H2020 FETHPC-1-2014

Enabling Exascale Fluid Dynamics Simulations
Project Number 671571

D3.3 - Validation Report on Flagship Applications

WP3: Validation & Case Studies

DOCUMENT INFORMATION

Deliverable Number	D3.3
Deliverable Name	Validation Report on Flagship Applications
Due Date	31/09/2018 (PM 36)
Deliverable lead	University of Southampton
Authors	Satya P. Jammy (University of Southampton) Markus Zauner (University of Southampton) Neil D. Sandham (University of Southampton) Philipp Schlatter (KTH) Ricardo Vinuesa (KTH) Nicolas Offermans (KTH) Adam Peplinski (KTH) Filipe F. Buscariolo (McLaren Racing/Imperial College London/USP) Julien J. F. Hoessler (McLaren Racing) Spencer J. Sherwin (Imperial College London) Francesco Bottone (McLaren Racing)
Responsible Author	Satya P. Jammy (University of Southampton) e-mail: `S.P.Jammy@soton.ac.uk`
Keywords	simulations, use cases, flagship calculations, performance, evaluation
WP	WP3
Nature	R
Dissemination Level	PU
Final Version Date	07/10/2018
Reviewed by	Erwin Laure (KTH), Niclas Jansson (KTH), Nick Johnson (UEDIN)
MGT Board Approval	07/10/2018

DOCUMENT HISTORY

Partner	**Date**	**Comment**	**Version**
University of Southampton	7/09/2018	Initial draft	0.1
University of Southampton	7/09/2018	Updated based on reviewer comments	0.2
University of Southampton	23/09/2018		0.3
KTH	26/09/2018		0.4
McLaren Racing	27/09/2018	Added McLaren results	0.5
Imperial, Southampton, KTH	5/10/2018	Final interpretation of results	0.6
KTH	7/10/2018	Final version	1.0

Executive Summary

This document presents flagship simulations for each ExaFLOW Use Case. These are large-scale runs that are designed to push the limits of the available hardware and highlight the success of the algorithmic developments achieved throughout this project by comparing against metrics such as scalability, runtime reduction, error rates, and energy efficiency. The specific algorithmic developments from WP1/WP2 that have been used in each use case are given, along with a discussion on the effects that these algorithmic improvements have had on the use case with respect to the initial results presented in Deliverable 3.1.

Contents

1 Introduction

This document follows on from the preliminary evaluation of the ExaFLOW use cases (see Deliverables 3.1 and 3.2) with respect to the novel algorithms developed in WP1 and implemented in WP2. Here, in particular, we detail large-scale 'flagship' calculations that push the limits of the algorithms and available computing resources, in order to showcase their effectiveness in preparing fluid dynamics codes for exascale. This includes using substantially more numerical grid points and more demanding set-ups, such as going to higher Reynolds numbers.

Each use case has its own dedicated section within this report, detailing the relavant novel ExaFLOW algorithms (developed in WP1 and implemented in WP2). Where possible, comparisons are made between the simulation from the initial Use Case assessment with the flagship simulation with respect to metrics such as scalability, runtime reduction, error rates, and energy efficiency, in order to demonstrate the achievements of the project.

2 Incompressible flow over airfoil

Within ExaFLOW project KTH has developed a fully functional SEM-based AMR solver for the incompressible Navier-Stokes equations. This allows much bigger test cases to be carried out at reduced cost, as the high resolution grid is placed only in the region where it is needed. As the grid modification is performed dynamically during the simulation, the computational error can be controlled on the fly improving the quality of the simulation. In addition, a bigger domain simplifies the treatment of boundary conditions and in some cases allows fully consistent LES or DNS simulations that do not rely on simplified models like RANS. An important additional advantage is the simplification of mesh generation, as both the flagship runs were used the same initial zero-level mesh, even though their final structure differs significantly. This brings an important limitation, as in our implementation refined element inherit their shape and aspect ratio form their coarse parents. That means we cannot easily adapt element shape at different resolution levels that could lead to over-resolved region in a given direction and finally reduce the time step due to the CFL condition.

The turbulent flow around a NACA4412 wing section with 5° angle of attack, at a Reynolds number based on inflow velocity U_∞ and chord length c of $Re_c = 200,000$ is considered as a flagship calculation by KTH. This case is part of a series of well-resolved large-eddy simulations (LESs) conducted with the spectral-element code Nek5000 [3], at Re_c values ranging from $100,000$ to $1,000,000$, and discussed in detail by Vinuesa *et al.* [13]. This flow configuration was chosen to illustrate the great benefit of using adaptive mesh refinement (AMR) approaches, in particular when it comes to the farfield region in the computational domain. Figure 1 (left) shows a sample spectral-element mesh used in one of our wing simulations, where a structured meshing strategy was used. It can be observed that the current methodology clusters large numbers of spectral elements towards the farfield, where there are no turbulent fluctuations and lower mesh resolutions could be used without loss of accuracy in the simulation. Furthermore, the moderate Reynolds number yields a tractable

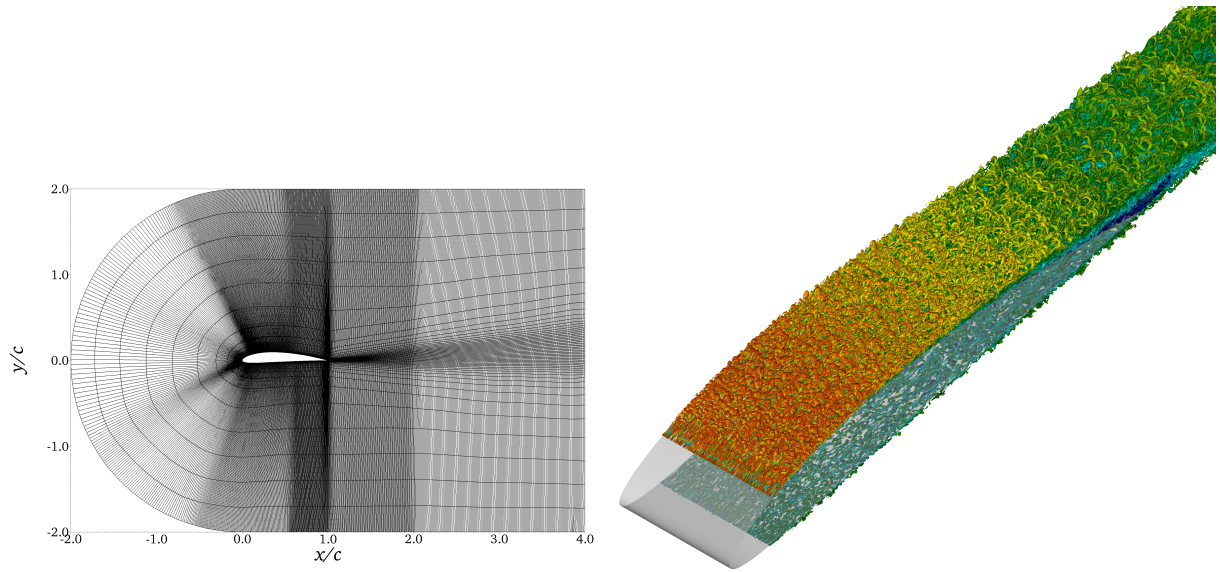


Figure 1: (Left) Two-dimensional slice of the computational domain showing the spectral-element distribution at $Re_c = 1,000,000$, but not the individual Gauss–Lobatto–Legendre (GLL) points. (Right) Instantaneous visualization showing coherent vortical structures around a NACA4412 wing section at $Re_c = 200,000$. The structures are colored by the magnitude of the streamwise velocity, where dark blue denotes -0.1 and dark red 2. Figures extracted from Vinuesa *et al.* [13].

problem size, for which the improvement in terms of required number of grid points can be illustrated at a reasonable computational cost. An instantaneous visualization of the flow around the wing at $Re_c = 200,000$ is provided in Figure 1 (right), which also shows the level of detail achieved in the characterization of the vortical structures in the turbulent boundary layers (TBLs) on both the suction and pressure sides. This figure also shows how the TBL on the suction side, which is exposed to a progressively stronger adverse pressure gradient (APG) in the streamwise direction, exhibits significant thickening towards the trailing edge and progressively more energetic large-scale motions due to the pressure gradient.

In this section we reproduce the simulation at $Re_c = 200,000$, which was originally [13] performed on a mesh similar to the one shown in Figure 1 (left), using AMR. The reference case was tripped [12] on both the suction and pressure sides, in order to prescribe the location of the transition to turbulence, and it was run for long enough to obtain converged turbulence statistics around the wing section. The computational domain had horizontal (x) and vertical (y) lengths $L_x/c = 6$ and $L_y/c = 4$, whereas in the periodic spanwise (z) direction $L_z/c = 0.2$. This implies that vertical distances of $2c$ were considered between the airfoil and the upper and lower boundaries. The reason to consider this relatively small domain in y is connected to the maximum allowed stretching of the spectral elements, which already in Figure 1 (left) exhibits a significantly larger dimension in y than in x and z. The high aspect ratios in the spectral elements lead to very high iteration counts to solve the pressure, with a detrimental effect in the overall efficiency of the code. In the reference case, the farfield

boundary conditions were based on a preliminary Reynolds-averaged Navier–Stokes (RANS) simulation [6], the result of which is imposed a Dirichlet boundary condition. The stabilized outflow by Dong *et al.* [2] was also considered. By using AMR, we aim at improving the current setup in two ways: first, we expect to reduce the number of elements in the far field region, therefore reducing the total number of elements of the simulation and improving the code performance; and second, the increased flexibility in element shape will allow us to get rid of the aspect ratio limitation in the spectral elements, thus allowing us to consider significantly larger computational domains and to avoid the use of RANS-based boundary conditions.

The mesh in the reference case was designed by following traditional guidelines for well-resolved LES around the wing section, with $\Delta x_t^+ < 18$, $\Delta y_{n,w}^+ < 0.64$ and $\Delta z^+ < 9$. Here x_t and y_n denote the coordinates in the wing tangential and normal directions, respectively, and $\Delta y_{n,w}$ is the grid spacing between the wall and the first grid point in the wall-normal direction. Viscous scaling in terms of $\ell^* = \nu/u_\tau$ (where ν is the fluid kinematic viscosity and u_τ is the friction velocity) is denoted by '+'. In the wake region, the mesh was designed following $\Delta x/\eta < 9$, where $\eta = (\nu^3/\varepsilon)^{1/4}$ is the Kolmogorov scale and ε is the local isotropic dissipation. Following these guidelines, the case reported by Vinuesa *et al.* [13] had 9,324 elements in the xy plane, with 36 uniformly distributed elements in z, which leads to a total of 335,664 spectral elements. With a polynomial order $N = 9$, this amounts to 335.66×10^6 grid points with $L_z/c = 0.2$.

To prepare the AMR simulations we start by constructing a very coarse (zero-level) conforming mesh, that is significantly bigger than the presented one in all three directions and is a starting point to develop, during the simulation, the finer, nonconforming mesh. The horizontal (x), vertical (y) and spanwise (z) lengths of the this mesh are $L_x/c = 40$, $L_y/c = 40$ and $L_z/c = 0.6$ respectively. We have to stress here, that this mesh is three times wider than the original one, which we have to take it into account when comparing the global number of elements and execution time of the AMR run to the reference one. The much bigger size of the new domain allows us to relax boundary condition constrain, so the AMR runs are fully consistent and do not rely on RANS simulations. We use simple Dirichlet inflow condition at the inflow, a 'do nothing' outflow condition, and normal outflow conditions on the top and bottom of the box. The domain is periodic in the spanwise direction and, in addition, we apply a sponge zone four units downstream the wing to reduce the length of the wake and minimise the number of elements in the far field. The position we start our sponge forcing corresponds roughly to the extent of the reference mesh, however the forcing is gradually increased downstream, so the visible wake is much longer. We have to notice here that the length of the wake visible on the upper-right panel in Figure 2 is not caused by the sponge forcing, but by the relatively short time of the simulation.

Creating this mesh we paid attention to mesh properties like the element aspect ratio and skewness, and also used a mesh smoother [7]. This allowed element's shapes to be significantly improved and helped to reduce the iteration count of the pressure solver. As in the reference case, the 3D zero-level mesh is an extension of the 2D mesh, with 730 elements in a spanwise direction, by uniformly distributing three elements in z, that gives

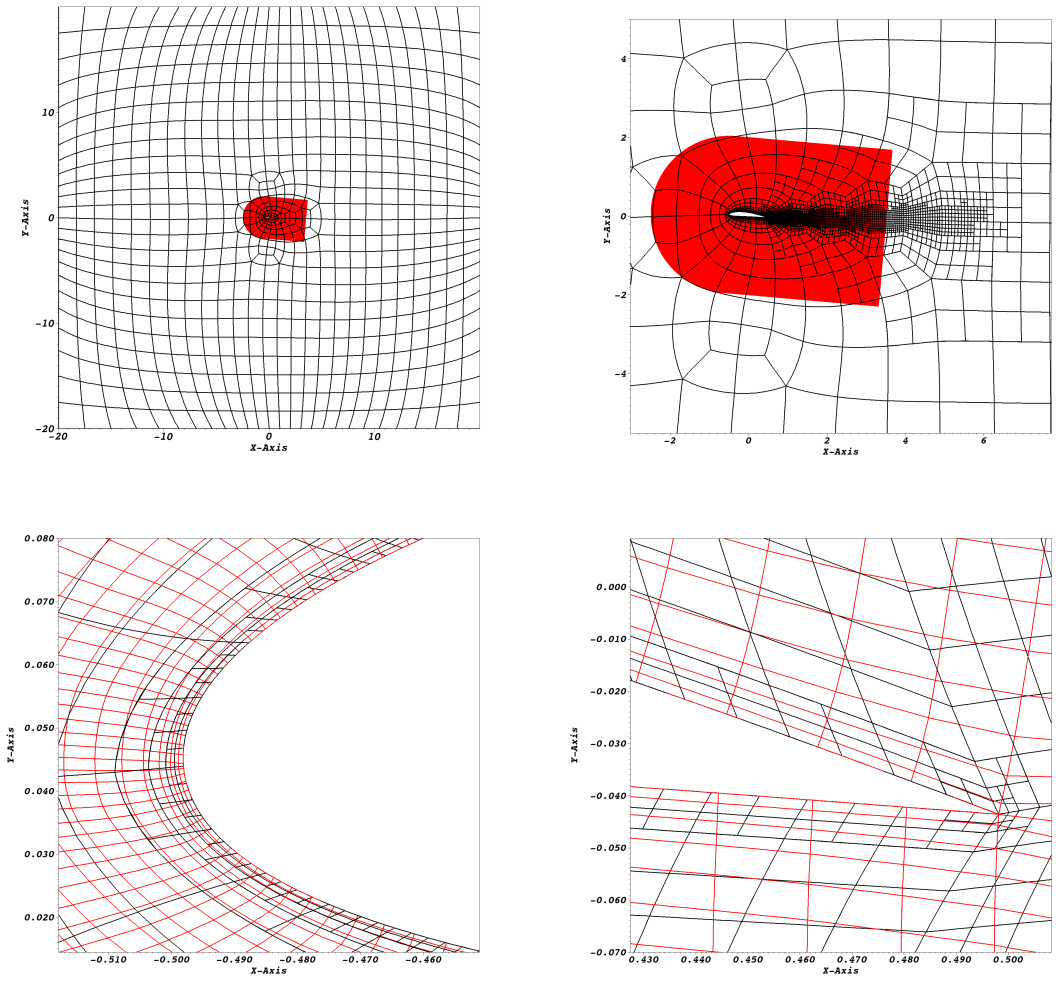


Figure 2: Comparison of the meshes of the reference and AMR simulations at $Re_c = 2 \times 10^5$. All plots present 2D cuts through the mesh, oriented perpendicular to the spanwise direction, with the reference and AMR elements boundaries marked in red and black respectively. Unlike the AMR mesh, the reference one does not evolve during the simulation. Upper left and right plots shows zero-level AMR mesh before any refinement take place and the evolved mesh at time $t = 7.2$ respectively. The refined region of the wake is clearly visible. The lower left and right panels give the details of the element structure at $t = 7.2$ at leading and trailing edges respectively.

