grid size.

II. Solution Methodology

The solver, known as Diablo, uses a parallel implicit Newton-Krylov-Schur algorithm to solve the Euler[4] and RANS[5] equations with the Spalart-Allmaras turbulence model.[6] A finite-difference spatial discretization is used, which is implemented using summation-by-parts operators of various orders and simultaneous approximation terms for weak enforcement of boundary and block interface conditions.[7–10] This solver has been

*Research Associate, tom.reist@utoronto.ca

†University of Toronto Distinguished Professor of Computational Aerodynamics and Sustainable Aviation, Director, Centre for Research in Sustainable Aviation, and Associate Fellow AIAA, dwz@oddjob.utias.utoronto.ca

validated extensively using two and three-dimensional validation cases from the TMR, as well as through participation in the AIAA's 5$^{\text{th}}$ Drag Prediction Workshop.[11] This section will highlight some elements of the solution methodology which will be referred to throughout this paper as they relate to solver convergence behaviour. Further details can be found in Del Rey Fernández et al.,[8] Hicken and Zingg,[4] and Osusky and Zingg.[5]

The governing equations to be solved are the RANS equations in three dimensions. In conservative form in Cartesian coordinates, they take the form

$$\frac{\partial \boldsymbol{Q}}{\partial t} + \frac{\partial \boldsymbol{E}}{\partial x} + \frac{\partial \boldsymbol{F}}{\partial y} + \frac{\partial \boldsymbol{G}}{\partial z} = \frac{\partial \boldsymbol{E_v}}{\partial x} + \frac{\partial \boldsymbol{E_v}}{\partial y} + \frac{\partial \boldsymbol{G_v}}{\partial z} \tag{1}$$

where $\boldsymbol{Q}$ is the 5-vector of conserved quantities (mass, momentum, and energy), $\boldsymbol{E}$, $\boldsymbol{F}$, and $\boldsymbol{G}$ are the inviscid fluxes, and $\boldsymbol{E_v}$, $\boldsymbol{F_v}$, and $\boldsymbol{G_v}$ are the viscous fluxes.[12] The variables are non-dimensionalized by a characteristic length and freestream quantities to bring their magnitudes close to unity. The viscous fluxes in Equation 1 contain terms due to turbulence which cannot be evaluated. Thus, the negative variant of the one-equation Spalart-Allmaras turbulence model (SA-neg)[6] is used to solve for the turbulent viscosity. This introduces a sixth PDE and solution variable, which is included in the discrete solution vector. While the Spalart-Allmaras model contains a trip function to account for laminar-turbulent transition, it is not used in this work and fully turbulent flow is assumed. The grid is decomposed into multiple blocks to facilitate parallel computation.

The introduction of the turbulence model and the discretization of the governing equations leads to a large system of nonlinear ordinary differential equations of the form

$$\frac{\partial \boldsymbol{\mathcal{Q}}}{\partial t} + \boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{Q}}) = 0 \tag{2}$$

where $\boldsymbol{\mathcal{Q}}$ and $\boldsymbol{\mathcal{R}}$ are the discrete solution vector and flow residual for the domain, respectively. In this work the steady solution which satisfies $\boldsymbol{\mathcal{R}}(\boldsymbol{\mathcal{Q}}) = 0$ is sought.

Equation 2 is solved using pseudo-transient continuation, which involves time-marching with the implicit Euler method. This requires the solution of, at time-step n,

$$\left(\boldsymbol{\mathcal{T}}^{(n)} + \boldsymbol{\mathcal{A}}^{(n)}\right) \Delta \boldsymbol{\mathcal{Q}}^{(n)} = -\boldsymbol{\mathcal{R}}^{(n)} \tag{3}$$

where $\boldsymbol{\mathcal{T}}$ is a diagonal matrix of the inverse of the locally varying time-steps, $\boldsymbol{\mathcal{A}}$ is the flow Jacobian $\frac{\partial \boldsymbol{\mathcal{R}}}{\partial \boldsymbol{\mathcal{Q}}}$, and $\Delta \boldsymbol{\mathcal{Q}}^{(n)} = \boldsymbol{\mathcal{Q}}^{(n+1)} - \boldsymbol{\mathcal{Q}}^{(n)}$ is the solution update. For an infinite time-step, i.e. $\boldsymbol{\mathcal{T}} = \boldsymbol{0}$, this reduces to Newton's method. Equation 3 is solved in two phases: 1) a start-up phase which aims to provide a good starting point for Newton's method, and 2) an inexact-Newton phase which drives the solution to steady-state. The difference between the two phases is the manner in which the time-step is ramped up, how the flow Jacobian is formed, and how accurately the linear problem is solved at each time-step.

In the start-up phase, the local time-step at each node is calculated for each time-step, n, as

$$\Delta t^{(n)} \propto ab^n \tag{4}$$

where $a = 10^{-3}$ and $b \in [1.05, 1.30]$. Smaller values for b give a more gradual ramp up of the timestep which can aid in difficult problems. Unless otherwise stated, $b = 1.30$ is used for the computations performed here. The time step $\Delta t^{(n)}$ is also a function of the metric Jacobian resulting from the coordinate transformation. In the start-up phase, the flow Jacobian is also replaced by a first-order approximation which: 1) increases the second-difference dissipation and removes the fourth-difference dissipation for the inviscid terms, 2) neglects the cross-derivative terms in the viscous stresses, and 3) the viscosity values in the viscous fluxes are treated as constant. These simplifications increase the sparsity of the flow-Jacobian.

In the inexact-Newton phase, the time-step is increased more quickly via

$$\Delta t^{(n)} \propto \max\left[\alpha \left(\frac{||\boldsymbol{\mathcal{R}}^{(n)}||_2}{||\boldsymbol{\mathcal{R}}^{0}||_2}\right)^{-\beta}, \Delta t^{(n-1)}\right] \tag{5}$$

where $\beta = 1.5$ for this work, and

$$\alpha = \Delta t^{(n_{\text{switch}})} \left(\frac{||\boldsymbol{\mathcal{R}}^{(n_{\text{switch}})}||_2}{||\boldsymbol{\mathcal{R}}^{0}||_2}\right)^{\beta} \tag{6}$$

where n_{switch} is the time-step at which the algorithm switches from the start-up to the inexact-Newton phase. In the inexact-Newton phase, the full Jacobian can either be computed explicitly or, since a Krylov subspace method is used for solving the linear problem and hence only Jacobian-vector products are required, these Jacobian-vector products can be approximated via a first-order Frechet derivative based upon the flow residual vector. The algorithm switches from the start-up to the inexact phase when the relative residual has been reduced by 3 orders of magnitude, unless otherwise stated.

For both the start-up and inexact-Newton phases, the linear problem arising at each time-step is solved using the generalized minimum residual (GMRES) method, together with approximate-Schur parallel preconditioning with ILU(p) factorization. The linear problem is solved to a relative tolerance, η, such that

$$\left|\left|\boldsymbol{\mathcal{R}}^{(n)} + \left(\boldsymbol{\mathcal{T}}^{(n)} + \boldsymbol{\mathcal{A}}^{(n)}\right)\Delta\boldsymbol{\mathcal{Q}}^{(n)}\right|\right|_2 \leq \eta \left|\left|\boldsymbol{\mathcal{R}}^{(n)}\right|\right|_2 . \tag{7}$$

A tolerance of $\eta = [0.01, 0.05]$ is used for the start-up phase, and this tolerance is tightened to $\eta = [0.001, 0.01]$ for the inexact phase. Unless stated otherwise, $\eta = 0.05$ and $\eta = 0.01$ are used for the start-up and inexact phases, respectively. During start-up, a fill level of 2 is used for the ILU factorization, which increases to 3 for the inexact phase. The most relevant algorithm parameters have been presented here. For a more detailed algorithm description together with a list of default parameters, see Reference 13. Diablo uses both the scalar dissipation scheme of Jameson et al.[14] and the matrix dissipation scheme of Swanson and Turkel.[15] In this paper, scalar dissipation will be used unless stated otherwise.

III. Performance Metrics

All computations are performed on the SciNet General Purpose Cluster in Toronto, Ontario, Canada. This cluster consists of 3,780 nodes, each with 8 Intel Xeon E5540 processors at 2.53GHz and 16GB of RAM. Nodes are connected with 5:1 blocking/oversubscribed QDR. Solver performance will be tracked on the basis of both solution accuracy and solver speed. The details of the respective metrics are described below.

A. Computational Cost

To help remove hardware dependencies, solver time is reported in TauBench units[a]. The TauBench benchmark case used comprises 2.5×10^5 grid nodes run on 1 processor with 10 steps. The average TauBench time for the above problem after 5 runs on SciNet is 8.188 seconds; this is defined as one TauBench work unit, wu, in this paper. CPU time (in TauBench units) is reported throughout this paper, and is calculated as the wall time required to reduce the L^2-norm of the residual below a certain tolerance, multiplied by the number of processors used. Solver performance is also reported in terms of equivalent residual evaluations, which is the total CPU time divided by the average time for one residual evaluation. This helps to remove hardware dependence and to make comparisons between different algorithms. Since Diablo is an implicit solver, the number of Krylov iterations is also presented, as this provides a hardware-independent metric of problem difficulty and cost.

B. Solver Accuracy

The accuracy of the CFD solver is examined based on comparison with experimental results and with other CFD codes. For the H3D and OM6 cases, pressure distributions are available from the TMR website. Since differences may exist between the experimental and CFD setup, e.g. the existence of the splitter plate at the root of the OM6, comparison with other codes is important. For comparison with other codes, only the aerodynamic functionals are available from the TMR website. These will be presented in this paper as a function of grid size to establish how the different codes behave as a function of grid size and whether they approach the same value with infinite refinement.

With a family of consistent grids, the grid-converged functionals, i.e. those in the limit of an infinitely fine grid, can be estimated using Richardson extrapolation,[16] which can only be applied if one is in the asymptotic region and if the convergence is monotonic.

From the three finest members of a grid family, the order of convergence, p, can be calculated as

$$p = \frac{1}{\ln r} \ln\left(\frac{|f_3 - f_2|}{|f_2 - f_1|}\right) \tag{8}$$

[a] http://www.ipacs-benchmark.org

Table 1: H3D grid family properties. The grid is partitioned into 368 blocks.

Level	Nodes	Avg. off-wall spacing	Avg. y^+
L0	82,800,000	2.93×10^{-5}	0.388
L1	11,200,000	5.90×10^{-5}	0.787
L2	1,620,000	1.19×10^{-4}	1.60
L3	270,000	2.45×10^{-4}	2.93

where f_1 is the functional on the finest grid, and f_3 is the functional value on the coarsest of the three. The refinement ratio, r, is the ratio of the grid spacings between successive grid levels. From this, the grid-converged estimate, f_0, can be obtained through

$$f_0 \approx f_1 + \frac{f_1 - f_2}{r^p - 1} \tag{9}$$

For the results presented here, if the computed order of convergence, p, is less than unity or the functionals on the finest grid levels are non-monotonic, first order extrapolation, $p = 1$, is performed, as in Brown and Zingg.[3]

IV. Three-Dimensional Hemispherical Cylinder

The first benchmark case is a cylinder capped with a hemisphere. The dimensions are as specified on the TMR website.[1] The grids provided for the workshop contain a polar singularity along the cylinder's axis at the tip of the hemisphere. Grids with this feature are currently not supported by Diablo. Thus, custom grids are created which aim to closely replicate the workshop grids. These grids feature a 'cap' at the tip of the hemisphere where the polar singularity would exist. The grid resolution on the surface is very similar to that in the workshop grids, and pertinent spacings, including off-wall, far-field, hemisphere-cylinder interface, outflow, and farfield are preserved, as are the mesh growth distributions. The four-member grid family is described in Table 1. For reference, the workshop L0-level grid has approximately 79.4×10^6 nodes. The grid sizes deviated slightly form the workshop grids due to topology differences[b], requirements imposed by domain decomposition, and minimum block sizes on the coarsest grids. From the finest L0-level grid, subsequent members are created by removing every other node, i.e. $r = 2$ in Equations 8 and 9. The grid is partitioned into 368 blocks for parallel computation. Each member of the grid family is shown in Figure 1, in which the hemisphere tip cap is evident.

The flow conditions for all of the H3D cases are at a Mach number of 0.60 and a Reynolds number of 0.35×10^6. The surface is an adiabatic no-slip surface, with Riemann extrapolation at the far-field, and a pressure outflow boundary where the outflow pressure is specified as the freestream pressure and the remaining quantities are extrapolated from the interior.[c] Computations are performed at angles of attack $\alpha = 0°$, 5°, 10°, 15°, and 19°. A steady-state solution is considered to be found when the L^2-norm of the residual is reduced by twelve orders of magnitude. Due to the increased flow complexity at the higher values of α, more robust solver parameters are used than the defaults described in Section II. At 15° and 19°, the algorithm switches between the start-up and inexact phases after reducing the residual four orders of magnitude, instead of the three orders required for the lower angles of attack. At $\alpha = 19°$, the time-step ramping parameter is decreased to $b = 1.25$ from $b = 1.30$. In the interest of clarity and brevity, only the results for $\alpha = 0°$, 10°, and 19° will be discussed in detail.

Comparisons are made between the computed (on the finest grid) and experimentally determined pressure distributions at various circumferential angles, ϕ, around the cylinder, and are shown in Figure 2, where $\phi = 0°$ is the top, i.e. leeward, side of the cylinder, and $\phi = 180°$ is the bottom, i.e. windward, side. A discrepancy between the computational and experimental results is evident near the suction peak. In general, the magnitude of the suction peak is greater in the computational results than in the experiment, and occurs upstream of the experimental peak.[d] Mayeur et al.[17] performed the same computations and

[b]There are two coincident nodes at block interfaces

[c]These boundary conditions are enforced using the SAT approach.

[d]However, due to the sparsity of the experimental data, the location of the experimental suction peak cannot be precisely determined.

Table 2: Forces and moments on the H3D. Entries with $p = 1$ have non-monotonic convergence or $p < 1$.

α	C_L			C_D			C_M		
	L0	Extrap.	p	L0	Extrap.	p	L0	Extrap.	p
0°	0.000	0.000	1.29	0.0112	0.0111	1.77	0.000	0.000	1
5°	0.0150	0.0149	1	0.0122	0.0121	1.72	−0.0023	−0.0022	1
10°	0.0344	0.0343	1	0.0161	0.0159	1.54	−0.0074	−0.0072	1
15°	0.0611	0.0607	1.12	0.0242	0.0240	1.40	−0.0168	−0.0163	1
19°	0.0874	0.0865	1.08	0.0351	0.0346	1.28	−0.0269	−0.0259	1.01

present results for $\alpha = 0°$, which exhibit the same behaviour. At $\alpha = 19°$, a region of separated flow is evident in the experimental results for $\phi \leq 60°$ and approximately $x \geq 4\,\text{in}$. This is not quite captured by the CFD solution. For the $\alpha = 19°$ case, the computed flow pattern is visualized in Figure 3, where a separated vortex is evident. The primary vortex forms between $1\,\text{in} \leq x \leq 4\,\text{in}$, and lifts off of the body by $x = 5\,\text{in}$ where a secondary near-body vortex forms. By $x = 7\,\text{in}$, three vortices are evident.

To characterize the resolution of the flow with grid refinement, pressure coefficient, C_p, and streamwise skin friction, $C_{f,x}$, are shown in Figure 4 at the symmetry plane as computed on each grid level. Over the length of the cylinder, resolution of $C_{f,x}$ is relatively insensitive to grid refinement up to $\alpha = 10°$, with the exception of the coarsest grid. Beyond this, particularly at $\alpha = 19°$, significant differences arise for $x \geq 4\,\text{in}$. Figure 5 shows details of the suction peak. The L0 and L1-level grids show a nearly indistinguishable difference in both C_p and $C_{f,x}$. The coarsest grid does a particularly poor job at predicting the correct $C_{f,x}$. Discontinuities are also evident, most noticeably in $C_{f,x}$, on the coarsest grid. These are due to the presence of the SATs at the block boundaries. As the grid is refined, the magnitude of these discontinuities approaches zero.

The convergence behaviour with grid refinement is shown in Figure 6, together with the validation data from the TMR website. The dashed horizontal lines represent the Richardson extrapolated values for each solver. The values on the finest grid and the extrapolated values are tabulated in Table 2. In general, Diablo agrees well with the results of the other codes. All of the codes agree quite well for C_D, while larger spreads exist for C_L and C_M, and this spread increases with increasing angle of attack. The functionals computed with Diablo are, in general, of slightly larger magnitude than the other solvers, with the greatest difference being in C_L and C_M. At the higher angles of attack, the error on the coarsest grids is larger than that of the other solvers. This error on coarse grids can be reduced through the use of matrix dissipation, as shown for Diablo in Osusky and Zingg;[5] this will be further discussed in Section V.

The solver performance is shown in Figure 7. For $\alpha = 15°$ and 19°, a solution was not attainable on the L0 grid when partitioned into 368 blocks. When partitioned into 1334 blocks a solution was obtained and is presented here. For large grids with a small number of block interface nodes, the CPU time difference due to different partitionings is relatively small and can thus be ignored. Performance is given by several metrics. First, the error as a function of grid size is shown, along with the CPU time vs. the functional error. Second, the total CPU time and the CPU time per grid node is given. As in Brown and Zingg,[3] a line of best fit for CPU time per grid node is included. In the case of perfect scaling, the slope of this line should be zero. However, in general the problem becomes more challenging with increasing grid size, such that a small positive slope is usually observed. On average, Diablo requires approximately 8×10^{-3} work units per grid node on this hardware.

Finally, to decouple solution time from hardware performance, the number of equivalent residual evaluations and Krylov iterations is given. Equivalent residual evaluations are counted as the total CPU time divided by the average CPU time for one residual evaluation. For this problem, Diablo requires 5,000-20,000 equivalent residual evaluations to converge, with most cases converging within 10,000. Tracking Krylov iterations is a good metric for the changes in problem difficulty with grid size, as it is independent of any time metric. As would be expected, the finest grids can require up to 5 times as many Krylov iterations. On the coarse grids, deep convergence is obtained quickly once the algorithm transitions to the inexact phase, typically within a few Newton iterations. However, the linear solves performed during the inexact phase still account for over 60% of the Krylov iterations. As the grid is refined, a larger number of Newton iterations are required during the inexact phase. This fact, together with the fact that the linear problem at each

Table 3: OM6 grid properties. Grid is partitioned into 748 blocks.

Level	Nodes	Avg. off-wall spacing	Avg. y^+
L0	73,634,000	1.3×10^{-6}	0.507
L1	9,818,000	2.6×10^{-6}	1.08
L2	1,391,000	5.5×10^{-6}	2.30
L3	220,000	1.2×10^{-5}	4.03

Newton iteration in the inexact phase is more computationally demanding due to the tighter Krylov solver tolerance, lack of simplifying approximations in the flow Jacobian, and the large pseudo-time step, gives rise to the significant increase in the number of Krylov iterations with grid refinement.

V. ONERA M6 Wing

As with the H3D grid, the workshop grids contain a polar singularity for the OM6. Thus, grids are created which have the same topology as the H3D grid in Section IV and mimic the spacings and growth ratios of the workshop OM6 grids. A family of grids is created by removing every other node from the finest grid, L0. The properties of these grids are given in Table 3, and they are shown in Figure 8. The grid sizes deviated slightly from the workshop grids due to topology differences[e], requirements imposed by domain decomposition, and minimum block sizes on the coarsest grids, but the near-body resolution remains the same as the workshop grids.

The test conditions for the OM6 are at a Mach number of 0.84 and a Reynolds number, based on the root chord, of 14.6×10^6. Two angles of attack are considered, 3.06° and 6.06°. The case at 3.06° has a double shock, but the flow is largely attached, while it is known that the 6.06° case exhibits significant turbulent boundary-layer separation.[18] This paper will examine both cases in terms of both solver performance and accuracy, grid resolution, as well as the effect of grid topology. All default algorithm parameters are used, with the exception that the time-ramping parameter is decreased to $b = 1.15$, and the tolerance at which to switch between the two phases is reduced to 10^{-4}. A steady-state solution is considered to be found when the L^2-norm of the residual is reduced by eleven orders of magnitude.

A. Angle of Attack = 3.06°

The accuracy of the solver is assessed by examining 1) agreement of pressure distributions with experimental data, 2) impact of grid refinement on flow resolution, and 3) comparison of CFD functionals with other codes. Figure 9 shows the pressure distributions compared with experimental data. In general, the CFD results agree well with the experimental data. As with some other computational studies[5,19] differences exist in the amount of lift carried at the mid-chord on the inboard station, as well as the resolution of the double shock at 80% span. The impact of grid resolution on the C_p and $C_{f,x}$ distributions at 20%, 65%, and 96% span is shown in Figure 10. The two finest grid levels, L0 and L1, resolve the primary flow features, including the suction peak and shocks, with the L1-level grid slightly smoothing out the shock and underpredicting the drop in $C_{f,x}$ after the shock. The two coarsest grid levels, L2 and L3, are insufficient to resolve the shocks, and the L3-level grid is unable to properly predict $C_{f,x}$.

Figure 11 compares the convergence of functionals with grid refinement with other CFD results provided on the TMR website. The functionals computed by Diablo on the finest grid and the extrapolated values are tabulated in Table 4. As demonstrated for Diablo in previous work by Osusky and Zingg,[5] the use of scalar dissipation increases functional error, particularly on coarse grids. Thus, results computed using matrix dissipation are included in Figure 11, where it can be seen that the use of matrix dissipation reduces the functional error by up to almost 50% on the coarsest grids. In some cases, the use of matrix dissipation makes the solution algorithm less robust, resulting in non-physical flow quantities or divergence of the solution. This can require the use of more conservative algorithm parameters, such as the use of $b = 1.05$ on the L0-level grid.

Solver performance is shown in Figure 12. The results presented use scalar dissipation. The solution

[e]There are two coincident nodes at block interfaces

Table 4: OM6 at $\alpha = 3.06°$: Forces and moments computed using scalar dissipation.

α	C_L			C_D			C_M		
	L0	Extrap.	p	L0	Extrap.	p	L0	Extrap.	p
3.06°	0.2716	0.2725	2.058	0.01713	0.01700	2.087	−0.1925	−0.1931	2.210

on the finest grid had difficulty converging to the required tolerance, resulting in increased cost as seen in the figure. After transitioning to the inexact phase, the algorithm was not able to drive the residual to the required tolerance of 10^{-11}. The data point shown in Figure 12 for the L0-level grid is that for which the residual of the mean flow equations has been reduced to the required tolerance, while the residual of the turbulence equation has been reduced by 6 orders of magnitude. At this depth of convergence, the functionals are accurate to more than 8 digits. On the L1 to L3-level grids, Diablo converges in approximately 1,000-2,500 Krylov iterations. On the coarsest grid, approximately 50% of the Krylov iterations take place in the start-up phase, while on the L1-level grid 65% occur in the start-up phase. A larger proportion of Krylov iterations are required in the start-up phase compared to the H3D cases. This is due to the use of the smaller residual reduction tolerance at which the algorithm transitions between phases. In general, for more challenging problems, a smaller switching tolerance is required to ensure a suitable initial iterate for Newton's method is found.

1. Alternate Grid Topology

The TMR test case for the OM6 specifies the use of a sharp trailing edge (TE). While this feature *can* be meshed using an O-topology as above, it lends itself well to an H-topology grid. The use of an H-topology grid for a sharp TE can both simplify gridding and avoid the highly skewed cells present when an O-topology is adapted for use with the sharp TE. However, an H-topology is inefficient in terms of the nodal distribution, particularly as it goes to the far-field. To help remedy this, a hybrid HO-topology is used. This uses an H-topology near the body, which is in turn surrounded by an O-topology blocking. A schematic of the HO-topology is shown in Figure 13. To investigate the impact of this alternate topology on the OM6 configuration, an HO-topology grid is created which has the same surface resolution (number of nodes and spacings) as the grid used previously. The finest grid level has 156×10^6 nodes. While this grid has the name near-body resolution as the workshop grids, by nature of the H-topology, this results in more nodes in the domain extending to the far-field, particularly emanating fore and aft along the wake plane, and hence a larger overall grid. A family of grids is created by removing every other node.