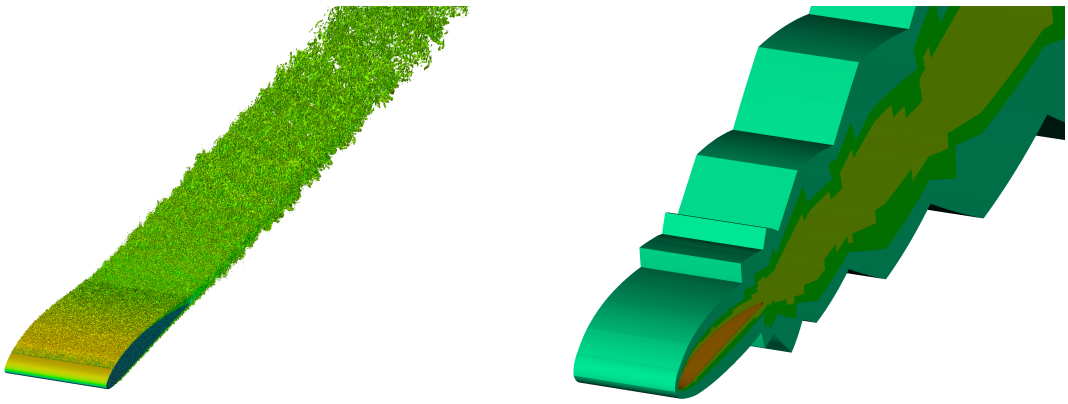


Figure 3: Vortical structures (λ_2 criterion) of the velocity field around the wing (left panel) and the part of the domain covered by refinement levels higher and equal to two (right panel) at simulation time 7.2 for the $Re_c = 2 \times 10^5$ case.

total 2190 elements of 3D base mesh. The comparison of the base and the reference meshes is presented in the upper-left plot in Figure 2. However the elements of the original mesh (marked red) are too small to be visible. To approximately match the resolution of the reference case at the wing surface we add 5 levels of refinement, so the final number of elements in z at the wall is equal to 96. As we use oct-refinement, the resolution in x and y grows accordingly and is comparable with the resolution of the reference case, however there is a visible difference between leading and trailing edge resolution due to the mesh smoothing process (lower plots in Figure 2). Even though the original 2D coarse mesh was designed to exactly match the wall-normal resolution of the reference one, the smoothing process stretched the elements close to the trailing edge by about 50% and reduced the wall-normal resolution. On the other hand, the aspect ratio of the refined elements is inherited from the coarse-mesh parents, leading usually to a higher wall-tangential resolution of the AMR mesh. As a single refinement increases the resolution in the given position by a factor of two, we fixed the maximum refinement level at the wing surface to 5, to have a case that is comparable with the reference one. The exception is the trailing edge itself, where the 6th refinement level is allowed. This provides flexibility to the AMR, which has the potential to reduce problems caused by sharp edges at minimal cost.

The base mesh was later modified according to the spectral error indicator with the refine and de-refine thresholds equal to 2×10^{-4} and 5×10^{-6} respectively. To see the effect of nonconforming faces relatively close to the wing, we have restricted the maximum refinement level of all the elements not touching the wing surface to be 4. The only exceptions are elements in the vicinity of the tripping line, that have the same maximum refinement level as the wing surface. This is introduced to avoid possible differences in the forcing due the to jump in the resolution at the tripping line, that could significantly alter the flow statistics, which we use for validation of our AMR implementation. To simplify spanwise averaging of the fields and to avoid interpolation we ensured that all the elements sharing the same

(x, y) coordinates (but having different z) have the same refinement level, and the refinement criteria is taken as a maximum of the estimated error of all the elements along lines parallel to the z axis.

The AMR mesh was evolved in the simulation with a polynomial order $N = 7$ for 7.2 time units, producing the mesh with 224,272 elements and 76.37×10^6 grid points. Taking into account the difference in the domain width we get about a 77% reduction in the element count with respect to the reference case. The visualisation of the instantaneous velocity field at $t = 7.2$ and the corresponding refinement level distribution is presented in Figure 3.

The turbulence statistics obtained from the present AMR simulations are compared with the reference data by Vinuesa *et al.* [13] in Figure 4. The statistics from the AMR case were obtained by averaging in time for 1.44 flow-over times and over the homogeneous spanwise direction. The results in Figure 4. (top) indicate that the outer-scaled mean velocity profile is in very good agreement with the reference data, both for the moderate APG cases on the suction side (ss) and the slightly accelerated case on the pressure side (ps). Regarding the selected Reynolds stresses shown in Figure 4. (bottom), excellent agreement is also observed in the Reynolds shear-stress profile at all the considered locations, whereas small differences are noticed for the spanwise velocity fluctuations, a fact that can be explained by a slight lack of convergence of some statistics. However, it can be stated that the proposed AMR methodology is adequate, and the flow is adequately simulated.

2.1 Scaling tests

We have performed a strong scaling test on two petascale Cray XC40 systems Beskow at PDC and Hazel Hen at HLRS. Beskow consists of 2,060 nodes with 67,456 cores in total and 2.438 PFLOPS peak performance. On the other hand Hazel Hen is built of 7,712 nodes with 185,088 cores in total and 7.420 PFLOPS peak performance. On a single node on Beskow and Hazel Hen we use 32 and 24 cores respectively.

We ran two sets of simulations with the small and big test cases based on the AMR turbulent flow around a NACA4412 wing section. The small one corresponds to the described $Re_c = 2 \times 10^5$ case with maximum refinement level equal 6. It consists of 224,272 elements, that for a polynomial order $N = 7$ give 76.37×10^6 grid points. For the big case $Re_c = 1 \times 10^6$ and the maximum refinement level is equal 8. The mesh is build of 2,517,080 elements giving 859.38×10^6 grid points. On Beskow we ran only a small case, however on Hazel Hen we executed both cases. We compare our results with the scaling results of the conforming Nek5000 presented in [9]. The most interesting test presented in this work is $Re_\tau = 360$ pipe flow, especially the upper-right plot in Figure 5, as it is similar in size with our smaller case. We should mention here, that our goal is not to improve the parallel performance of the conforming code, but to retain performance in the presence of the additional operator introduced in the direct stiffness summation in the nonconforming solver.

Both sets of simulations are run for 100 steps starting from the same initial conditions for small and big case respectively. We run a single MPI process per core and increase the number of computing nodes as a power of two, using:

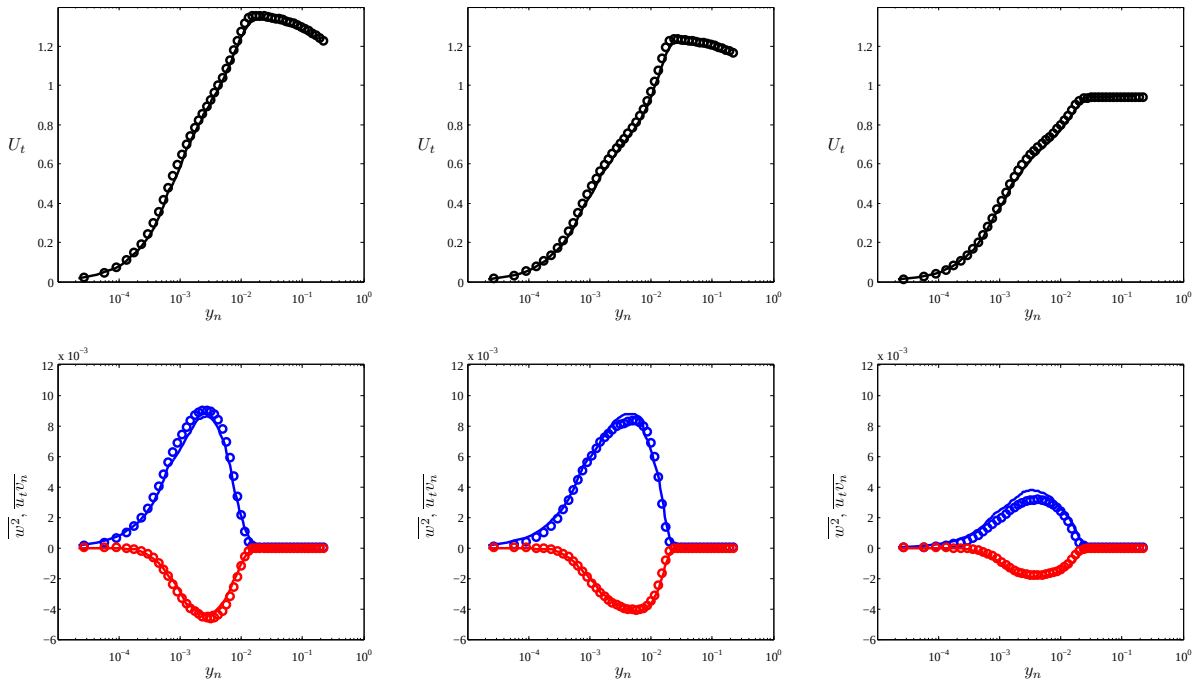


Figure 4: Comparison of turbulence statistics from (circles) reference data from Vinuesa *et al.* [13] and (solid lines) present AMR simulation. (Top) Mean velocity profiles and (bottom) selected Reynolds stresses at (left) $x_{ss}/c = 0.4$, (middle) $x_{ss}/c = 0.6$ and (right) $x_{ps}/c = 0.9$. Statistics scaled with U_∞ and c.

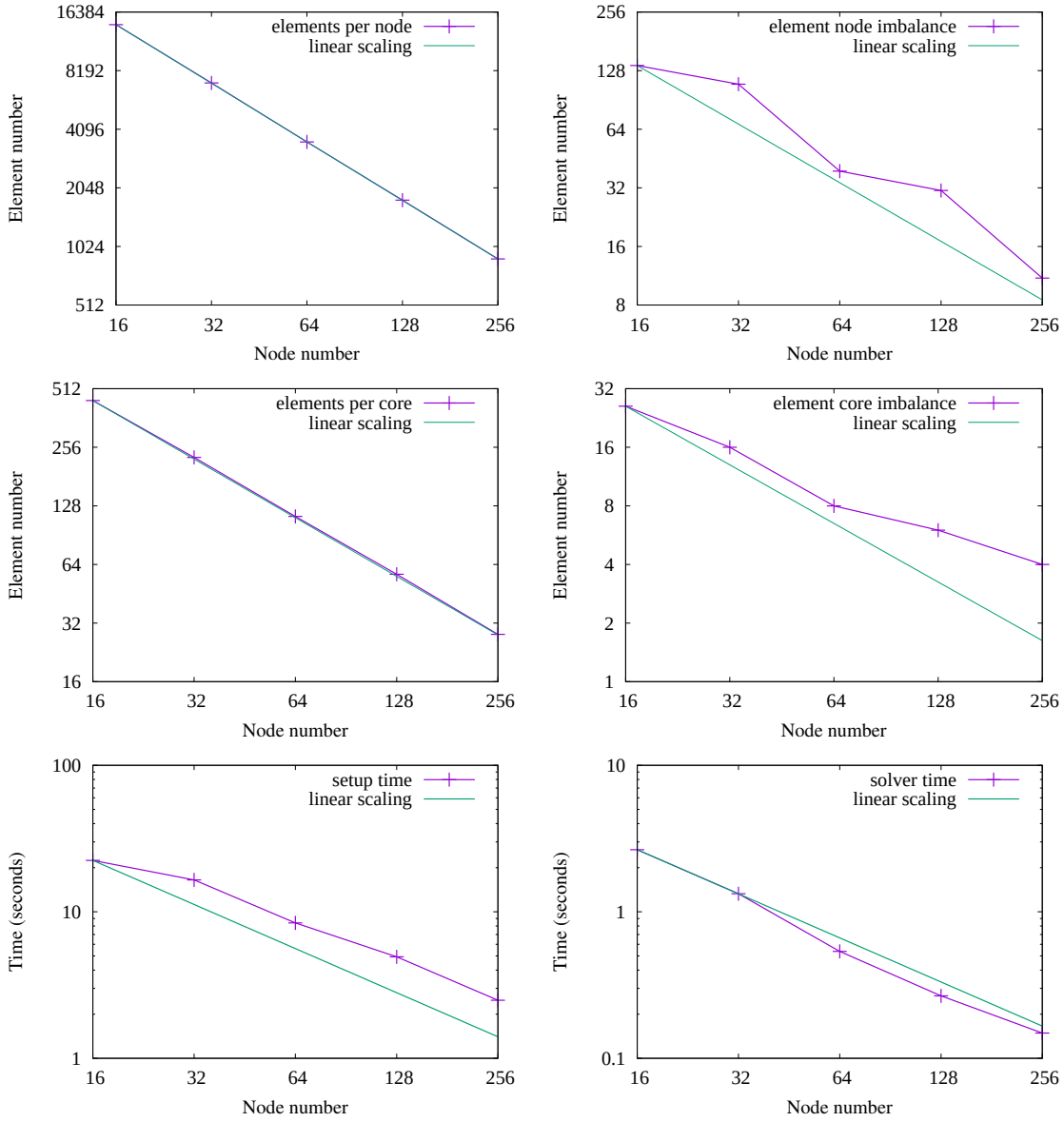


Figure 5: Parallel performance of nonconforming solver for the small test case executed on Beskow. In this case each node consists of 32 cores. The top and the middle rows show results of the two-level partitioning, respectively the inter- and intra-node phase. In each case we present the maximum number of elements per node/core (left plot) and the element imbalance (right plot) as a function of node number. The lower left and right plots show strong scaling of the coarse grid setup phase and the time per time step for the nonconforming solver.

- 16, 32, 64, 128, 256 nodes on Beskow,
- 32, 64, 128, 256 nodes for the small case, and 512, 1024 nodes for the big case on Hazel Hen.

The maximum number of the computing nodes used in the test was not set by the parallel properties of the nonconforming Nek5000, but by the quality of domain partitioning provided by ParMETIS. Within ExaFLOW we developed a new grid partitioning scheme for Nek5000 that takes into account a core distribution among the nodes, and consists of two steps of inter- and intra-node partitioning. Although this two-level partitioning scheme significantly improves the efficiency of the coarse grid operations for XXT, especially during the setup phase, it relies on the quality of the inter-node partitioning. If the first step gives subdomains with disjoint graphs, the second step cannot be performed. We found that probability of getting disjoint graphs increases with decreasing number of elements per node, virtually prohibiting the runs with less than 1000 elements per node. However, this limit can differ between simulations, and the big test case performed on 2048 nodes on Hazel Hen exited due to disjoint graphs with 1251 elements per node. We have to mention here, that in the standard production use of the solver this limitation is not critical, as according to [9] it usually is close to the strong scaling limit of conforming Nek5000, which for Beskow was estimated to be between $30,000$ and $50,000$ degrees of freedom per core.

Studying parallel performance of the code we focus on the strong scaling of the nonconforming solver (time evolution loop only excluding code initialisation, finalisation and any I/O operations), the coarse grid solver setup times, and the work balance that is result of the two-level partitioning.

Results of the small scaling tests executed on Beskow are presented in Figure 5 with the outcome of the two-level partitioning shown in the top and the middle rows (inter- and intra-node graph bisection respectively). The left plots give number of elements per node/core and show ParMETIS to be an efficient grid partitioner, as both curves follow closely the linear scaling line. However, the closer look at the element imbalance (the right plots) demonstrates the dependency of partitioning quality on the size of the partitioned graph. On the node level the element imbalance fluctuates and never exceeds 2% of the maximum number of elements per node, but on the core level (intra-node partitioning) this number constantly grows starting from about 6% for 16 nodes and ending at 14% for 256 nodes. This is a clear disadvantage of nonconforming solver with respect to the conforming one, as the conforming Nek5000 keeps the work imbalance at a level of one element. However, the original partitioning scheme is performed in the preprocessing step and cannot be directly applied to the dynamical grid modification.