Solutions are computed on the this grid family using scalar dissipation. A converged solution was not attained on the L0-level grid, caused by divergence of the turbulence residual. For the remaining grid levels, the aerodynamic functionals are shown in Figure 14 along with those computed on the O-topology grid from previously (i.e. that which mimics the workshop grid.) Both results use scalar dissipation. The HO-topology grid possesses slightly more error on the coarser grids for a given grid size, particularly in C_L and C_M, with good agreement in C_D. The extrapolated values computed from the results on both topologies are within 1% of each other. However, as noted above, the HO-topology requires more grid nodes in the domain for a given surface resolution. So, if the curves for the HO-topology results in Figure 14 were to be shifted horizontally to the right to match the O-topology curves on a surface node basis, the HO-topology grids would produce the same C_L and C_M as the O-topology grid, while the error in C_D, most notably $C_{D,p}$, would be almost 50% lower. The pressure distributions computed on the O and HO-topology grids are compared in Figure 15. The solid black line is the solution on the L0-level O-topology grid. From this figure, it can be seen that on the coarser grids, particularly L2 and L3, the HO-topology grid better captures the suction peak. This is likely a result of the off-body clustering at the LE, a region critical to drag prediction, due to the nature of the HO-topology. Details of the LE grid are shown in Figure 16. Since the boundary-layer resolving blocks must extend to the far-field, the LE and TE off-body regions are naturally better resolved with this topology. This does however, also imply the increased grid size as discussed above.

While the HO-topology grid offers more accurate functionals for a given surface resolution, this is obtained at increased computational cost for two reasons. First, the inefficient nature of the node distribution for the HO-topology results in the need for more computational resources (for fixed nodes per CPU). Second, at least in the cases presented here, the algorithm has difficulty reducing the turbulence residual. In the

results presented above, the mean flow equations have been solved to within the specified tolerance, while the turbulence equation cannot be solved to within a relative tolerance less than 10^{-7}. Regardless of the turbulence equation, the problem algorithm has more difficulty solving on the HO-topology grids. If the mean flow residual reduction is taken as the convergence criterion, then the HO-topology grid requires up to 60% more Krylov iterations to converge.

B. Angle of Attack = 6.06°

The OM6 at 6.06° presents a particularly challenging case due to the strong shocks and resulting separation. Indeed, Diablo is unable to obtain a steady solution on all but the coarsest grid. To investigate this case further, an unsteady analysis is performed on the L1-level grid. Diablo has been extended for time-accurate integration by Boom and Zingg.[9] Here, a 2^{nd}-order backwards differencing time-marching method is used with a non-dimensional time-step of $\Delta t = 5 \times 10^{-2}$. The unsteady solution is run for 125 time units, the history of which is shown in Figure 17 in terms of C_L and C_D. After the initial flow has developed, periodic behaviour develops after approximately $t = 40$. Three full periods are shown in the figure. Over the last two periods, the time-averaged C_L and C_D are 0.521 and 0.05869, respectively. The solution at four separate instances is shown in Figure 18. The first and third instances correspond to the first and fourth crests in the periodic portion of the unsteady history, and the second and fourth instances correspond to the first and fourth troughs. It is evident that the flow on the inboard portion of the wing is steady, and it is only the outboard 25% which varies in time. Over this outboard portion, the chordwise location of the tip shock varies with time, as does the size and position of the resulting separated flow.

VI. Conclusions

This paper presents the application of a Newton-Krylov-Schur solver for the Reynolds-averaged Navier-Stokes equations to two turbulent flow benchmark problems. The first is the subsonic flow about a three-dimensional hemispherical-capped cylinder for angles of attack between 0° and 19°. Comparisons are made with both experimental data as well as the results of other CFD codes. For large angles of attack, significant differences exist between the computational and experimental results, particularly in the resolution of the suction peak. In general, agreement with experimental data was better on the windward side of the cylinder than that on the leeward side. However, the computed aerodynamic coefficients agree well with those of other CFD codes. The second benchmark problem, the ONERA M6 wing, was considered at two angles of attack; 3.06° and 6.06°. At an angle of attack of 3.06°, the computed pressure distributions agree well with experimental results, with the same discrepancies at the inboard and shock locations as widely noted in the literature observed. The computed aerodynamic coefficients agree well with those computed via other codes. At the more challenging angle of attack of 6.06°, a steady solution is only obtained on the coarsest grids considered. These results support the conclusion, as has been reported, of the existence of highly separated flow on the outboard wing, leading to an unsteady flow. Unsteady analysis was performed on this case and a region of unsteady flow was identified towards the wing tip, with movement in the shock location and recirculation region identified. The inboard flow was found to be steady.

Acknowledgments

Computations were performed on the General Purpose Cluster supercomputer at the SciNet HPC Consortium. SciNet is funded by: the Canada Foundation for Innovation under the auspices of Compute Canada; the Government of Ontario; Ontario Research Fund - Research Excellence; and the University of Toronto.

References

[1] "Turbulence Modelling Resource," `http://turbmodels.larc.nasa.gov`, NASA Langley Research Center, Langley, VA.

[2] Diskin, B. and Thomas, J. L., "Introduction: Evaluation of RANS Solvers on Benchmark Aerodynamic Flows," *AIAA Journal*, Vol. 54, No. 9, 2016, pp. 2561–2562.

[3] Brown, D. A. and Zingg, D. W., "Performance of a Newton-Krylov-Schur Algorithm for the Numerical Solution of the Steady Reynolds-Averaged Navier-Stokes Equations," *AIAA Journal*, Vol. 54, No. 9, 2016, pp. 2645–2658.

[4] Hicken, J. E. and Zingg, D. W., "A Parallel Newton-Krylov Solver for the Euler Equations Discretized Using Simultaneous Approximation Terms," *AIAA Journal*, Vol. 46, No. 11, 2008, pp. 2773–2786.

[5]Osusky, M. and Zingg, D. W., "A Parallel Newton-Krylov-Schur Flow Solver for the Navier-Stokes Equations Discretized Using Summation-By-Parts Operators," *AIAA Journal*, Vol. 51, No. 12, 2013, pp. 2833–2851.

[6]Spalart, P. R. and Allmaras, S. R., "One-Equation Turbulence Model for Aerodynamic Flows," *30th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA-0092-0439, Reno, NV, 1992.

[7]Svard, M. and Nordstrom, J., "Review of Summation-by-Parts Schemes for Initial-Boundary-Value-Problems," *Journal of Computational Physics*, Vol. 268, No. 1, 2014, pp. 17–38.

[8]Del Rey Fernández, D. C., Hicken, J. E., and Zingg, D. W., "Review of Summation-by-Parts Operators with Simultaneous Approximation Terms for the Numerical Solution of Partial Differential Equations," *Journal of Computers and Fluids*, Vol. 95, 2014, pp. 171–196.

[9]Boom, P. D. and Zingg, D. W., "Time-Accurate Flow Simulations Using an Efficient ewton-Krylov-Schur Approach with High-Order emporal and Spatial Discretization," *51st AIAA Aerospace Science Meeting*, AIAA-2013-0383, Grapevine, TX, 2013.

[10]Dias, S. C. and Zingg, D. W., "A High-Order Parallel Newton-Krylov Flow Solver for the Euler Equations," *19th AIAA Computational Fluid Dynamics Conference*, AIAA-2009-3657, San Antonio, TX, 2009.

[11]Osusky, M., Boom, P. D., and Zingg, D. W., "Results from the Fifth AIAA Drag Prediction Workshop obtained with a parallel Newton-Krylov-Schur flow solver discretized using summation-by-parts operators," *31st AIAA Applied Aerodynamics Conference*, AIAA-2013-2511, San Diego, CA, June 2013.

[12]Pulliam, T. H. and Zingg, D. W., *Fundamental Algorithms in Computational Fluid Dynamics*, Springer-Verlag, 2014.

[13]Osusky, M., *A Parallel Newton-Krylov-Schur Algorithm for the Reynolds-Averaged Navier-Stokes Equations*, Ph.D. thesis, University of Toronto, 2013.

[14]Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes," *14th Fluid and Plasma Dynamics Conference*, AIAA-1981-1259, Palo Alto, CA, 1981.

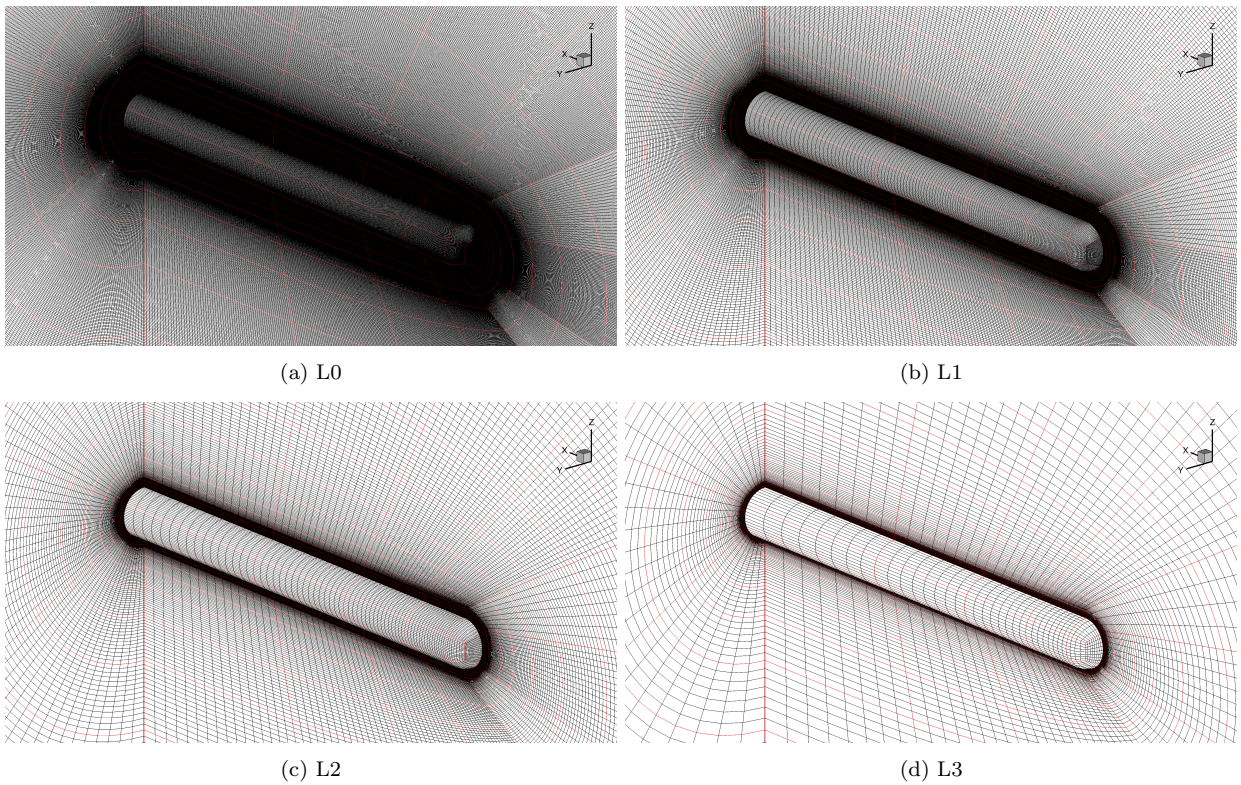
[15]Swanson, R. C. and Turkel, E., "On central-Difference and Upwind Schemes," *Journal of Computational Physics*, Vol. 101, No. 2, 1992, pp. 292–306.

[16]Roache, P. J., *Verification and Validation in Computational Science and Engineering*, Hermosa Publishers, Albuquerque, NM, 1998.

[17]Mayeur, J., Dumont, A., Destarac, D., and Gleize, V., "RANS simulations on TMR 3D test cases with the Onera elsA flow solver," *54th AIAA Aerospace Sciences Meeting*, AIAA-2016-1357, San Diego, CA, 2016.

[18]Schmitt, V. and Charpin, F., "Pressure Distributions on the ONERA-M6 Wing at Transonic Mach Numbers," Tech. Rep. AGARD AR 138, ONERA, 1979.

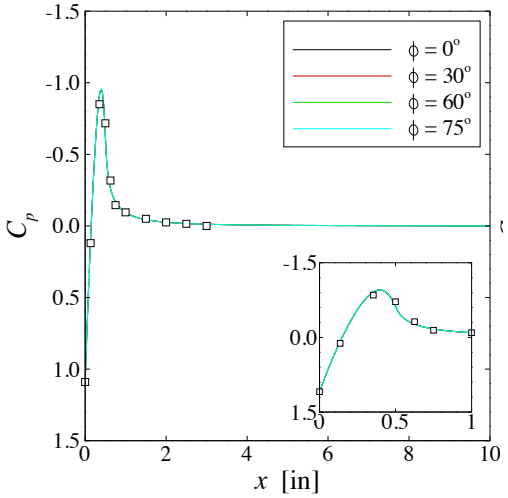
[19]Mayeur, J., Dumont, A., Destarac, D., and Gleize, V., "Reynolds-Averaged NavierStokes Simulations on NACA0012 and ONERA-M6 Wing with the ONERA elsA Solver," *AIAA Journal*, Vol. 54, No. 9, 2016, pp. 2671–2687.



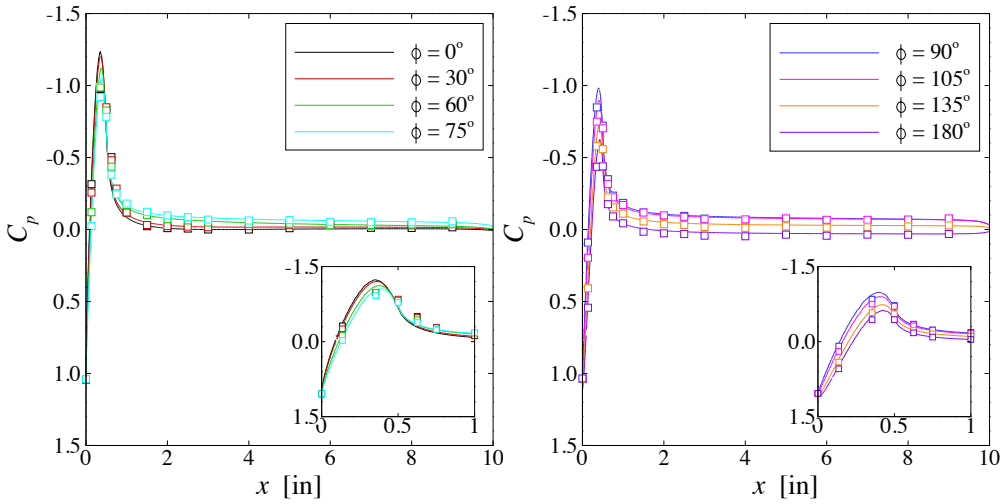
(a) L0 (b) L1

(c) L2 (d) L3

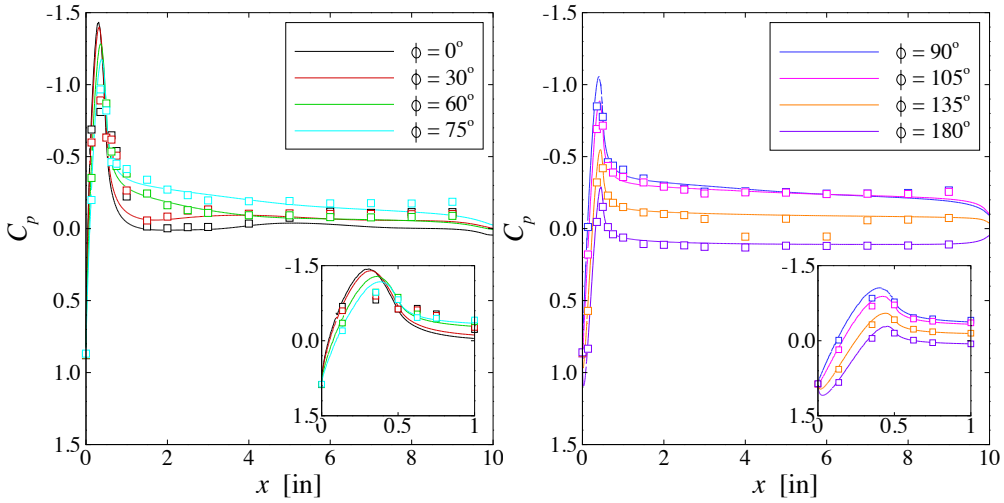
Figure 1: H3D grid family. Red lines denote block boundaries.



(a) $\alpha = 0°$

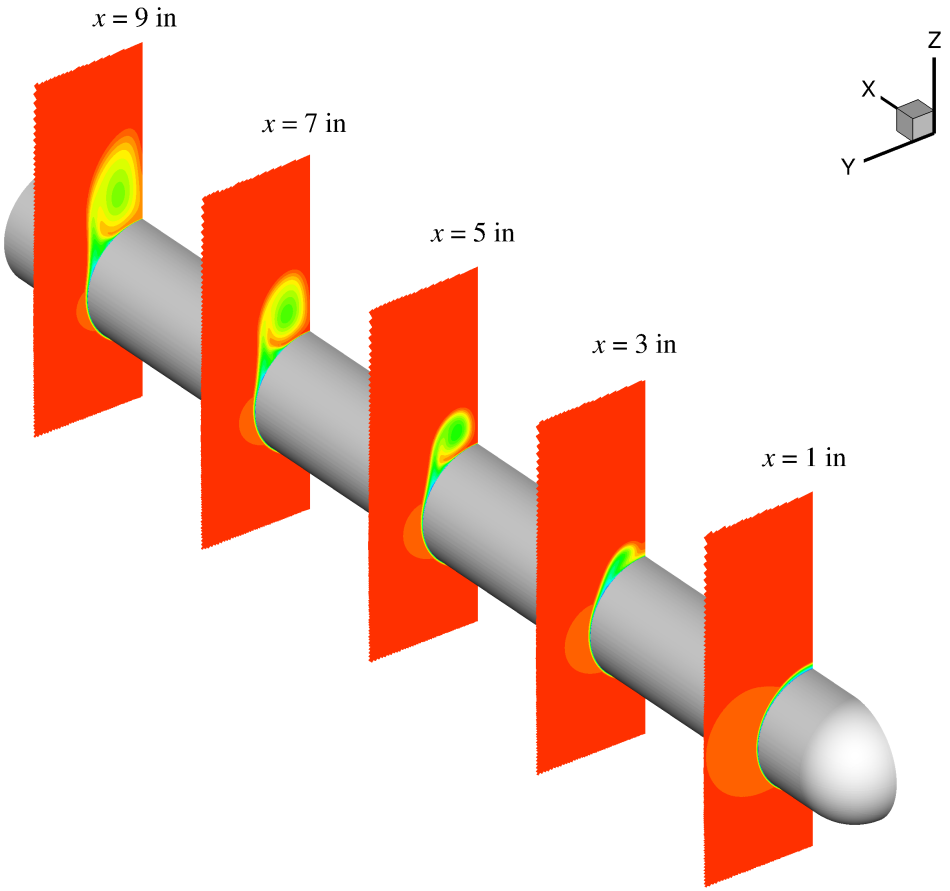


(b) $\alpha = 10°$

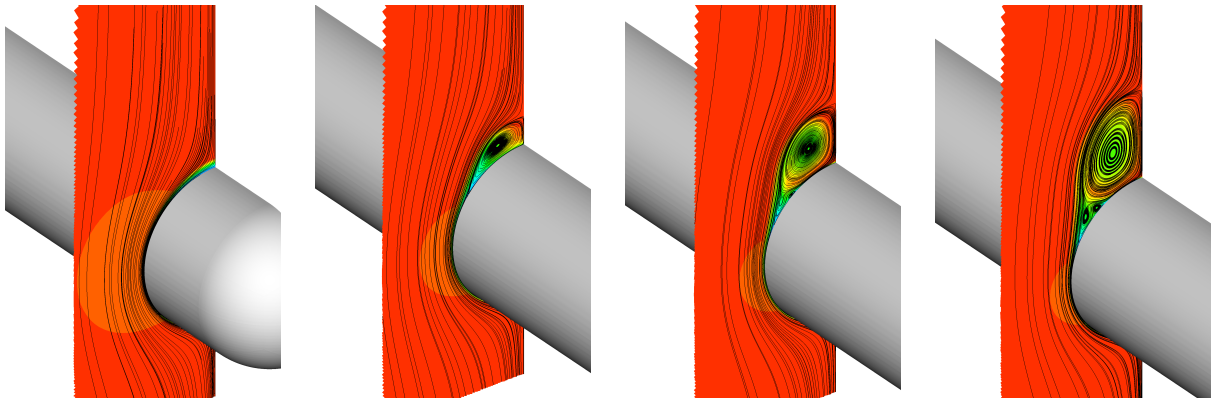


(c) $\alpha = 19°$

Figure 2: H3D: Comparison of computed Cp distributions with experimental results for selected angles of attack.

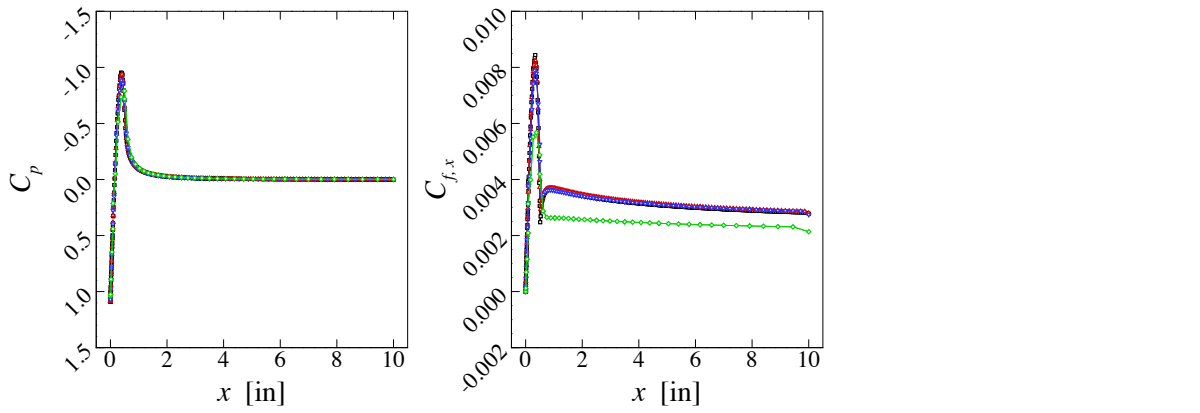


(a) Five slices of total pressure showing vortex formation.