On the other hand we found the two-level partitioning to be a quite efficient scheme, as the coarse grid solver setup time decreases with increasing node number (lower-left plot). Although its scaling is far from linear, it is a significant improvement with respect to the one-level partitioning (grid decomposition for all the cores performed in one step) that was giving a significant increase of setup time with increasing core count. The last plot shows the strong scaling of the nonconforming solver, and is almost identical with upper-right plot in

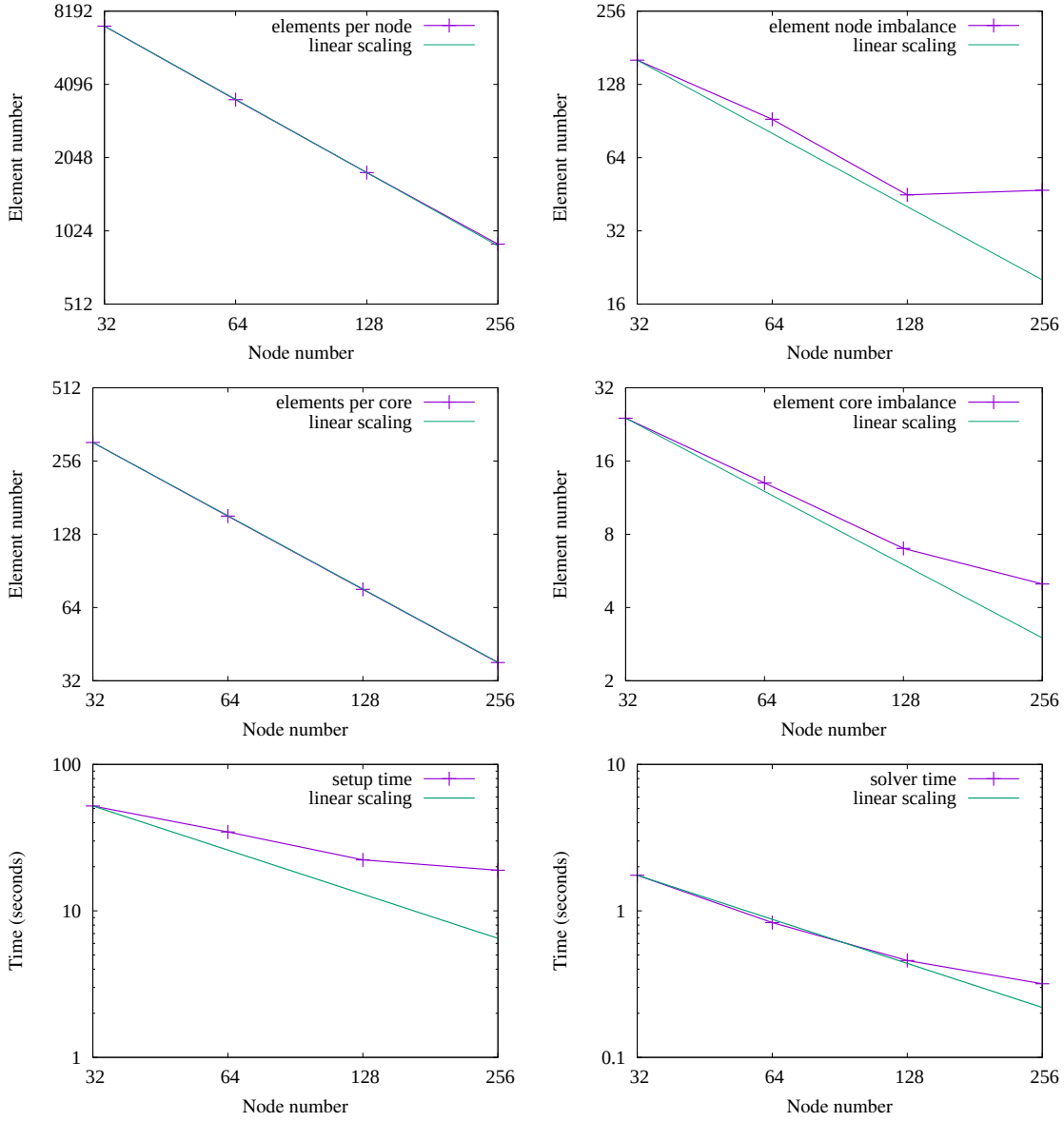


Figure 6: Parallel performance of nonconforming solver for the small test case executed on Hazel Hen. In this case each node consists of 24 cores. For plot description see caption in Figure 5.

Figure 5 in [9]. In both cases there is super-linear scaling between 32 and 256 nodes despite growing work imbalance for the nonconforming solver. We also reach the lower bound of the strong scaling limit at 256 nodes. This shows that parallel performance of nonconforming and conforming solvers is almost the same and proves the efficiency of our implementation. The parallel efficiency $E(N) = T_{serial}/(N * T_N)$ of the coarse grid solver setup and the nonconforming Nek5000 iteration at $N = 256$ nodes is equal 0.6 and 1.1 respectively. In the case of nonconforming Nek5000 we are still int the super-linear scaling region.

Results of tests performed on Hazel Hen are presented in Figures 6 and 7 for small and big test cases respectively. They are similar to the results obtained on Beskow, although the parallel efficiency of the solver differs between computers. The parallel efficiency of the coarse grid solver setup and the nonconforming Nek5000 iteration for the small test case on Hazel Hen at $N = 256$ nodes is equal 0.3 and 0.7 respectively. The difference is caused by the fact that, unlike Beskow, the number of cores per node on Hazel Hen is not a power of two, which affects performance of XXT. The other visible difference is a rapid increase of the node element imbalance for the small test case, that for 256 nodes reaches 5.2 % of the maximum number of elements per node. In the case of the bigger test we have only two points on the plot corresponding to 12288 and 24576 cores, that does not allow to draw any conclusion, as the number of degrees of freedom per core is much higher than expected strong scale limit. Unfortunately in this case the run performed on 2048 nodes exited due to disjoint graphs in the two-level partitioning scheme. We are currently working on improving grid decomposition to be able to perform simulations on 2048 and 4096 nodes.

2.2 Conclusions

The test cases presented in this section have shown that our AMR implementation to be correct and efficient. An important success is the fact that parallel performance of the conforming and nonconforming solvers is very similar, despite the increased complexity of the nonconforming solver. We have also shown that use of the AMR technique allows a significant reduction of simulation costs by reducing number of degrees of freedom. However, there is still room for improvement and we are going to continue our work on AMR implementation for Nek5000. As in the previous EU project CRESTA the main limitation of our solver comes from the use of external grid partitioner. Although in CRESTA the most important constraint was the strong scaling limit of ParMETIS, this time the critical aspect is quality of the graph partition, as the two-level partitioning would not accept disjoint graphs on the node's subdomain. We will investigate other options for graph partitioners. The other aspect is the stability of the solution, that is related to the nonconforming grid/conforming space approach, so we will implement mortar elements in Nek5000 as well. The last point is the oct-tree refinement that fixes element aspect ratio. We are going to investigate other grid managers, that would relax this constraint.

The code developed within ExaFLOW will become part of the next official Nek5000 release.

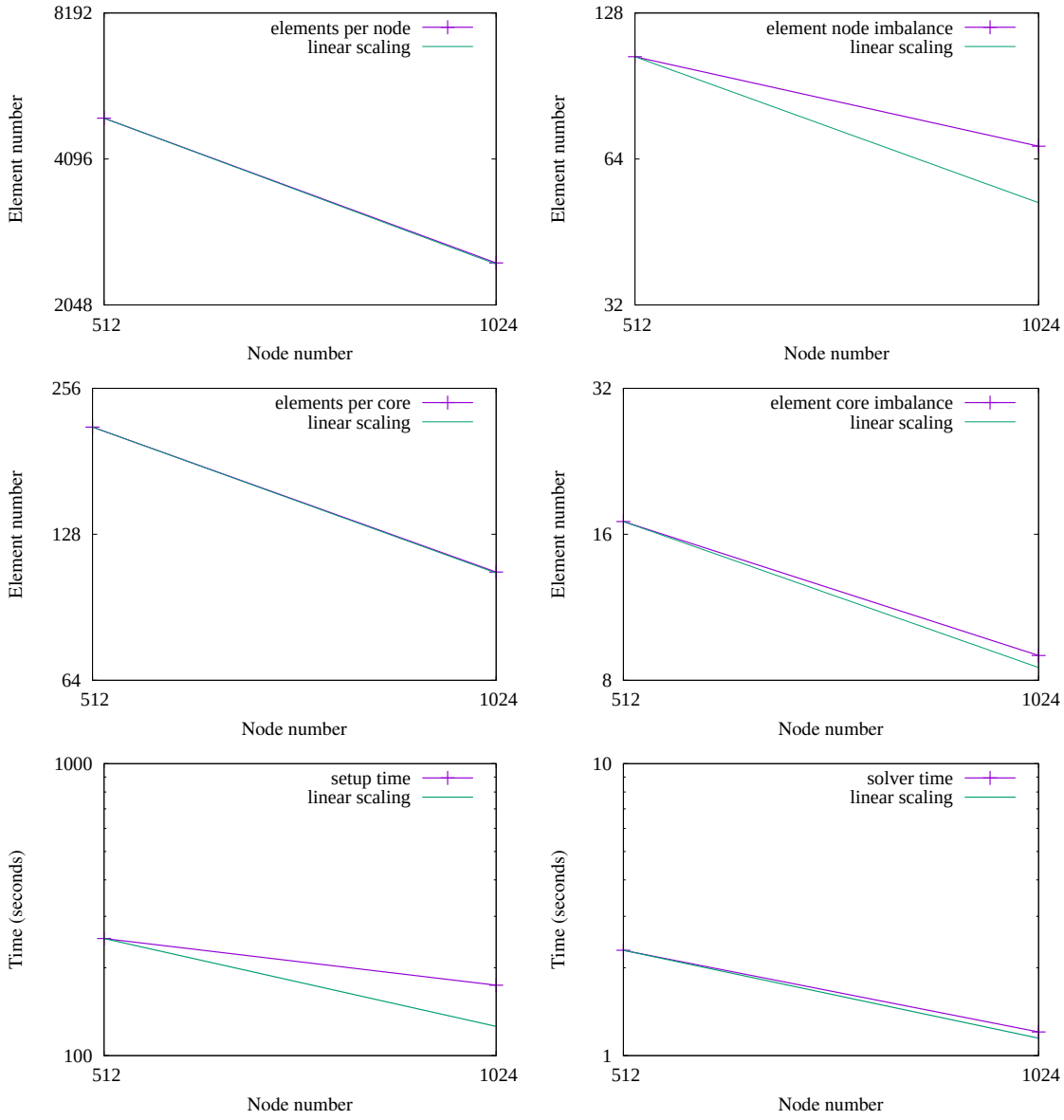


Figure 7: Parallel performance of nonconforming solver for the big test case executed on Hazel Hen. In this case each node consists of 24 cores. For plot description see caption in Figure 5.

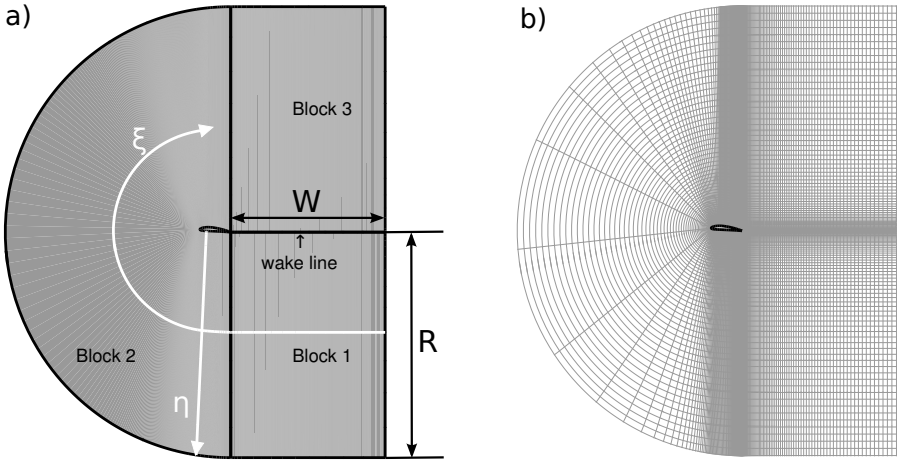


Figure 8: Computational domain arrangement for NACA4412 test case: (a) multi-block domain set up, (b) computational grid, showing only one in every 10 grid points.

3 Compressible flow over airfoil

The compressible flow over airfoils is considered as the flagship calculation by the University of Southampton. The rest of this section is organised as follows: First the improvements in runtime due to the algorithm developments in WP2 for the original NACA airfoil test case in deliverable 3.1 is presented. Then the calculations for a low-drag V2C airfoil at two different Reynolds numbers using one billion grid points are presented. This includes details of mesh generation, use of error indicators (documented in WP1) to refine the mesh and results. As discussed in deliverable 1.3, the low-drag compressible flow over V2C airfoil is chosen, as PRACE resources were awarded for a project aimed at understanding the buffeting phenomena for this airfoil, using meshes with up to 5 billion grid points.

3.1 Computational domain

The computational domain is composed of three blocks, as can be seen in figure 8(a). Block 2 is a C-type structured grid fitted around the airfoil surface; it interfaces with the structured blocks 1 and 3, which resolve the wake of the airfoil. Since Block 1 and Block 3 both contain the wake line, the wake line solution at each time step is obtained by averaging between the solutions obtained in the two blocks. This is because flow asymmetries near the airfoil trailing edge and/or small differences in initial conditions may cause the wake line solutions in the two blocks to diverge. For the NACA simulations the computational domain dimensions are $W = 5.0c$ and $R = 7.3c$ (see figure 8(a)), where c is the chord length. The total domain length is $12.3c$ and the total height is $14.6c$. For the V2C asimilar domain size is used, with $R = 7.5c$, total domain length $13.5c$ and the total height $15c$

A representation of the computational grid employed in the current numerical simulations is shown in figure 8(b), where only one in every ten grid points is plotted.

3.2 Perfomance improvement

The original test case presented in deliverable 3.1 is used to summarise the effects of algorithmic developments presented in WP2. As a reminder, the original test case considered is the compressible, subsonic flow over a NACA-4412 airfoil at 5° incidence, for a freestream Mach number $M_\infty = 0.4$ and Reynolds number based on the airfoil chord of $Re_c = 50000$. The airfoil geometry, which includes a sharp (zero thickness) trailing edge, was obtained by modifying the last coefficient in the 4-digit NACA airfoil equation (see equation 6.2 in [1]) from -0.1015 to -0.1036. This modification leads to minimal changes to the overall airfoil geometry.

Of the five different algorithms presented in WP2 deliverables 2.2 and 2.4 and published in [5], two different algorithms are considered in OpenSBLI. The base line (BL) algorithm, which follows the conventional programming approach and the performance of this is compared to the original SBLI code. The second one is a modified intermediate storage algorithm, in which only first derivatives of velocity components are stored (SS algorithm).

As OpenSBLI uses automatic source code-generation approach there are some differences compared to the static hand-written SBLI code. The similarities and differences of the two codes are described below.

Similarities: The numerical method used is fourth-order central difference scheme for the spatial derivatives. The treatment of one-sided derivative formulation near the boundaries follow the same way as the SBLI code, i.e. using Carpenter schemes for evaluating the derivatives. The time advancement scheme is the same. Following the approach of the SBLI code, the convective and viscous terms are evaluated separately, i.e. the contribution of the convective terms is evaluated first for each equation and then the viscous terms are evaluated. This reduces the memory footprint of the code since the work arrays used for evaluating derivatives in the convective terms are subsequently used to evaluate derivatives in the viscous terms. The number of derivative evaluations remains constant in both codes for the viscous terms.

Differences: The non-linear spatial derivatives in the equations are solved using a skew symmetric formulation in OpenSBLI and using entropy-splitting in the SBLI code. For this low-Mach number test case, OpenSBLI uses constant viscosity, whereas SBLI uses variable viscosity. As SBLI is hard coded for variable viscosity flows even a constant viscosity flow is solved using variable viscosity. Changing the governing equations based on flow-physics is one of the advantages of the code-generation techniques used in OpenSBLI.

The differences in the number of calls to derivative routines requiring storage in 3D arrays is summarised in table 1. With a different numerical method for the convection terms, the difference in the total number of derivative loops in SBLI and the BL algorithm of OpenSBLI is 15 for this test case. In the SS algorithm, only the first derivatives of velocity components (9) are stored, thus reducing the number of work arrays used by 33.