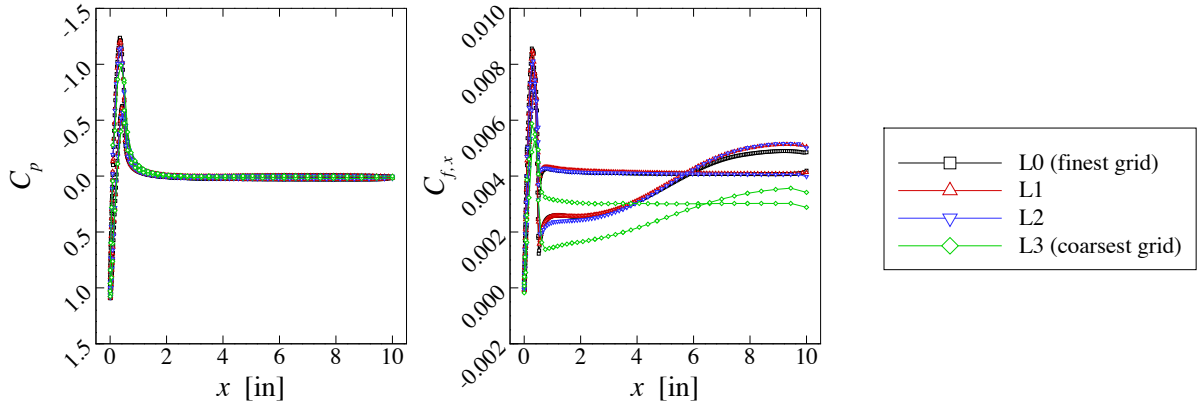


(b) $x = 1\,\text{in}$ (c) $x = 3\,\text{in}$ (d) $x = 5\,\text{in}$ (e) $x = 7\,\text{in}$

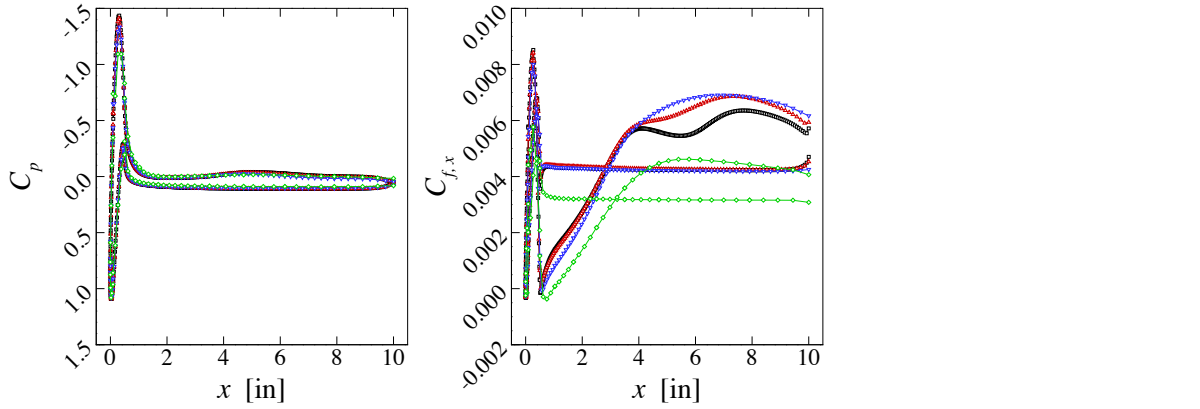
Figure 3: H3D at $\alpha = 19°$: Five slices of total pressure showing vortex formation, with details of four slices showing in-plane streamlines.



(a) $\alpha = 0°$

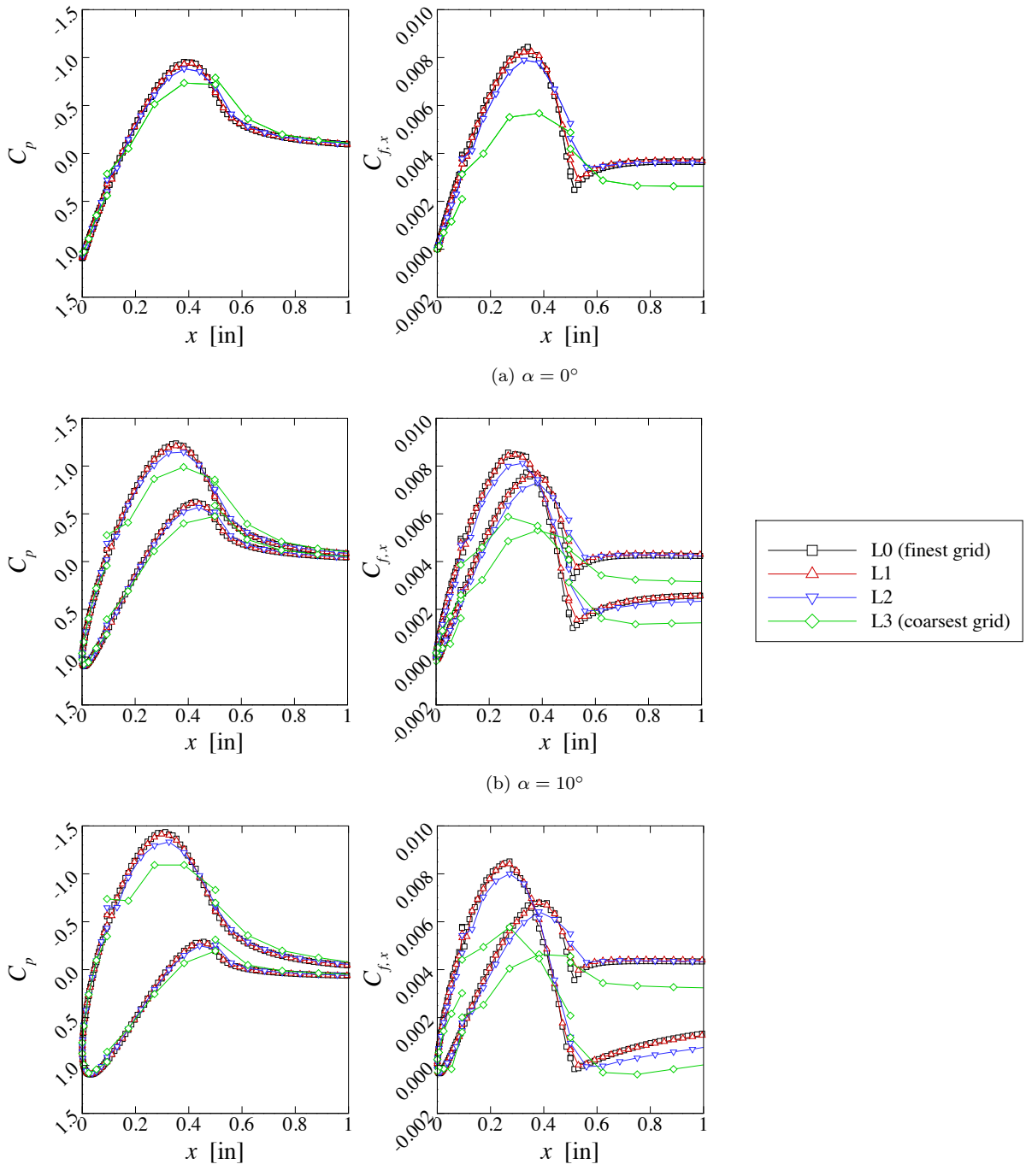


(b) $\alpha = 10°$



(c) $\alpha = 19°$

Figure 4: H3D: Resolution of C_p and $C_{f,x}$ with grid refinement.

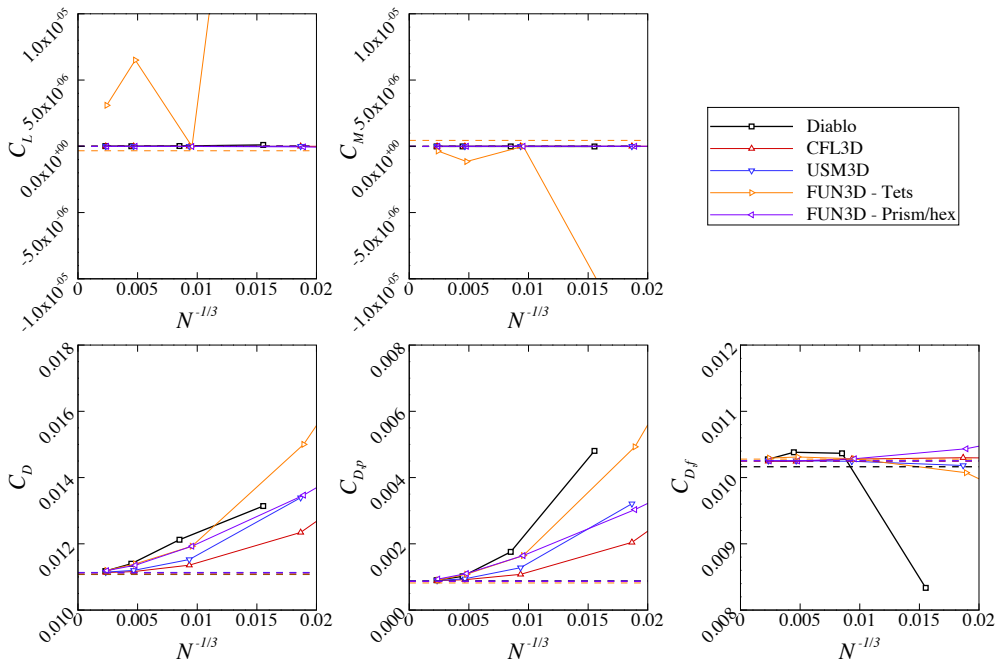


(a) $\alpha = 0°$

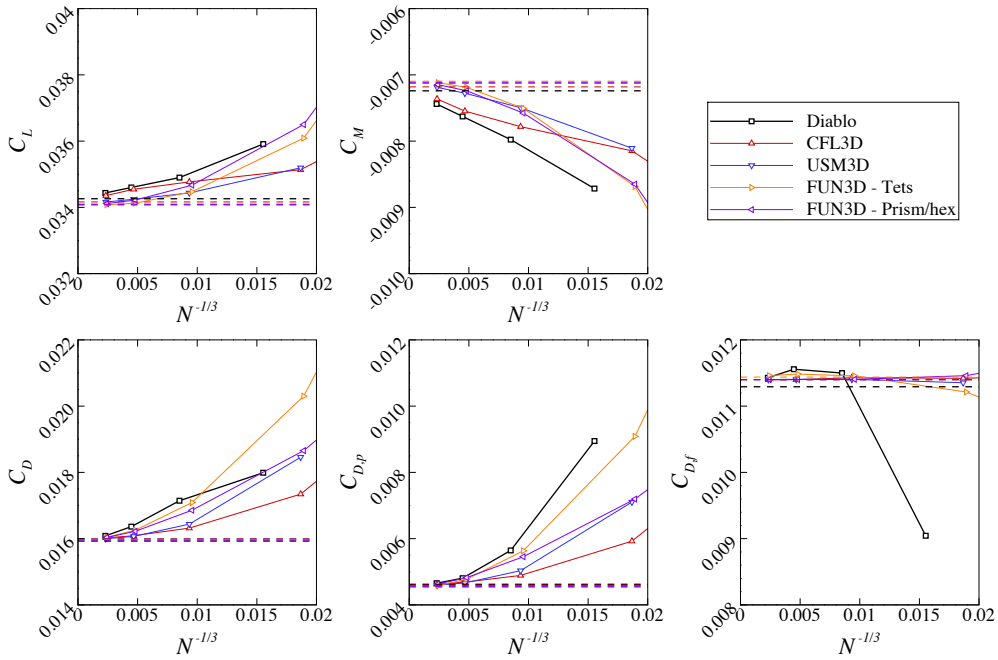
(b) $\alpha = 10°$

(c) $\alpha = 19°$

Figure 5: H3D: Detail of C_p and $C_{f,x}$ near the suction peak with grid refinement.

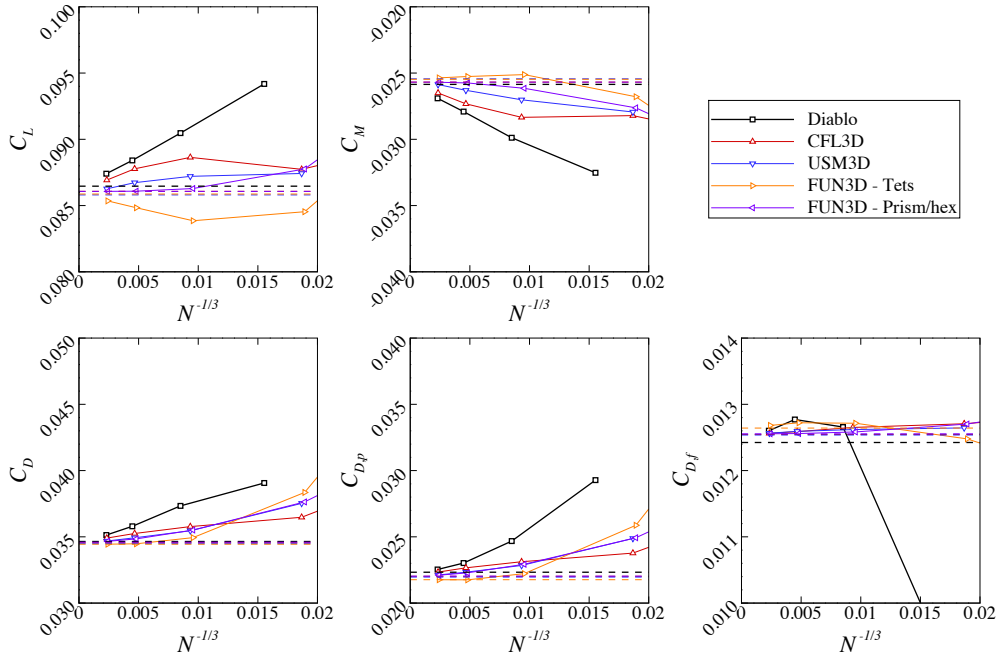


(a) $\alpha = 0°$



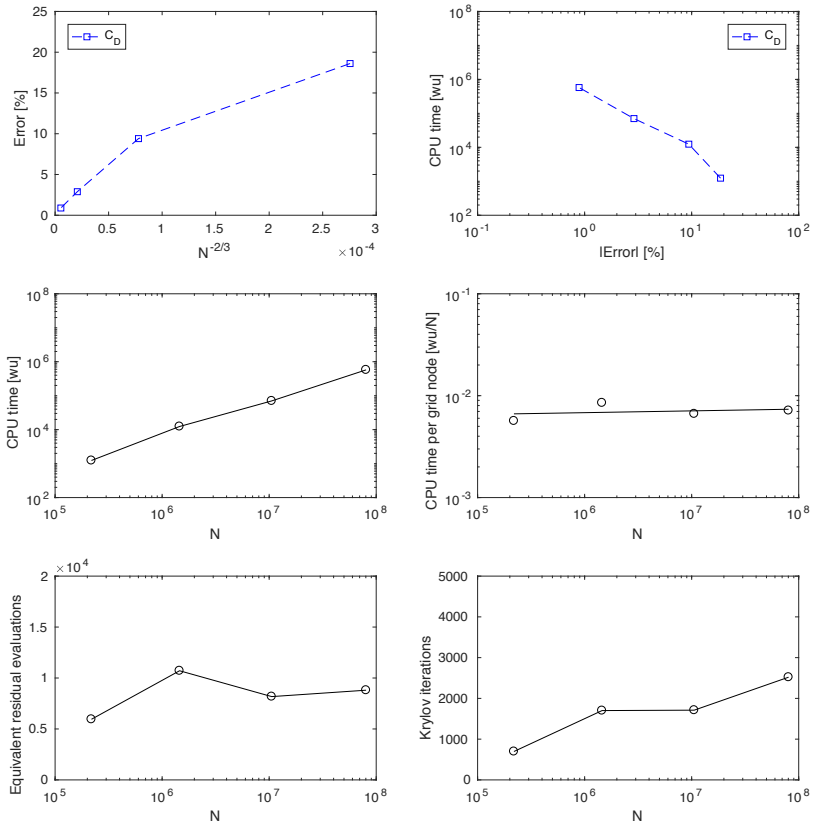
(b) $\alpha = 10°$

Figure 6: H3D: Force convergence behaviour of Diablo compared with other solvers. Dashed horizontal lines represent extrapolated values for each solver.



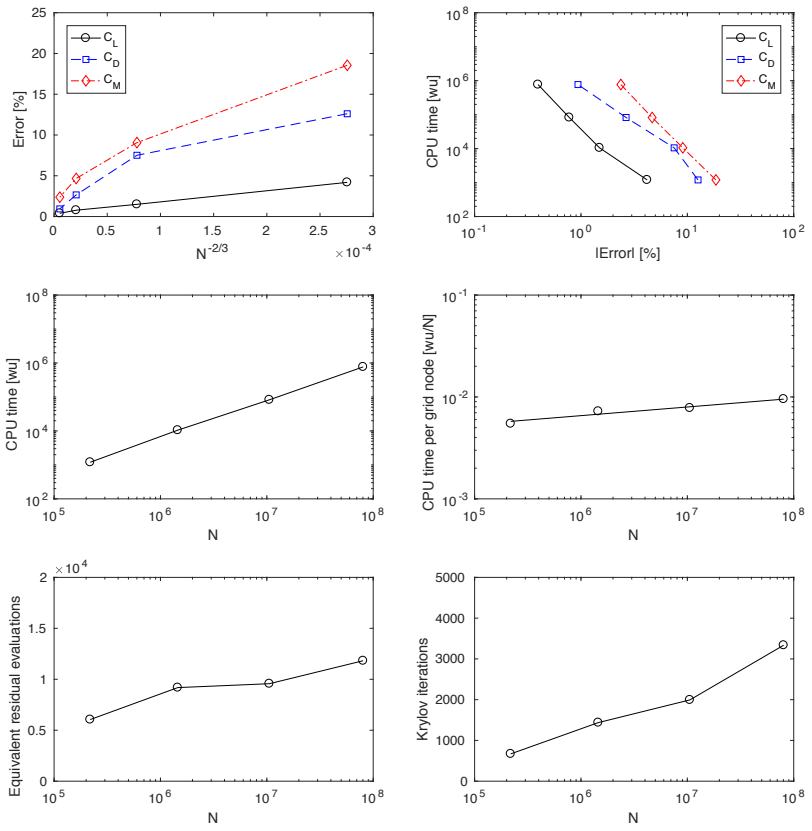
(c) $\alpha = 19°$

Figure 6 (continued): H3D: Force convergence behaviour of Diablo compared with other solvers. Dashed horizontal lines represent extrapolated values for each solver.

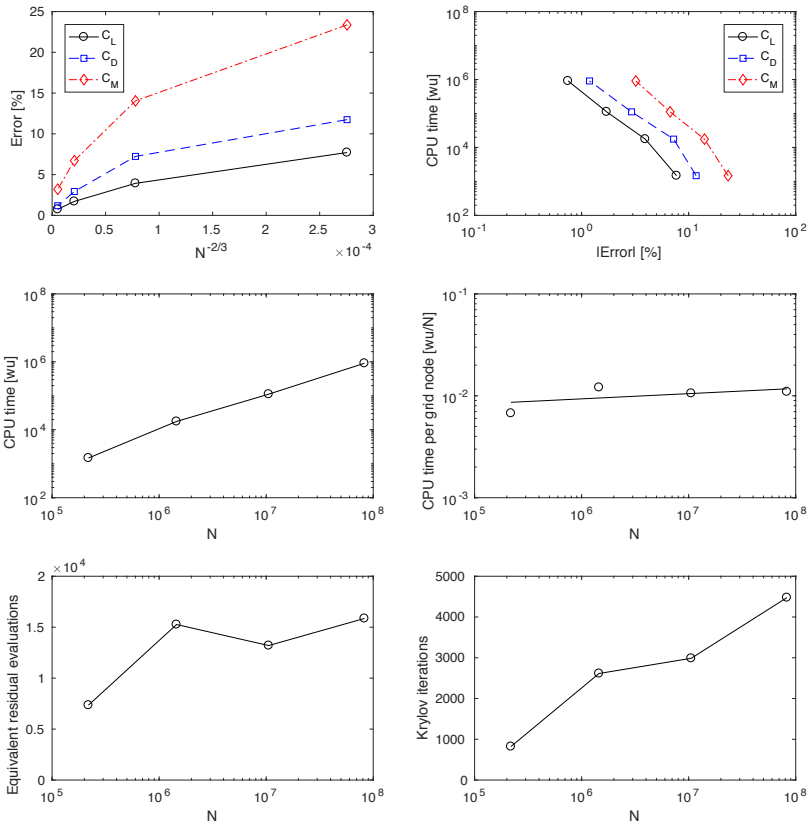


(a) $\alpha = 0°$

Figure 7: H3D: Solver convergence behaviour at selected angles of attack.

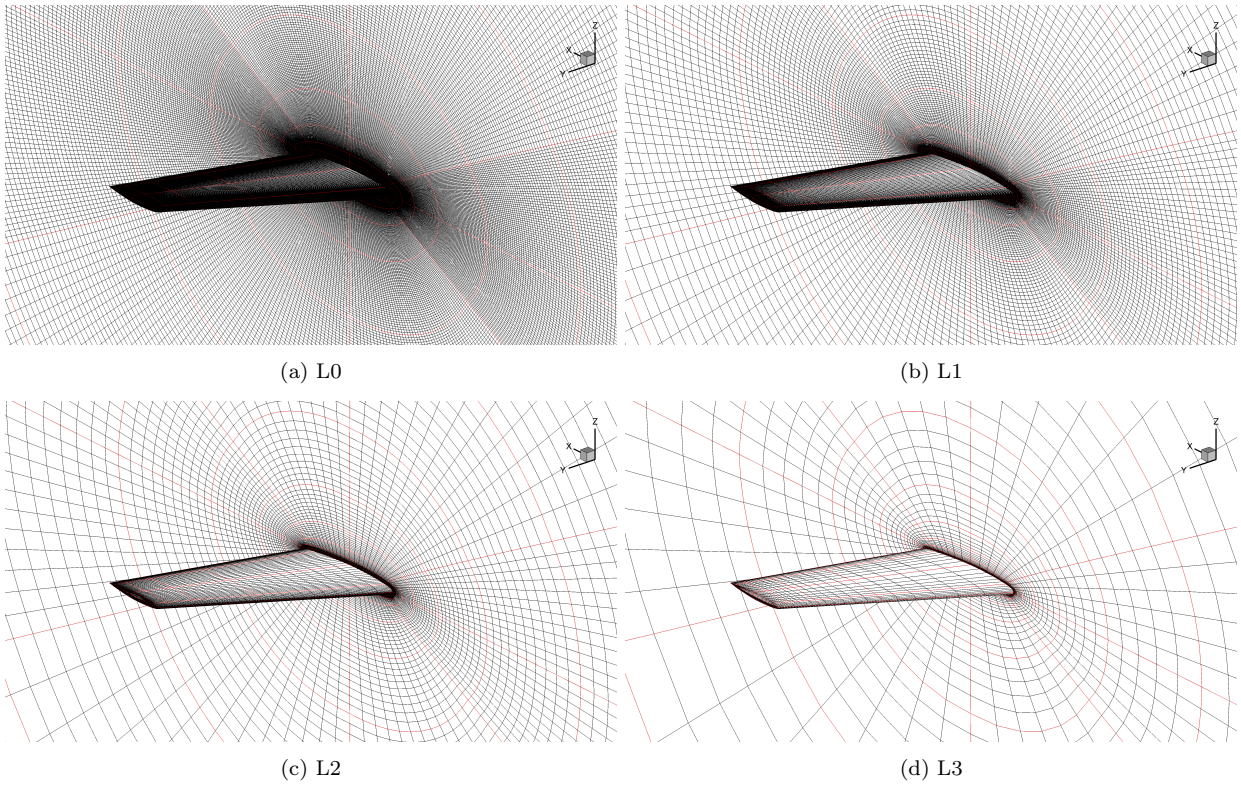


(b) $\alpha = 10°$



(c) $\alpha = 19°$

Figure 7 (continued): H3D: Solver convergence behaviour at selected angles of attack.



(a) L0 (b) L1

(c) L2 (d) L3

Figure 8: OM6 grid family. Red lines denote block boundaries.