Code	x-conv	y-conv	z-conv	Entropy-split	viscous	other	**Total**
SBLI	5	5	5	15	35	0	**68**
OpenSBLI (BL)	20	20	13	0	0	33	**83**
OpenSBLI (SS)	0	0	0	0	0	9	**9**

Table 1: Comparison of the number of derivative evaluations requiring 3D storage arrays in SBLI and OpenSBLI (BL and SS algorithms) for a 3D airfoil problem with span-periodicity.

3.2.1 Grid arrangement

The grid designed for the validation case presented in deliverable 3.1 is used to evaluate the performance. The number of grid points employed per block for a 2D slice of the grid is given in table 2. The grid was designed to resolve all the flow features around the airfoil and the far field acoustic waves generated by the flow structures, for the current Reynolds and Mach numbers. This 2D grid is extruded in the z direction (wing span) using a constant grid spacing of $\Delta z = 0.002$. The number of grid points to be used along the span N_z is a user-specified parameter that can be used to modify the size of the numerical simulation. Since we impose Δz to be a constant, changing N_z means changing the spanwise extent of the computational domain.

Block	1	2	3
N_ξ	801	1799	801
N_η	692	692	692

Table 2: Number of grid points per block.

A 2D base flow is calculated to validate the implementation. Then the solution and the grid are extruded in the $z-$direction to compare the performance of algorithmms. All the simulations are performed using 3 nodes on the UK supercomputing facility (ARCHER) at EPCC. The allocation to processors in OpenSBLI can be varied by using automatic domain decomposition or by manually forcing the number of processors in particular directions. First the effect of the domain decomposition is studied for OpenSBLI using 5 grid points across the span.

Table 3 shows the runtime per iteration in seconds for different domain decompositions and algorithms. As seen, a domain decomposition in the ratio $12\times2\times1$ gives the best runtime for OpenSBLI BL algorithm and the same decomposition is used for both SS algorithm and SBLI code.

The runtime of the BL algorithm is similar to the SBLI code, while the SS algorithm gives a 30% reduction in runtime at this grid size. Table 4 shows the performance of the codes with 100 grid points across the span. Similar to the simulation with 5 grid points across span, the runtime of SBLI and BL algorithm of OpenSBLI are within 3% of each other and the SS algorithm reduces the runtime by 35% compared to the BL.

Note that all the runtimes are averaged over 5 runs and the difference between the runs is $2-5\%$. As presented in deliverable 2.4 (the performance of the different algorithms and

	Runtime (s) per iteration	**Processor decomposition**
OpenSBLI (BL)	2.3	$4 \times 3 \times 2$ Auto decomposition)
OpenSBLI (BL)	1.6	$6 \times 4 \times 1$ Manual forced z =1)
OpenSBLI (BL)	1.69	$12 \times 2 \times 1$ (Manual forced x=12, z=1
OpenSBLI (SS)	1.16	$12 \times 2 \times 1$ (Manual forced x=12, z =1)
SBLI	1.74	$12 \times 2 \times 1$

Table 3: Performance of SBLI and OpenSBLI for 3d Airfoil with 5 grid points across the span.

	Runtime (s) per iteration	**Processor decompositon**
SBLI	2.94	$12 \times 2 \times 10$
OpenSBLI (BL)	3.04	$12 \times 2 \times 10$
OpenSBLI (SS)	1.92	$12 \times 2 \times 10$

Table 4: Performance of SBLI and OpenSBLI for 3d Airfoil with 100 grid points across the span.

scalability on various architectures), these results hold good for the multi-block test case too. The different back-end optimisations such as tiling, hybrid MPI + OpenMP, CUDA, MPI+CUDA and others are inherently applicable to the multi-block test case as well. In summary it can be concluded that the code scales well up to 98,304 cores on CPU's on ARCHER (80% of the machine), 4000 K20 GPU's on TITAN and 64 Nvidia V100 GPU's on the Cambridge CSD3 machine.

3.3 Flagship calculations

The flagship calculations are performed for compressible flow over the low-drag V2C airfoil. DNS databases for airfoil simulations for moderate Reynolds numbers are currently very limited. The simulations for the flagship runs are aimed to extend the DNS database of fully-resolved transonic flow. The V2C profile is analysed at a fixed inclination of $\alpha = 4°$, considering a Mach number of M = 0.7 at Reynolds numbers 200,000, 500,000 and 800,000. This airfoil was the subject of experimental as well as numerical studies in the course of the TFAST project (part of EU's Horizon 2020 programme). Simulating such high Reynolds number flows require higher grid sizes, theoretically to run the airfoil at Reynolds number of 0.8 Million it requires 512 times the number of grid points compared to the original test case to resolve all the flow features. The simulations were initially carried out as part of an ARCHER Leadership Project. Following the award of HLRS access under a PRACE project (2018 - 2019), these simulations are currently being continued at larger scales on HazelHen,

with scaling tests and some of the early results shown here.

3.3.1 Grid generation

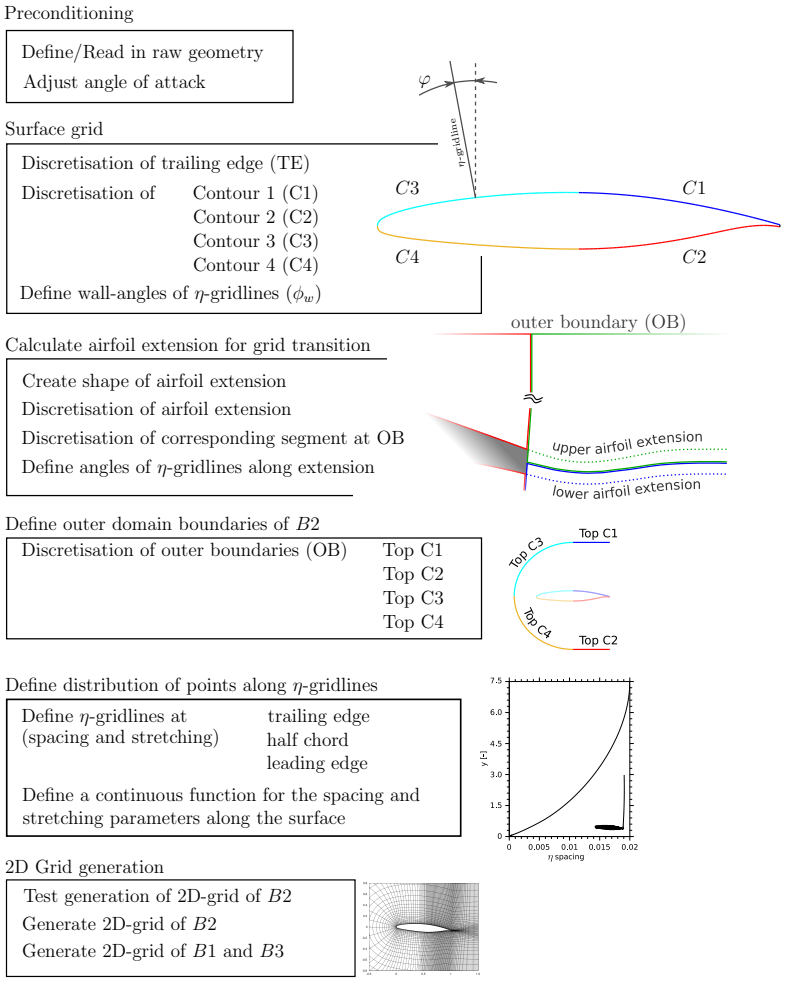


Figure 9: Flow chart outlining the grid-generation process. Image by [14].

The use of high-order central-difference methods for structured curvilinear grids, requires continuous metric terms up to the second order of derivatives or higher. Otherwise, artificial disturbances are introduced into the Navier-Stokes equations thus increasing the numerical error. To overcome the deficiencies of commercial grid generators, the group at University of Southampton has developed a new open-source grid solver. The detailed description of the grid generation was presented at ICCFD conference 2018 [14]. The grid-generation process is described in the following section. Figure 9 can be seen as a guideline of the work flow. It outlines a process consisting of six main steps. After setting the angle of attack and general

parameters regarding the 2D domain size, the airfoil surface and the grid around the trailing edge is defined. The domain boundaries are discretised in a next step, which is followed by the design of the η-gridlines. In a final step the full 2D grid is generated and written out.

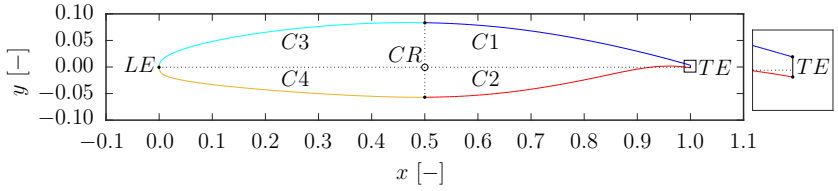


Figure 10: FlSegmentation of Dassault Aviation's V2C airfoil. Image by [14].

In order to increase the flexibility regarding the distribution of grid points, the airfoil is divided into four sections corresponding to figure 10 , labelled as C1, C2, C3 and C4. For the flow cases in [14], sections C1 and C2 needed to be highly resolved to capture the laminar/turbulent transition process, whereas sections C3 and C4 allow higher stretching due to the laminar boundary layer. The centre of rotation (CR) to adjust the angle of attack (α) is located at half chord, on the mean chord line (the horizontal dotted line in figure 10 with y = 0). The reference length is the axial chord length of the airfoil at zero inclination and independent of the angle of attack. After the airfoil is rotated by an angle of attack (the red contour in figure 11), it is vertically shifted, so that the leading edge (LE) again coincides with the abscissa of the Cartesian coordinate system (the green contour in figure 11).

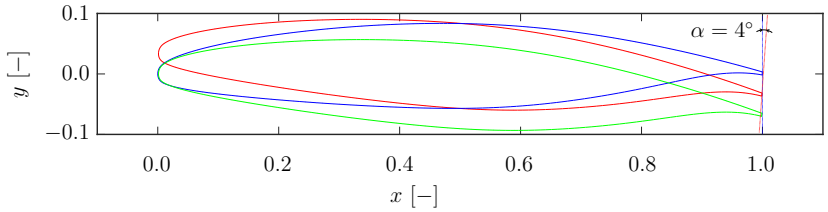


Figure 11: Transformation of the original profile (blue) to an airfoil with 4 angle of attack. The purely rotated profile is described by the red contour, whereas the green contour considers also a translation in order to have the origin at the leading edge. Image by [14].

A beta version of the code for generating the grids is available online in a Github repository (https://github.com/ZaunerM/PolyGridWizZ). The grid generation software in conjunction with the error indicators explained in WP1 deliverable 1.3 and published in [4], is

used for the flagship calculations.

A grid refinement approach was set up with a workflow consisting of the following four steps as presented in [15]:

1. Generation and iterative optimisation of the 2D grid. Error-indicator analysis is used to optimise the grid in the leading and trailing edge regions.

2. Running 3D simulations and analyse instantaneous representative xy- and xz-planes to identify critical regions.

3. Calculating averaged error-severity levels for a sufficient number of snapshots to confirm the criticality of identified regions. In the presented case, averages over 150 snapshots were sufficient.

4. Refining the grid in critical regions and repeat the process.

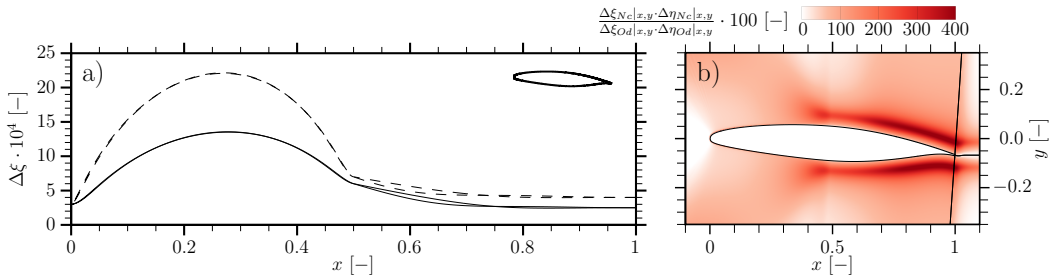


Figure 12: (a) Grid spacing in the ξ-direction as a function of x along the airfoil surface. Dashed lines correspond to orginal grid and solid lines to the refined grid. (b) Contour plot of the ratio between local cell area values of orignal and refined grids.

An overview of the refined regions is shown in figure 12, comparing the cell size of the refined grid. Figure 12(a) shows the grid spacing along the surface of the airfoil before and after the refining. Figure 12(b) shows the contour of the refinement, dark red regions indicate a significantly increased grid resolution after refinement.

The error indicator analysis of 2D simulation results captures critical regions at the leading and trailing edge. The leading edge is critical for the generation of an efficient grid, as the minimum Courant number is expected upstream of the stagnation point and limits the time step. Therefore, the error indicator is a useful tool for early stages in the grid-generation process, which can efficiently use 2D simulations. The trailing edge needs to be sufficiently resolved to avoid numerical instabilities near the singular points. Such a procedure helps in reducing superfluous grid points based on the airfoil geometry and angle of attack.

Tag	*Grid*	L_z	N_z	N_{total}	*Re*	*M*	α	*Runtime*	*Core hours*	*Remarks*
$C0$	*Nc*	0.05	150	$1.07 \cdot 10^9$	500,000	0.7	4°	25 *TU*	$3.6 \cdot 10^6$	reference case
$C1$	*Nc*	0.25	750	$5.35 \cdot 10^9$	500,000	0.7	4°	8 *TU*	$5.6 \cdot 10^6$	extended domain
$G1$	*Od*	0.05	150	$1.68 \cdot 10^9$	500,000	0.7	4°	8 *TU*	$1.2 \cdot 10^6$	refined $x-y$-plane
$G2$	*Nc*	0.05	250	$1.78 \cdot 10^9$	500,000	0.7	4°	8 *TU*	$1.7 \cdot 10^6$	refined in z-direction

Table 5: Overview of the different cases run on ARCHER and key parameters treated in this contribution.

3.3.2 Results

Some of the important flow physics obtained for the different Reynolds numbers are presented in this section. First the Reynolds number 500, 000 test case is presented and then the effects of Reynolds number and a wider domain case (run on HazelHen) are presented.

Reynolds number 500,000

The details of different simulations performed on ARCHER for this Reynolds number [16] is shown in table 5, the tags $C0$ and $C1$ correspond to the unrefined grids with two different spanwise domain extents, whereas the simulations $G1$ and $G2$ correspond to refined grids.

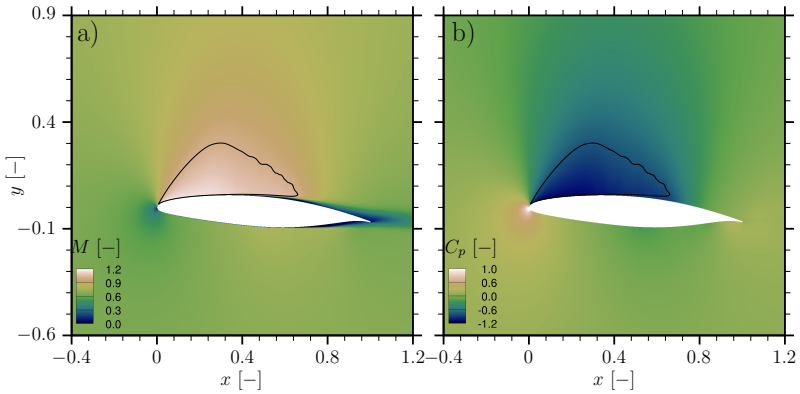


Figure 13: Contour plots showing time- and span-averaged Mach number (a) and pressure coefficient C_p (b) averaged over 25 time units. The black curve denotes the sonic line.

Figure 13 shows the local Mach number (a) and pressure coefficient (b) of the time- and span-averaged flowfield of case C0. The black sonic line in figure 13(a) bounds a region of supersonic flow covering 70% of the suction side. The acceleration of the flow causes a significant drop in pressure within the first quarter of the suction side of the airfoil (figure 13(a)). This is also observed in figure 14(a), showing the averaged pressure coefficient C_p as a function of the axial chord position. The dotted line denotes the critical pressure coefficient

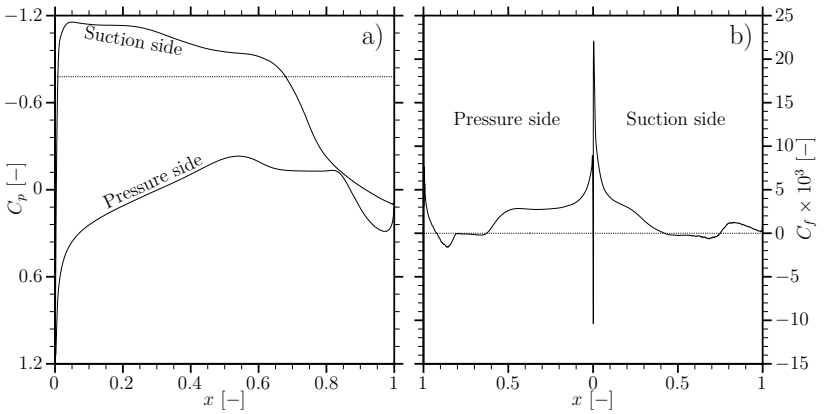


Figure 14: Time- and span-averaged pressure coefficient C_p (a) and skin-friction coefficient C_f (b) after 25 time units. The dotted lines indicate (a) the critical pressure coefficient of $C_{p.crit.} = 0.779$ and (b) the zero-line of C_f

to highlight again the supersonic region over the suction side. The steep drop of the upper C_f coincides with the flow recirculation that is indicated by the negative average skin-friction coefficient in figure 14(b). A regime of separated flow is also present on the pressure side near the trailing edge (TE).

The pseudo-Schlieren contour-plot in figure 17(a) shows the pressure gradient magnitude ($\|\nabla p\|$) of the instantaneous flowfield and illustrates the generation of acoustic waves (AW) by multiple sources in the aft-section on both sides of the airfoil. Significant emitters are the big turbulent vortices convecting downstream over the airfoil and into the wake.

The flow structures are also captured by plotting Q-criteria surfaces colored by vorticity in figure 17(b). The silhouette of 2D Kelvin-Helmholtz roll-ups persists also after transition, as seen in figure 17(a). Close to the surface, figure 17(b) shows also weak vortical structures that are convecting slowly upstream within the separated boundary layer.

Besides the standard-width case shown here (width $0.05c$) a wide domain case (with width $0.25cc$) has entered production runs on HazelHen. This case has 5 billion grid points and is the largest feasible calculation on current generation machines. Figure 18 shows a snapshot of the flowfield, showing some of the multi-scale effects that are captures, including span-coherent Kelvin-Helmholtz vortices, shockwaves (visible in the background grey-scale contours of streamwise pressure gradient), braid structures strained between the large rollers and structure down to scales of the order of the Kolmogorov micro-scale in the developing turbulent boundary layer downstream.

As part of a PRACE preparatory project, scaling data was obtained using up to 91,440 cores, whereas 30,480 cores were used for the calculation presented here. The code exhibits

near-perfect weak scaling (e.g. speed up factors of 1 ± 0.02, see figure 15). Figure 16 shows a strong scaling tests for a fixed problem size of 1.4 billion grid points showed a speed up of 2.866 when going from 15,240 cores to 48,768 cores, which is within 10% of the ideal speed up factor of 3.2. (As a side comment it is noted that node failures are not uncommon with this size of simulation, necessitating frequent check-pointing, and also the data requirements of these simulations have challenged the file system, with the incorporation of data compression a priority for future code development).

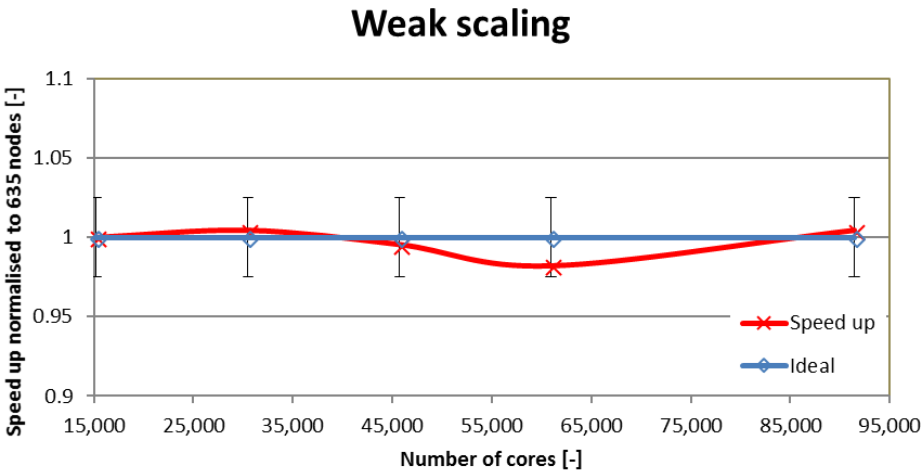


Figure 15: Weak scalability on HazelHen.

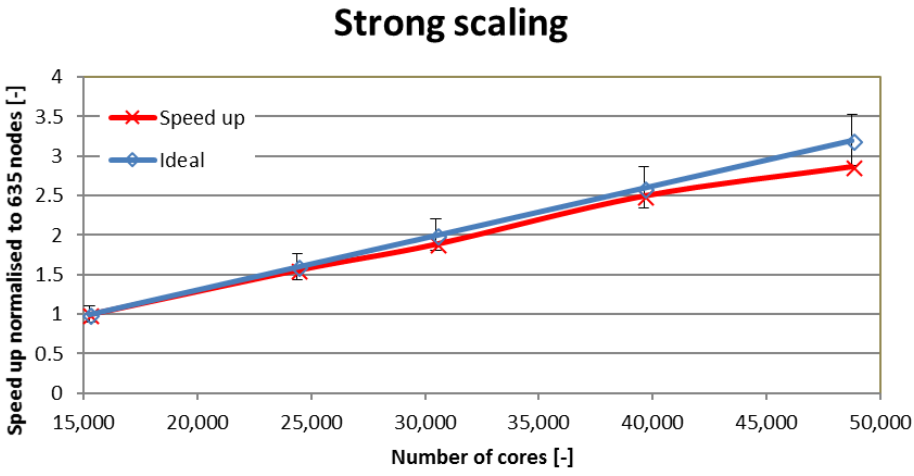


Figure 16: Strong scalability on HazelHen.

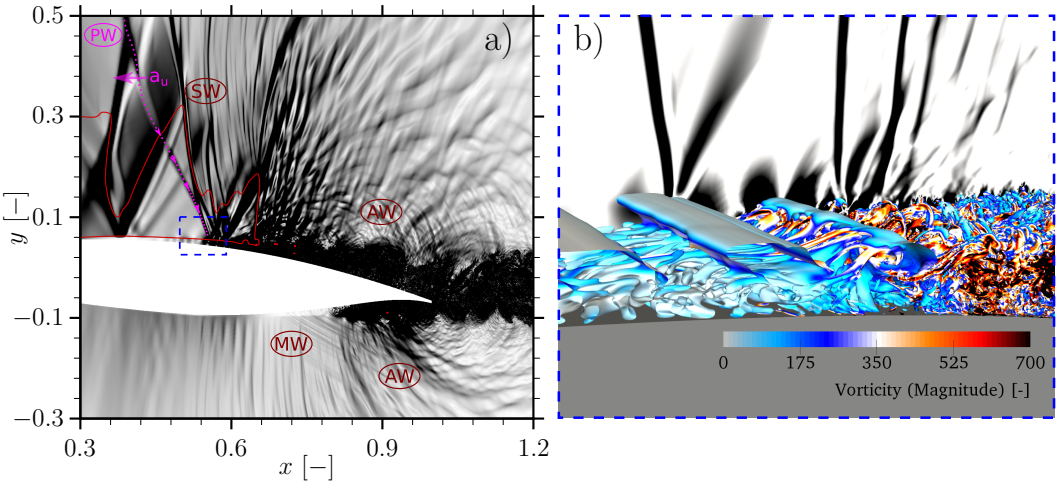


Figure 17: (a) Pseudo-Schlieren plot of the pressure gradient magnitude at t = 3.58, where strong waves ($\|\nabla p\| > 1.5$) show up black. The red curve denotes the sonic line. The region within the dashed blue rectangle is magnified in plot (b), showing Q-criteria surfaces (Q=10) of vortex structures coloured by vorticity magnitude at a different time step for $0.55 < x < 0.69$. In the background, strong pressure gradients ($\|\partial p/\partial x\| > 10$) are highlighted in black

Effect of Reynolds number

The Reynolds number of the above test case presented is increased to 0.8 Million to understand the flow physics and is run for 6 time units ($TU = c/U_{\text{inf}}$) that are defined as the ratio of chord length over free-stream velocity. A lower Reynolds number ($200,000$) simulation is also performed to uncover the Reynolds number dependency.

Figure 19 shows the Reynolds number dependancy on the wall pressure coefficient (C_p). Raising the Reynolds number to $Re = 800,000$ further increases the velocities over the upper surface. Compared to $Re = 500,000$, the wall pressure on the suction side mainly decreases around the half chord location (solid red line in figure 19). The C_p measured in the experiments ($Re = 2,850,000$) [11] is shown as symbols in the figure, and confirms the trend of decreasing minimum suction-side pressure around the half chord location.

3.4 Conclusions

The test cases have showcased some of the developments in ExaFLOW. The error analysis technique was used for the first time on a production run, greatly speeding up the grid design and increasing our level of confidence in the results. The potential for algorithmic improvements has been confirmed in large-scale tests, for example the 40% reduction of runtime for the 'Store Some' algorithm in OpenSBLI. These algorithmic changes were originally designed to investigate the potential on GPU-based supercomputers to improve efficiency by carrying out more floating point operations, whilst reducing memory transfers. What was surprising

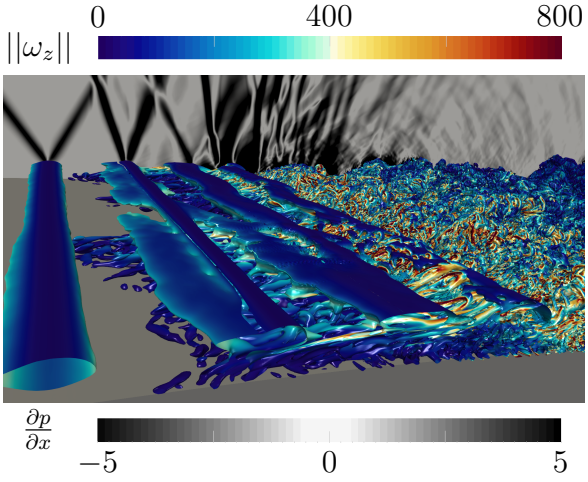


Figure 18: Snapshot of wide domain case showing Q-criteria surfaces (Q = 200) coloured by vorticity magnitude. Black regions in the background indicate high axial pressure gradients.

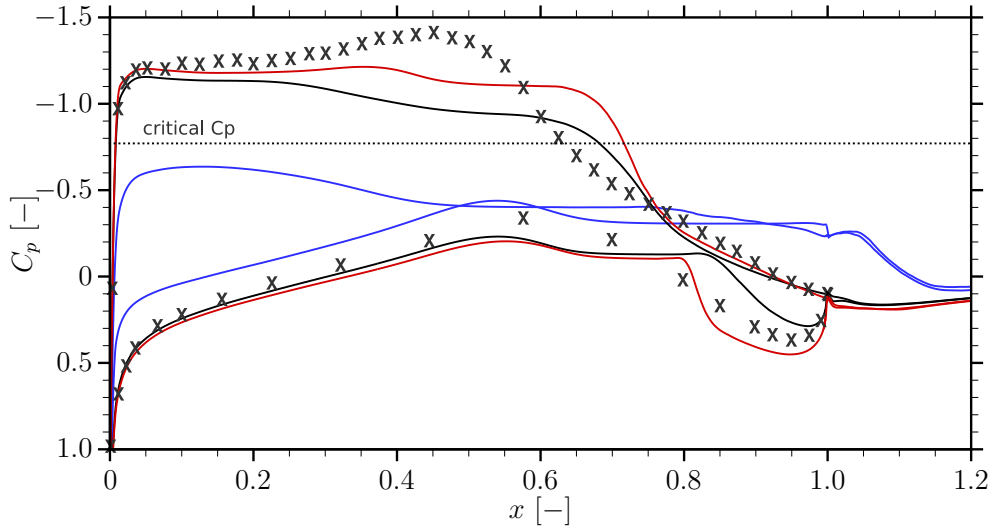


Figure 19: Time- and span-averaged pressure coefficient considering Reynolds numbers of Re = 200,000 (solid blue curve), Re = 500,000 (solid black curve) and Re = 800,000 (solid red curve). Symbols in the C_p -plot correspond to experimental investigations of [11] at Reynolds numbers of 2.85 million. The corresponding critical pressure coefficient is indicated by the dotted line.

was that the same algorithms that were good for GPU machines were also near-optimal for CPU machines. From the work in D2.4 this is also associated with a reduction in energy usage.

4 Turbulent flow around the Imperial Front Wing

External Aerodynamics is one of the key performance differentiators in Formula 1 and therefore one of the main focus areas for a racing team. Indeed a 1-2% improvement in aerodynamics performance can mean the difference between being first or tenth and so the requirements on aerodynamics modelling are very demanding. The front wing is a key component of the aerodynamics performance since it is responsible for generating load on the front axle as well as contributing to the control the front wheel wake.

An example of such front wing, based on the McLaren 17D race car, was investigated experimentally at Imperial College London by [10], and is referred in the following as the Imperial Front Wing. The current tools used to optimise this type of geometry would be, in ascending order of computational cost and accuracy, Reynolds Averaged Navier Stokes ($RANS$), unsteady $RANS$, Detached Eddy Simulation (DES), and wall modelled Large Eddy Simulation ($WMLES$), implicit (or explicit) Large Eddy Simulation – sometimes refereed to as under-resolved Direct Numerical Simulation (iLES/LES/uDNS) – and Direct Numerical Simulation (DNS).

In order to assess the technological readiness of one of the high order codes of the ExaFLOW consortium, $Nektar++$, we have made a comparison against the most accurate tool currently available to industrial engineers, Large Eddy Simulation. This study also helps us to assess the technical readiness of the codes since they have been executed by expert users within McLaren Racing Ltd. Wall resolving LES is not very commonly used by industry due to its high computational expense and a need to demonstrate the ability of these tools to replicate/replace experimental investigations. Currently the discretisation error or modelling assumptions implicitly made by these tools to represent the turbulence flow physics can limit their accuracy. Therefore the objective is two-fold: first to understand the relative expense of iLES/uDNS against commercial tools, and second to see if there is evidence of improvement in the ability to resolve detailed flow structures.

As mentioned, the investigation presented below is a comparison of the predicted flow around the McLaren 17D front wing using a state-of-the-art commercial finite volume LES (hereinafter FV LES) and the spectral/hp code $Nektar++$ against newly acquired time resolved PIV results under the ExaFLOW project. The test case for both wind tunnel experiments and numerical simulation consists of a three element front wing with endplates, footplates and canards, attached to a simplified nose-cone, as presented on Figure 20. The wind tunnel setup is shown on Figure 21.