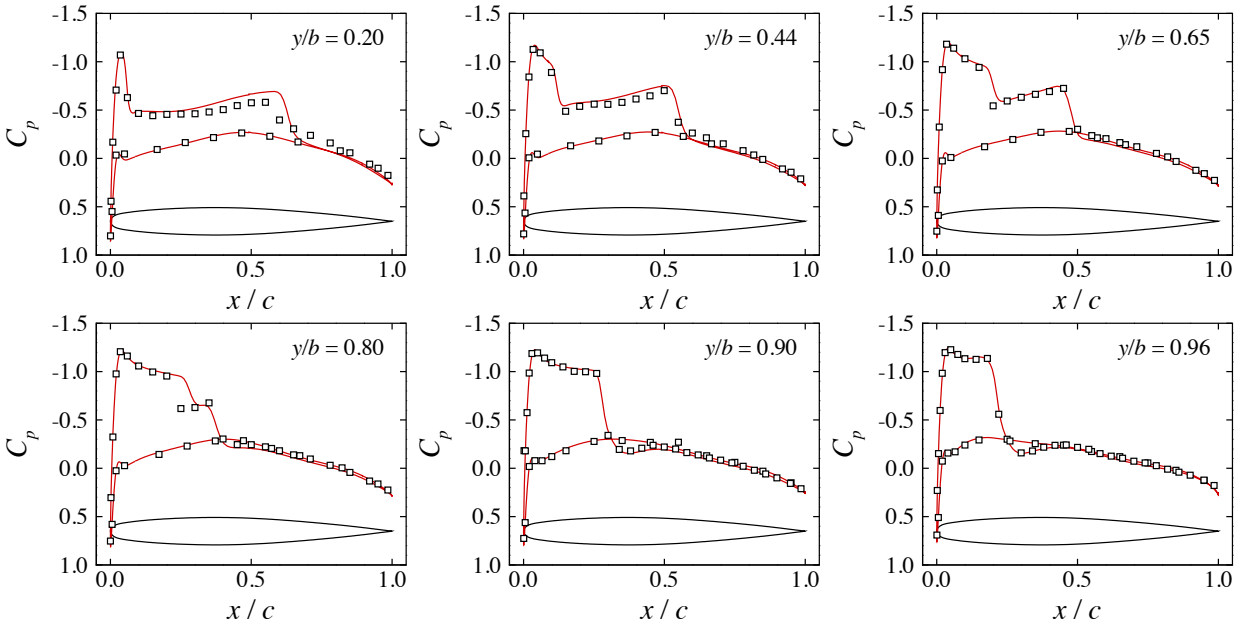
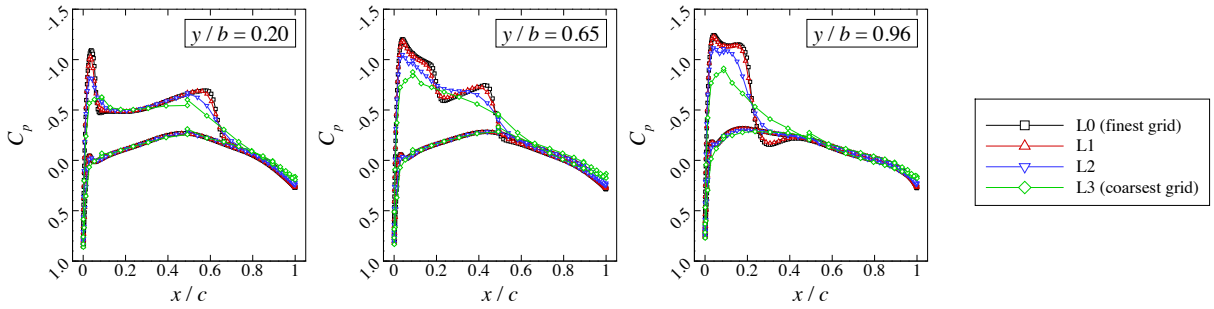
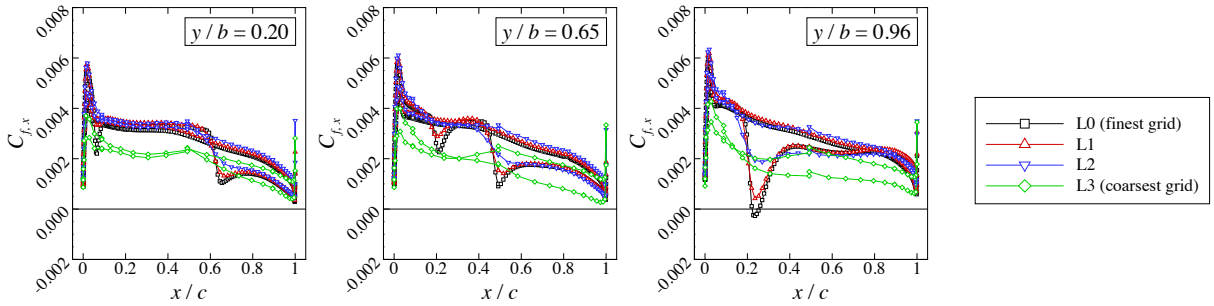


Figure 9: OM6 at $\alpha = 3.06°$: Comparison of computed Cp distributions, computed on the finest grid using scalar dissipation, with experimental results.



(a) C_p distributions



(b) $C_{f,x}$ distributions

Figure 10: OM6 at $\alpha = 3.06°$: Resolution of C_p and $C_{f,x}$ distributions with grid refinement at selected span stations computed with scalar dissipation.

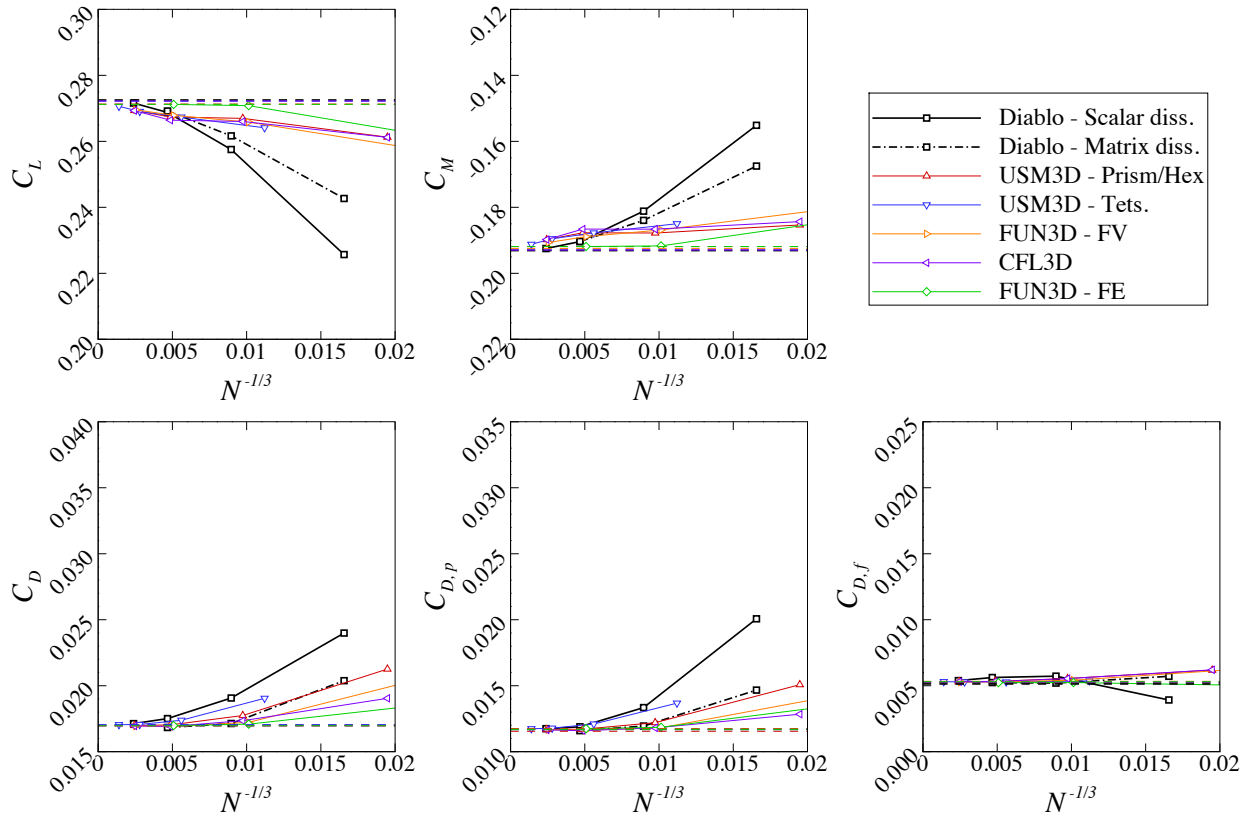


Figure 11: OM6 at $\alpha = 3.06°$: Force convergence behaviour of Diablo compared with other solvers. Dashed horizontal lines represent extrapolated values.

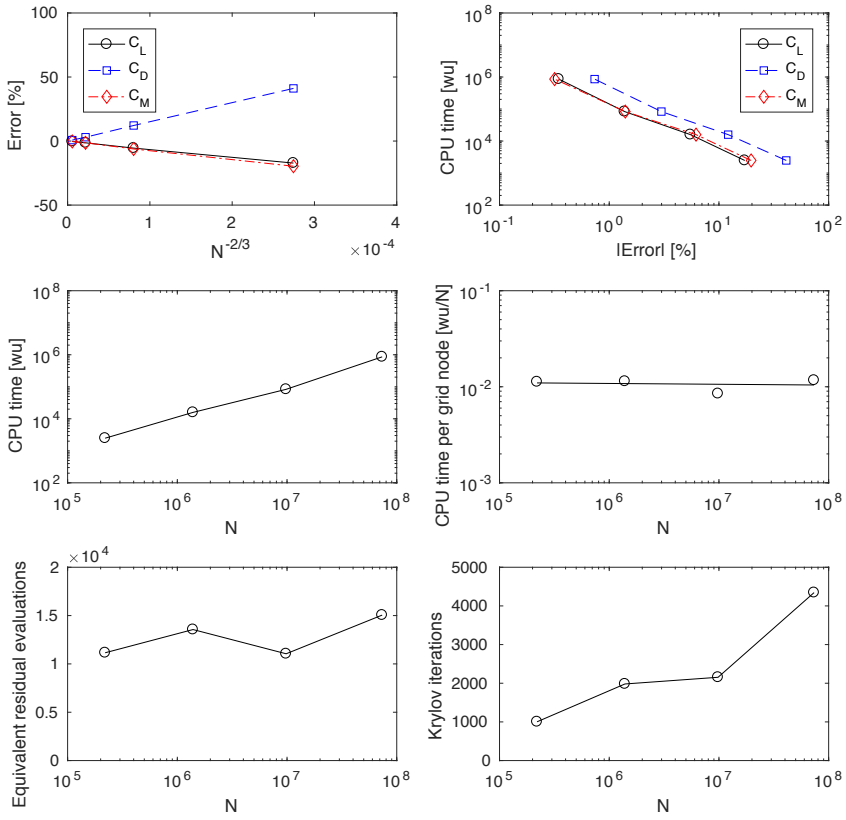


Figure 12: OM6 at $\alpha = 3.06°$: Grid convergence behaviour as computed on the workshop topology grid. Grid is partitioned into 748 blocks.

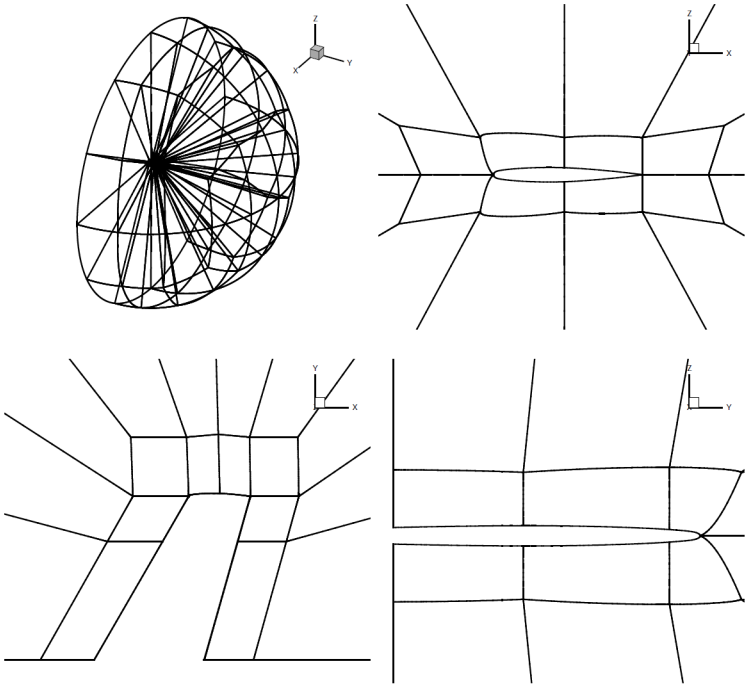


Figure 13: Schematic of the HO-topology grid. Near-body blocks enlarged for clarity.[13]