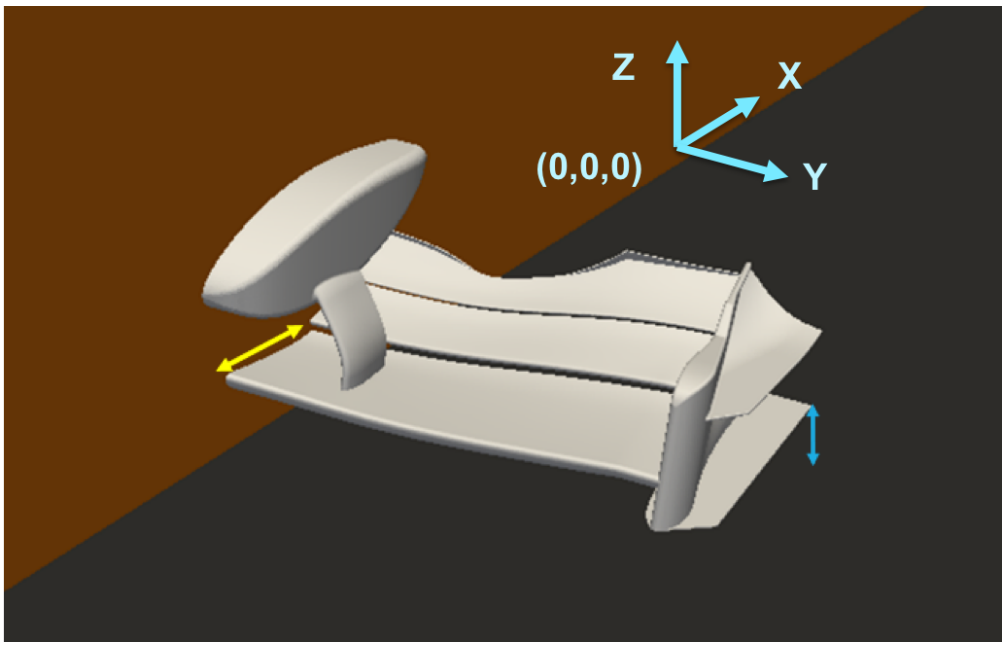


Figure 20: Imperial Front Wing geometry. Nomenclature: ride height h (light blue line), chord c (yellow line), main plane (dark orange), moving belt (dark grey)

Figure 21: Imperial Front Wing assembly on Imperial College London 10x5 Wind Tunnel.

The expected flow behaviour on the outboard section of the front wing, consisting in multiple co-rotating vortices, generating a 3D complex system is demonstrated in Figure 22.

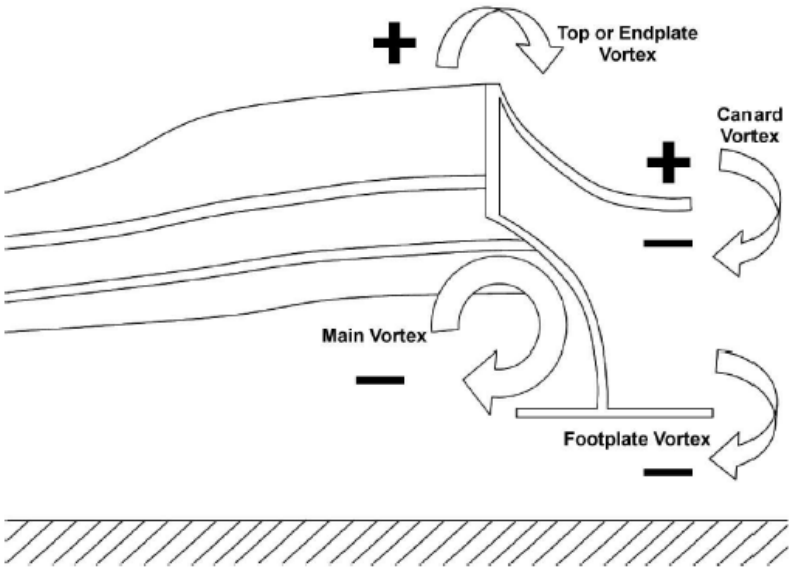


Figure 22: Topology of the complex vortex system downstream of the front wing. Courtesy of [10]

4.1 Methodology

Domain Parametrisation

In the following, X is the streamwise direction, Y the spanwise direction and Z the vertical direction. The trailing edge of the front wing is located -300 mm Full Scale in X direction and the ground plane at -25 mm Full Scale in Z direction.

To quantify the ground effect the wing is subjected to, the non-dimensional ratio between the ride-height h (the distance between the ground and the lowest part of the front wing endplate) and chord length of the first element c, h/c, is introduced. The lower h/c is, the highest the ground effect and hence the load on the wing.

Finally, the position of the wing in the domain is fully characterised by a pitch angle of 1.094° which will be kept constant for each h/c ratio. The configurations considered are the following:

- $h/c = 0.36$ without the wheel (low front ride height, high ground effect configuration)
- $h/c = 0.48$ without the wheel (high front ride height, low ground effect configuration)
- $h/c = 0.36$ with the wheel
- $h/c = 0.48$ with the wheel

Experimental Setup

The updated experiment was performed at the 10×5 Wind Tunnel in the Department of Aeronautics at Imperial College London. It is a closed return, closed twin test sections and temperature controlled facility. The lower test section is $20m$ long and its cross sectional area measures $3m$ x $1.5m$. The wind tunnel is equipped with a rolling road, traverse and full boundary layer control with flow uniformity around 1.0% and turbulence intensity <0.25% in the section used to test. The model tested is a 50% scale, as previous studies.

Surface Flow Patterns Measurements

A first point of comparison was the analysis of wall constrained streaklines on the wing, and to that end, flow visualization paint was applied on the wing in order to identify separation and transitional regions, at a reduced velocity of $15m/s$. Visualization results will be compared to wall shear stress contours from the numerical simulations.

The reason for that initial test was that [10] had to run his initial experiments with transition strip lines on the main plane, and indeed, the results shown in section 3.1 show clear evidences of transition lines and separation bubbles, followed by reattachment.

Given those initial results, in this study it was elected in the following to run the experiment without transition strips as the goal is to provide the most reproducible setup for numerical simulations.

PIV setup

Time resolved Particle Image Velocimetry (PIV) measurements were performed in the following iso-X planes, based on the coordinate system presented on Figure 20:

- $X = -294mm$;
- $X = -250mm$;
- $X = -150mm$;
- $X = 150mm$;

Additionally to the planes above, time resolved stereoscopic PIV was also performed for one extra plane:

- Stereo PIV plane: $X = -24mm$;

For the time-resolved PIV measurements, and for each plane considered, the total averaging time was 10 seconds, at an acquisition frequency of $250Hz$.

Due to the complexity of the experimental setup to obtain stereoscopic PIV, the first configurations considered were without the wheel, and those are the ones reported below as the simulations with the wheel could not be included here. We hope to be able to tackle those in subsequent runs. However, the bulk of the experimental data was performed on the no-wheel configuration.

Simulations Setup

Details for the computational simulations setup and mesh generation are presented and briefly explained in this section. In the following, we will refer as convective time units the non-dimensional time c/U.

Domain

As the computational model is performed in full scale, the size of the cross sectional area for the simulation was updated to $6m \times 3m$, in order to maintain the same blockage as the experiment.

Boundary Conditions

The Reynolds number calculated based on the chord length c of 0.25m and the free stream velocity U of $25m/s$ is $Re = 2.2 \times 10^5$ for both experiment and simulations. In terms of boundary conditions for the computational study, the following are considered:

- Both front wing and nosebox are set as wall with no-slip condition;
- Half of the model geometry was considered;
- Symmetry was imposed at $Y = 0$;
- Uniform velocity profile was imposed at the inlet;
- Outflow condition at the outlet;
- The ground is moving at speed U along X to mimic the moving ground;

Mesh Generation

FV LES

The computation grid is composed of a hexa-dominant mesh with an overall cell count of 181 million cells. In the far field, the characteristic mesh size is of 32mm, while around the front wing (FW) there are two refinement volumes with a characteristic cell size of 4 and 2 mm, respectively. A near wall refinement with prismatic cell layers was also applied around the FW and nose cone with the aim of capturing the boundary layer. Around the FW, 20 layers with a first cell height of 6.0×10^{-6} m and 1.2 geometrical growth rate were used, while for the nose cone, 15 layers with 1.0×10^{-5} m first cell height and 1.25 geometric growth rate were used.

Nektar++

In terms of mesh generation, the mesh for Nektar++ is first generated with a FV mesher using a conformal hybrid tetahedron/prism layout with a single macro prism layer of 4mm as shown on Figure 23. The mesh is overall much coarser than its FV LES counterpart aiming to find efficiency at higher polynomial orders.

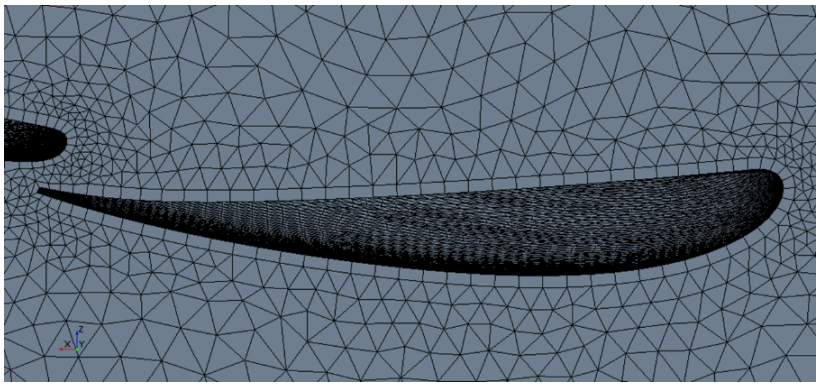


Figure 23: Detailed view of the macro prism layer.

The mesh is then processed through NekMesh, the Nektar++ high-order mesh generator, which is able to handle now complex geometries due to the ExaFLOW development on this tool. On Nekmesh the surface curvature is applied first, then the prism layer is split using an isoparametric splitting technique developed by [8] and illustrated on Figure 24, which guarantees valid elements.

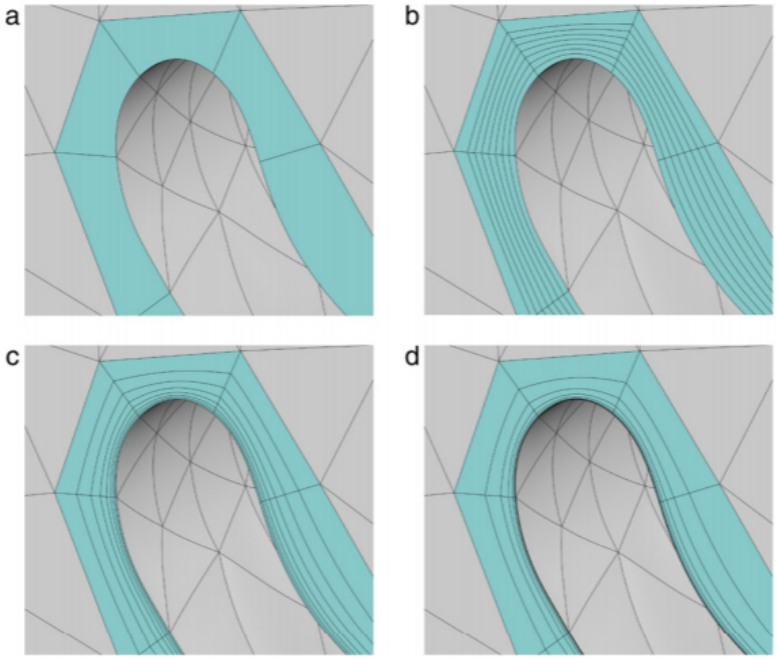


Figure 24: A sequence of meshes obtained by splitting macro-elements into n = 8 elements using a geometric progression and various values of r. (a) The macro-element mesh; (b) r = 1; (c) r = 3/2; (d) r = 2. Reproduction of [8]

The following setup is used to split the macro prism layer on both the front wing and on the floor: 8 layers and a geometrical growth rate r of 2. For all Nektar++ simulation cases, the high-order mesh considered is of sixth order. Smaller growth rates of 1.4 and 1.6 were also considered and offer small computational efficiency gains, but only the $r = 2$ cases will be presented here.

Simulation Cases

The Large Eddy Simulations used as benchmark for the following were performed using a commercial software available on the market, and they will be referred in the following as finite-volume LES ($FVLES$).

Experimental data will be compared with FV LES, as well as 3^{rd}, 4^{th} and 5^{th} order solutions using Nektar++, referred in the following as NM32, NM43 and NM54 - in which the two integers give the order of the velocity and pressure solution space respectively. For the FV LES, the simulation was averaged for 25 convective time units. All Nektar++ simulations were only averaged over one convective time unit, after the simulation had been run for eight convective time units with a resolution of NM32.

4.2 Simulation Description

The starting condition for both FV LES and Nektar++ simulations is a RANS input from a previous run, which is interpolated into the target meshes and used as an initial guess.

FV LES

For FV LES case, the simulation was performed using the Smagorinsky-Lilly subgrid scale model with a subgrid viscosity coefficient of 0.1 without any near wall treatment.

In terms of solver settings, the simulation was run with an implicit incompressible solver, using a segregated approach, which is the closest match to the splitting scheme used in Nektar++. The system of ODE is solved using an Algebraic-multi-grid solver.

In terms of numerical scheme, the time integration used a second order implicit scheme with non-dimensional time step of 1.25×10^{-4}, while for the convective scheme the third-order $MUSCL$ scheme was used blended with a 15% bounded central-differencing contribution. The Green-Gauss scheme has been used for the gradient calculation. The simulation was run for a total of 27 convective lengths and averaged over 25.

Nektar++

The setup scheme for Nektar++ for the Incompressible Navier-Stokes solver considers a velocity correction scheme solver, combined with a mixed continuous and discontinuous Galerkin Projection due to the use of the sub-stepping.

In terms of stabilization techniques, both *Spectral Dealiasing* and *Spectral Vanish Viscosity - SVV*, using the *DGKernel* setting are applied to the solution, which are the newly implemented stabilization techniques for high Reynolds Numbers, we present on a industrial case for the first time. The parameters used for all cases are the same, except for the time step, quadrature points and solver tolerances which are tuned at run level to keep the CFL under control. We are also currently evaluating the new HDF5 input formatting, developed under ExaFLOW which has a significant influence on the parallel partitioning and start up time. The setting is detailed on figure 25.

```
<CONDITIONS>

    <SOLVERINFO>
    <I PROPERTY="SolverType"     VALUE="VelocityCorrectionScheme" />
        <I PROPERTY="EQTYPE"         VALUE="UnsteadyNavierStokes" />
        <I PROPERTY="AdvectionForm"  VALUE="Convective" />
        <I PROPERTY="GLOBALSYSSOLN"  VALUE="IterativeStaticCond" />
        <I PROPERTY="SuccessiveRHS"  VALUE="30" />
        <I PROPERTY="SpectralhpDealiasing"  VALUE="True" />
        <I PROPERTY="PRECONDITIONER" VALUE="FullLinearSpaceWithDiagonal" />
        <I PROPERTY="WeightPartitions" VALUE="Boundary" />
        <I PROPERTY="SpectralVanishingViscosity" VALUE="DGKernel" />
        <I PROPERTY="Projection"           VALUE="Mixed_CG_Discontinuous" />
        <I PROPERTY="TimeIntegrationMethod" VALUE="BDFImplicitOrder1" />
        <I PROPERTY="SubStepIntScheme"      VALUE="RungeKutta2_ImprovedEuler" />
        <I PROPERTY="Extrapolation"         VALUE="SubStepping" />
    </SOLVERINFO>

    <PARAMETERS>
        <P> OutputEvery                 = 1000  </P>
        <P> TimeStep                    = 1.0e-5 </P>
        <P> NumSteps                    = 0.25/TimeStep  </P>
        <P> IO_CheckSteps               = OutputEvery    </P>
        <P> IO_InfoSteps                = 10   </P>
        <P> IO_CFLSteps                 = 10   </P>
        <P> Linf                        = 0.25 </P>
        <P> Uinf                        = 1.00   </P>
        <P> Winf                        = 0.00  </P>
        <P> Re                          = 2.2e5 </P>
        <P> Kinvis                      = Uinf*Linf/Re  </P>
        <P> SVVDiffCoeff                = 0.1</P>
        <P> SubstepCFL                  = 0.8 </P>
</PARAMETERS>
```

Figure 25: Solver and Parameters settings for Nektar++ simulations.

To achieve the polynomial order sweep reported below, the first simulation was started from the RANS initial field and ran for 8 convective time units at NM32. The solution obtained was then used as a starting point for both the 4^{th} and 5^{th} order simulations, which both ran for only one extra convective time unit due to limited resources. The results for the 9^{th} convective length for the 3 high-order polynomial expansion cases are then used in the comparison.

A sampling window of a single convective length is not sufficient to achieve statistically converged results, and barely enough to wash out the transient from one order to the next. On-going efforts are currently being made to help address this point.

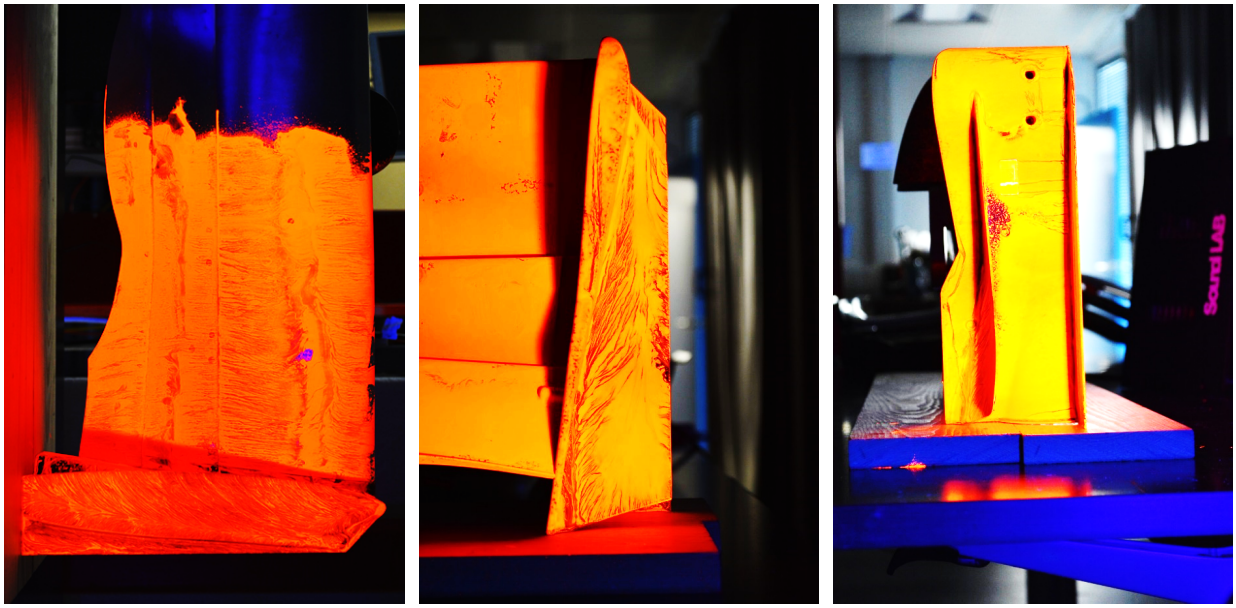
4.3 Results

4.3.1 Flow Visualization

First results present are the wall constrained streaklines over the wing, comparing experimental images with CFD simulations.

Experimental Results

Flow visualization image test result for Imperial Front Wing in bottom, top and side views are shown in Figures 26a, 26b and 26c, considering reduced velocity of $15m/s$.



(a) Bottom view. (b) Top view. (c) Side view.

Figure 26: Imperial Front Wing tested with velocity of $15m/s$.

Simulation Results

In Figure 27 and 28 the flow visualization comparative results from the simulations with Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion and FV LES cases are compared with experimental results.

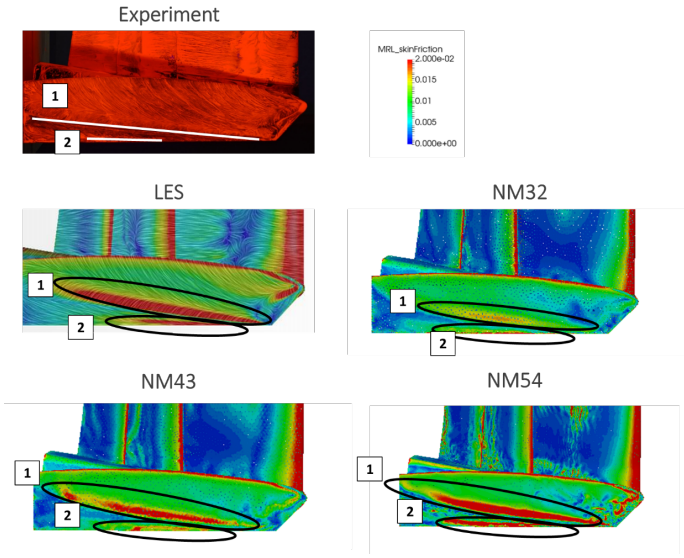


Figure 27: Flow visualization results from experiment (top) compared with LES contour and shear stress lines and Nektar++ contour for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for bottom view.

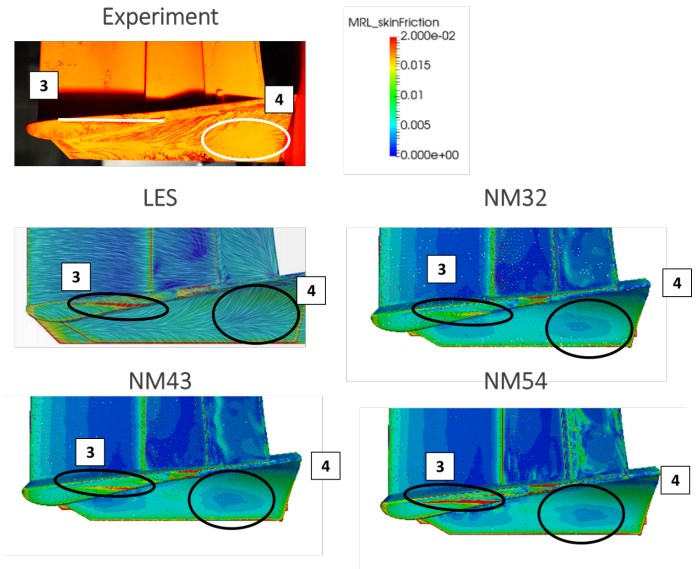


Figure 28: Flow visualization results from experiment (top) compared with LES contour and shear stress lines and Nektar++ contour for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for top view.

Figure 27 shows similar attachment lines on the footplate indicating that the vortical structure positioning is consistent. The transition bubble in the main plane is well captured by FV LES. NM32 capture its position correctly but the reduction in wall shear stress

downstream indicates that the resolution is perhaps insufficient in the near wall region. This was to be expected given the relative coarseness of the mesh and one would expect this situation to improve at higher polynomial orders. Indeed, NM43 shows that the situation is evolving and we capture with better accuracy the near wall behaviour by increasing the order of the solution. Similar to the previous case, NM54 shows clear improvements in terms of capturing the correct structures highlighted (1 and 2) compared to previous lower order cases, enhancing the polynomial order jump benefits to correct predict the flow behaviour. Figure 28 show a consistent prediction on the top edge and canard attachment lines for the two types of simulations. Both NM43 and NM54 have better wall shear stress after the separation lines, especially on the vane and flap, showing that increasing the polynomial order has a positive effect on the quality of the prediction.

4.3.2 PIV Planes

In this section, the PIV results on several planes at different downstream locations are compared against the corresponding contours plots obtained from the numerical simulations.

Plane $X = -294mm$ Full Scale

This first plane is the closest to the wing endplate, and was indeed positioned at the smallest possible distance experimentally to avoid laser reflection effects.

The non dimensionalised averaged velocity contours are presented for V_y (horizontal) and V_z (vertical) on Figures 29 and 30, respectively.

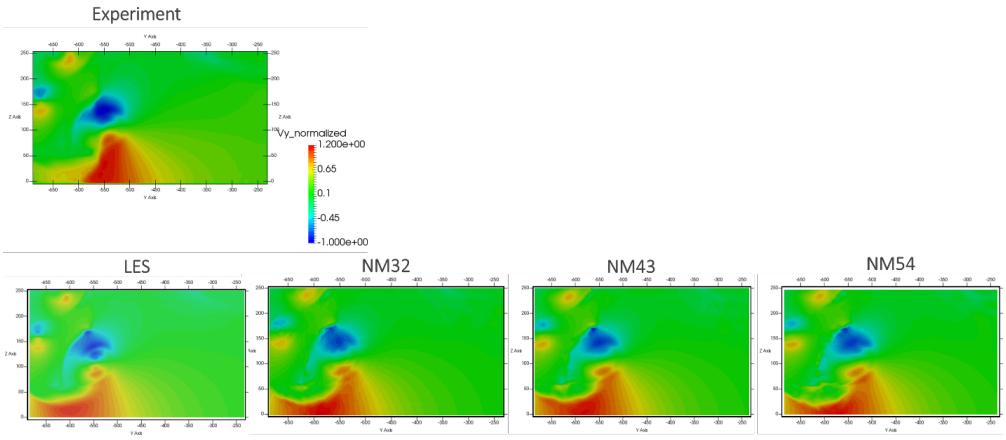


Figure 29: Average normalized V_y velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -294 mm.

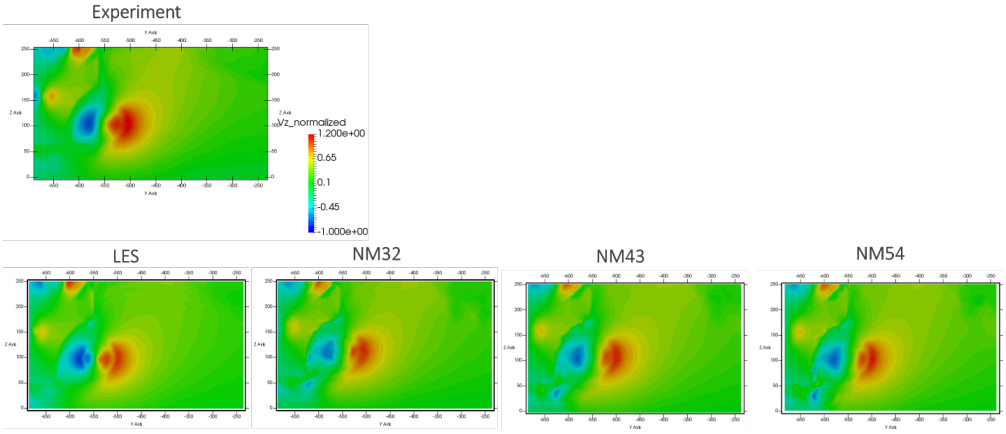


Figure 30: Average normalized V_z velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -294 mm.

The main structure is the effect of the merging of the main and footplate vortices, the top left one is the top edge vortex and the middle left one the canard vortex. It can be seen from Figure 29 that despite a correct positioning, both simulation under-predict the strength of the main vortex, and perhaps show a delayed merging with the footplate vortex. At this stage, all simulation cases presented good agreement with experimental results, although this is not the most sensitive plane.

Plane $X = -250mm$ Full Scale

Moving downstream, the top edge and canard vortices are better predicted by Nektar++, and this is where we would expect to benefit from the lower numerical dissipation. However, we will obviously not recover from the initial lack of strength. The prediction of the main vortex structures are well captured with all simulations. For the Nektar++ cases, we observe how structures contours provide better matches against the experiment as the polynomial order increases, specially for NM54, although the structures are not fully developed due to small averaging time interval.

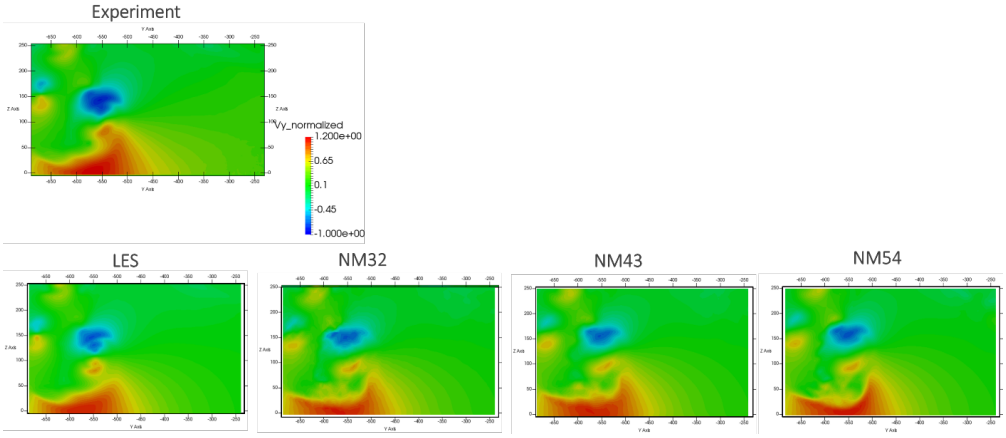


Figure 31: Average normalized V_y velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -250 mm.

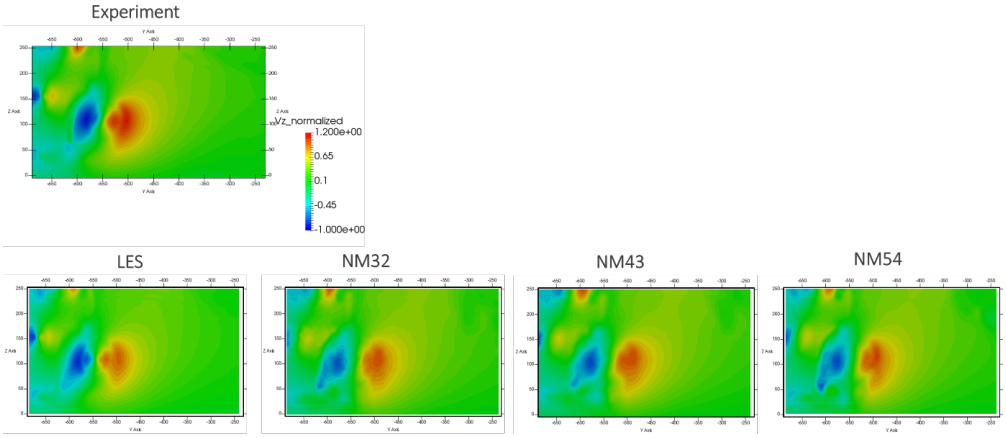


Figure 32: Average normalized V_z velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -250 mm.

Plane $X = -150mm$ **Full Scale**

Moving further down, the FV LES is the best match, due to the longer temporal evolution of the vortex core and averaging period of this simulation case. We expect to improve results for Nektar++ in terms of both shape and quantities once we increase the simulation time and have similar averaging as the FV LES. Despite this fact, Nektar++ cases are able to predict the main flow features.

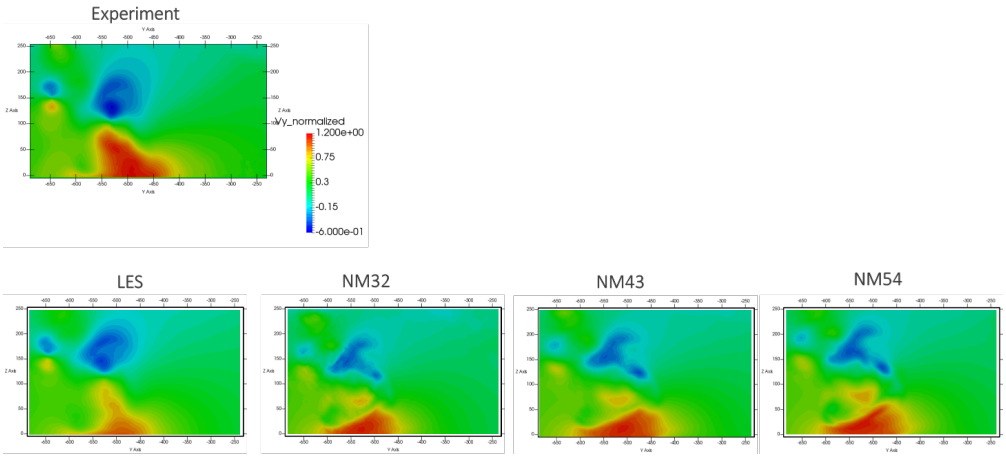


Figure 33: Average normalized V_y velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -150 mm.