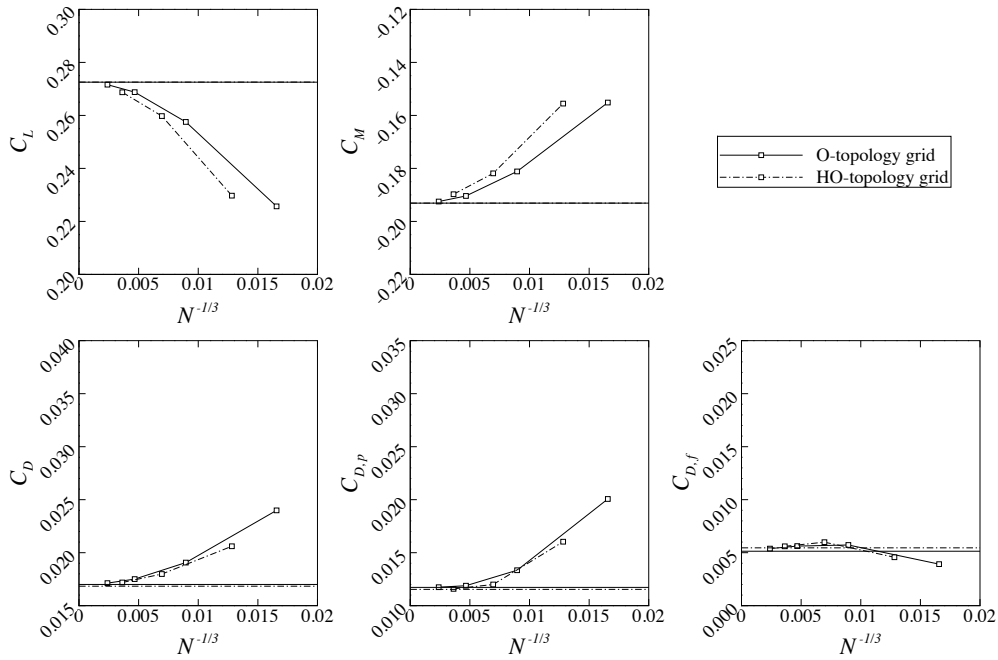
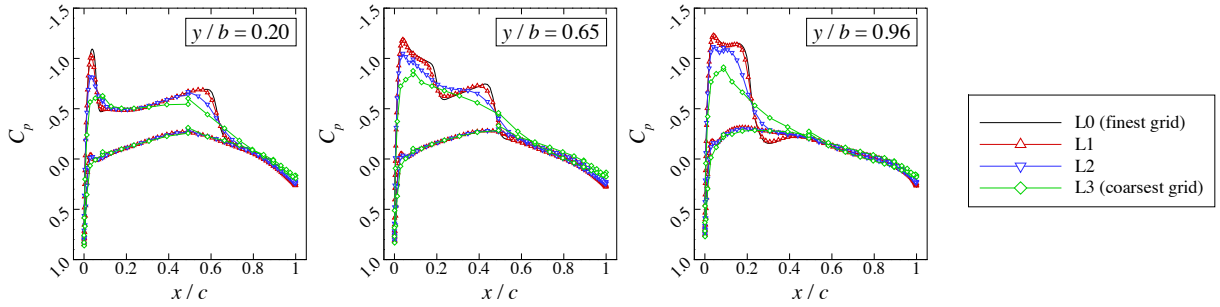
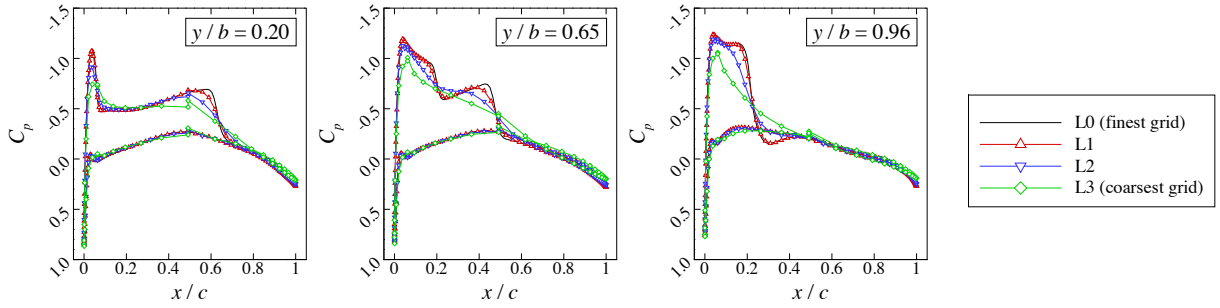


Figure 14: Comparison of functionals computed on the O-topology grid and the alternative HO-topology grid. Both grids have the same surface and near-body resolution.

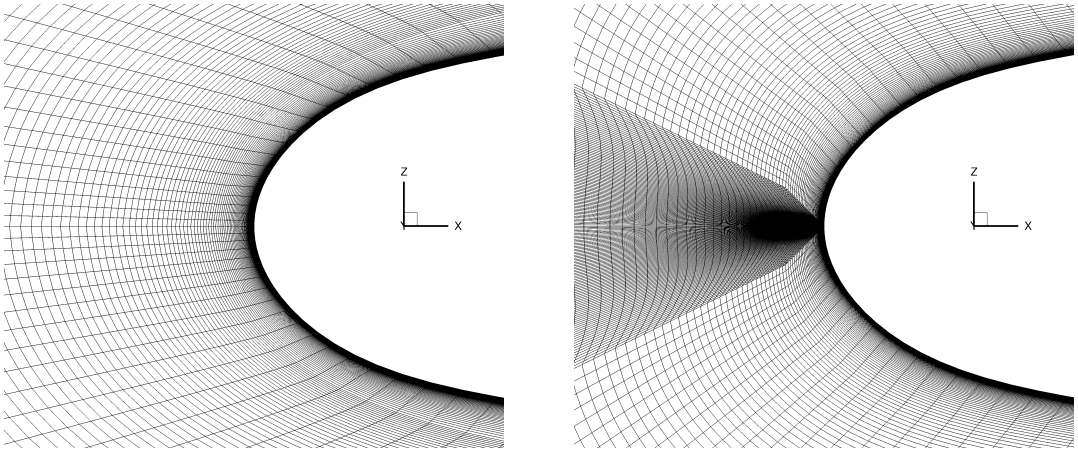


(a) C_p computed on O-topology grids



(b) C_p computed on HO-topology grids

Figure 15: OM6 at $\alpha = 3.06°$: C_p compared on both O and HO-topology grids computed with scalar dissipation.



(a) O-topology grid (b) HO-topology grid

Figure 16: OM6: Detail of the LE region of the L1-level O and HO-topology grids.

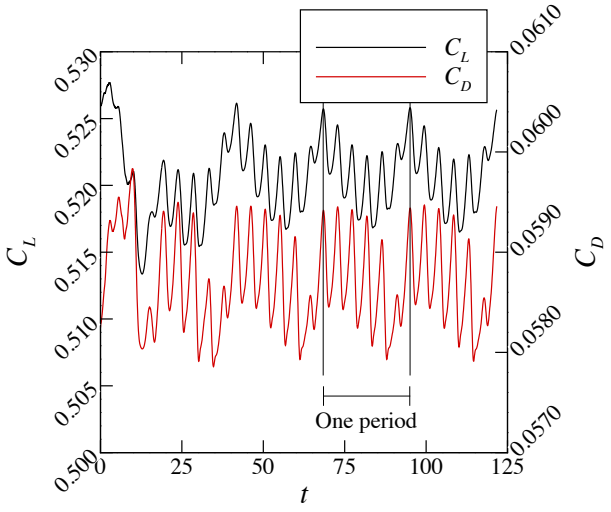
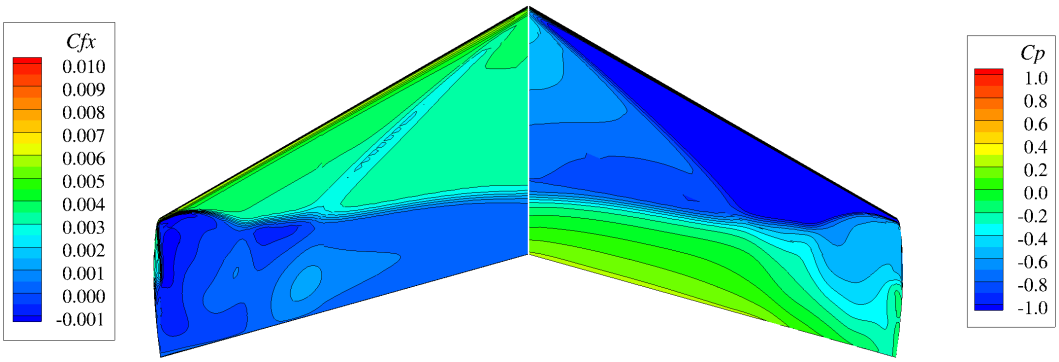
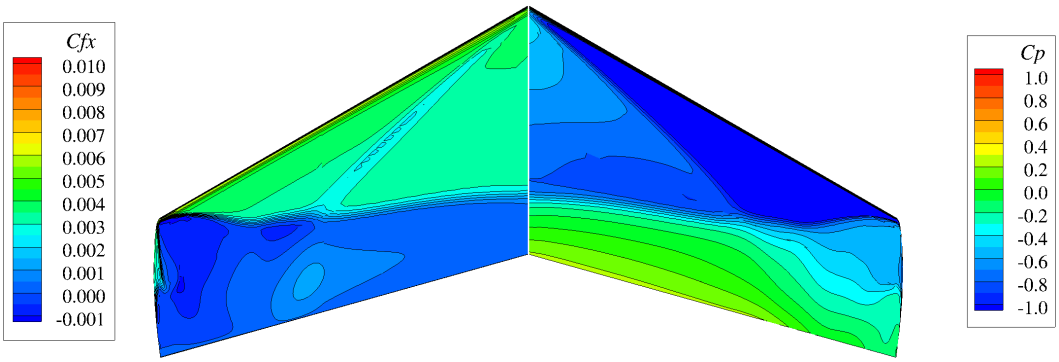


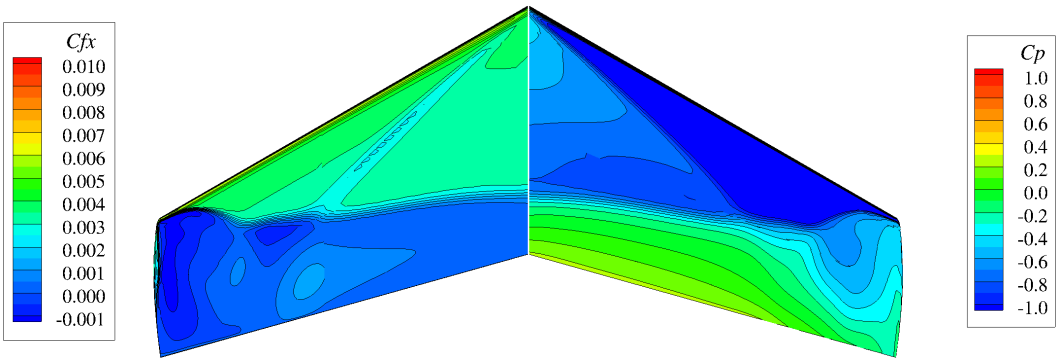
Figure 17: OM6 at $\alpha = 6.06°$: Unsteady force history.



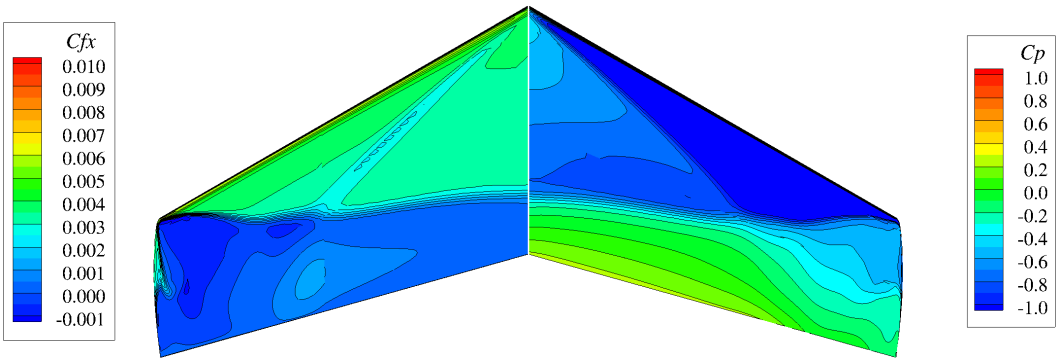
(a) $t = 68.7$



(b) $t = 70.2$



(c) $t = 81.7$



(d) $t = 83.7$

Figure 18: OM6 at $\alpha = 6.06°$: Upper surface $C_{f,x}$ and C_p at four time instances.