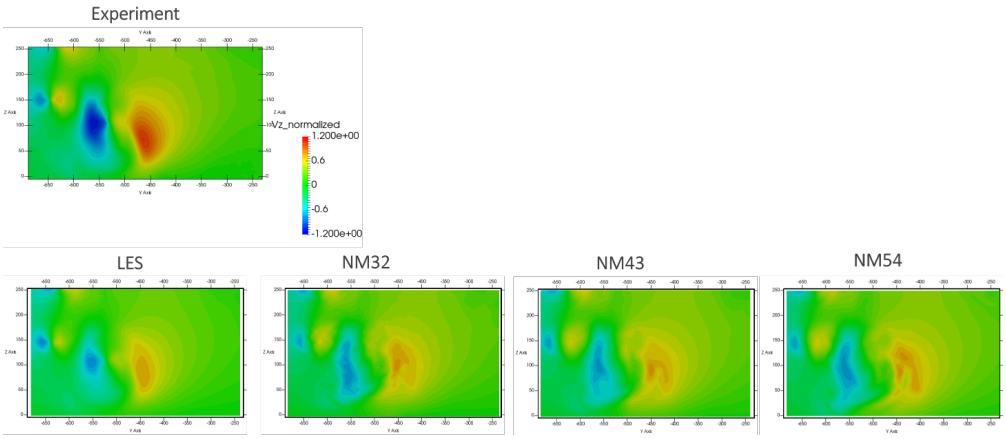


Figure 34: Average normalized V_z velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -150 mm.

In terms of vertical velocity, the decay due to numerical dissipation seems smaller in Nektar++, but the overall shape of the main vortex is distorted.

Plane $X = 150mm$ Full Scale

We are now more than one chord length from the wing, and it can clearly be seen that the vortices are stronger in Nektar++, generally in better agreement. The footplate vortex being ingested by the main vortex is also better preserved in Nektar++. We would conjecture that the numerical benefits of the high order scheme are starting to appear here, as there is no

pressure gradient or external effects which could explain this slower decay. In particular this highlights the usefulness of higher order methods within the study of F1 aerodynamics, where vortices are advected sometimes along the full length of the car.

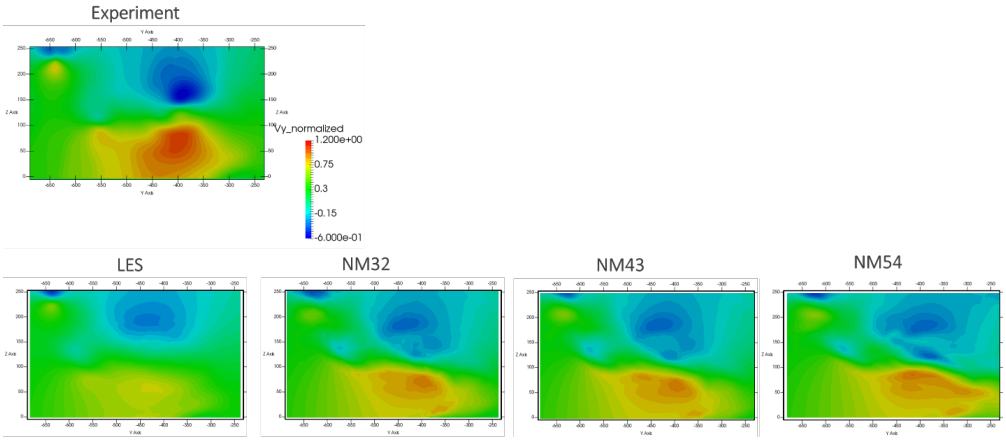


Figure 35: Average normalized V_y velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = 150 mm.

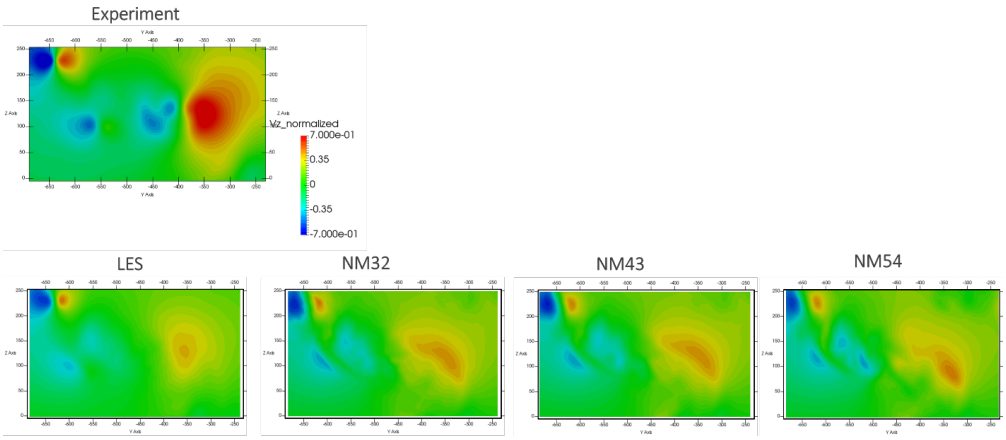


Figure 36: Average normalized V_z velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = 150 mm.

Stereo PIV - Plane x = -24 mm Full Scale

Finally, a stereoscopic PIV plane is presented. From figures 37 and 38, we see once more that as the flow advances, the diffusion effect on the FV LES starts to increase. Nektar++ is able to predict the flow structures with better accuracy in terms of capturing similar contours

to the experimental results. As the polynomial order increases, we believe that it will be possible to observe better agreement in terms of contour scales.

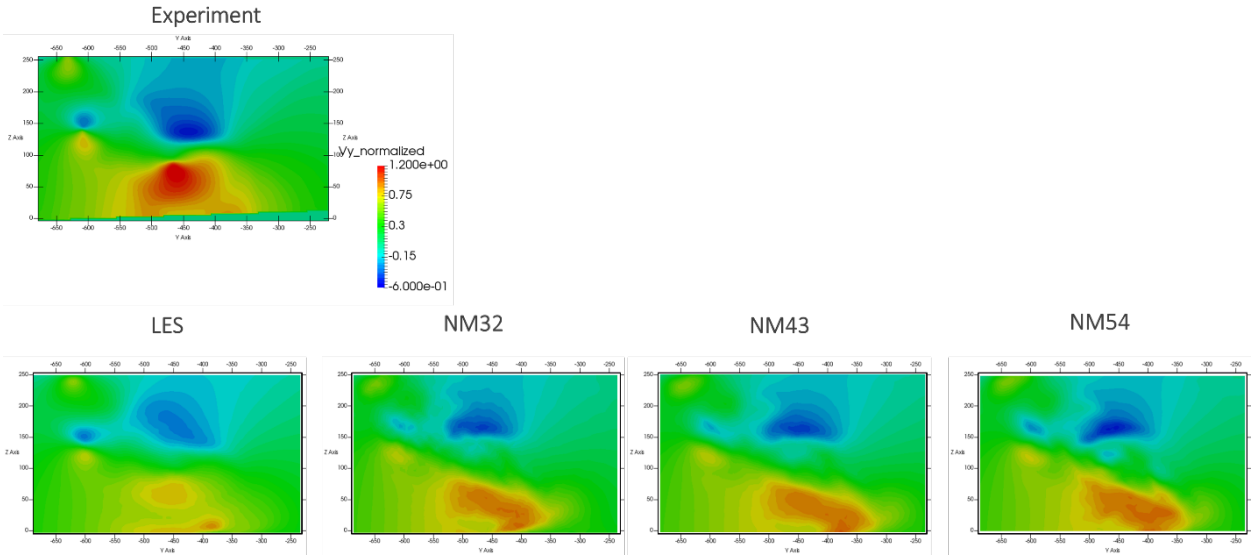


Figure 37: Average normalized Vy velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -24 mm.

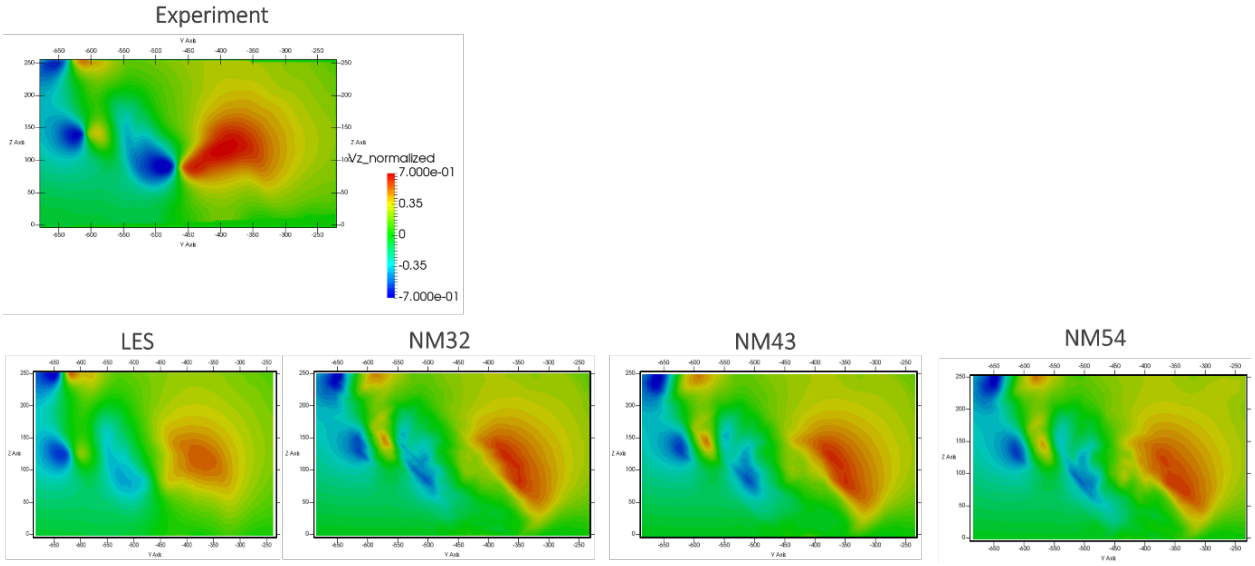


Figure 38: Average normalized Vz velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -24 mm.

The reconstruction of the axial component on figure 20 warrants more investigation as there is no clear physical reason for the acceleration seen on the right hand of the field, and is only presented for reference. Since it was the only plane for which the reconstruction could be performed, it is somewhat difficult to judge its validity and we will discard from the analysis for now.

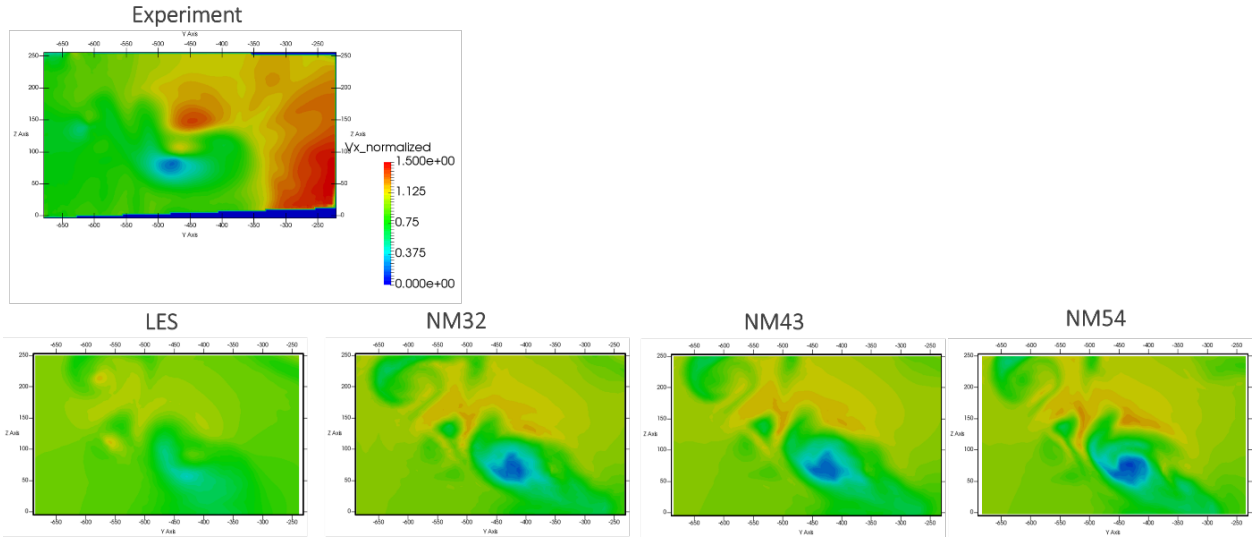


Figure 39: Average normalized Vx velocity results from experiment (top) compared with LES, Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion for plane x = -24 mm.

4.3.3 Convergence of the front wing load

To investigate the convergence of the FW load with both time and polynomial order in Nektar++ in a more quantifiable manner, the pressure distribution profile

To that end, an additional study was conducted considering 25% reduction on the macro boundary layer, changing the number of layers to seven and growth rate value of 1.6 to better cluster the resolution near the wall.

Figure 40 shows the evolution of the pressure coefficient distribution (averaged over one convective time unit) from the eigth to the ninth convective time unit at $Y = -550mmFS$, close to the endplate, showing that the solution is still slowly evolving towards a more loaded wing, which in turn will shed stronger vortices. Those simulations will be extended for up to twenty convective time units, similar to the FV LES cases.

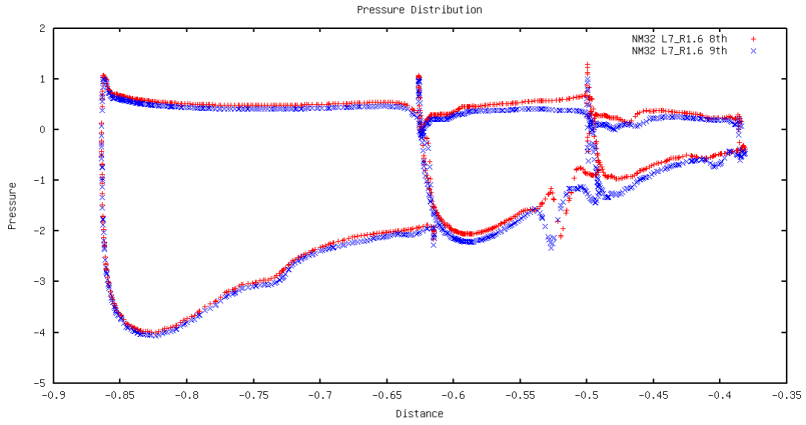


Figure 40: Pressure distribution from the 8^{th} to the 9^{th} convective time unit for the plane $Y = -550mm$ on the front wing elements

Figure 41 shows the effect of increasing the polynomial order on the pressure coefficient distribution at this same location $y = -550mmFS$ averaged over one convective time unit. Again, the load increases with the polynomial order, and we will thus have to wait for the NM32 to settle before increasing the order again.

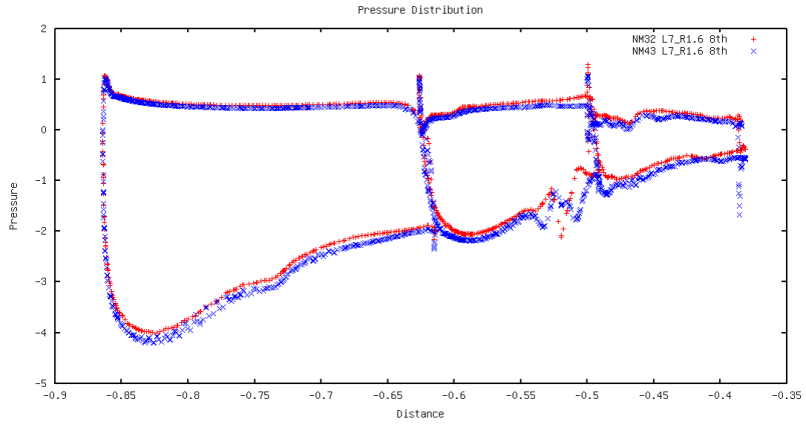


Figure 41: Time averaged pressure coefficient distribution at $Y = -550mm$ on the front wing elements, comparing 3^{rd} (NM32) and $4^{th}(NM43)$ polynomial order solutions

Finally, Figure 42 shows the effect of increased polynomial order on iso-surfaces of total pressure coefficient on the same mesh, demonstrating the reduction of the wake close to the centreline underneath the nose cone as well as the improved vortex definition downstream.

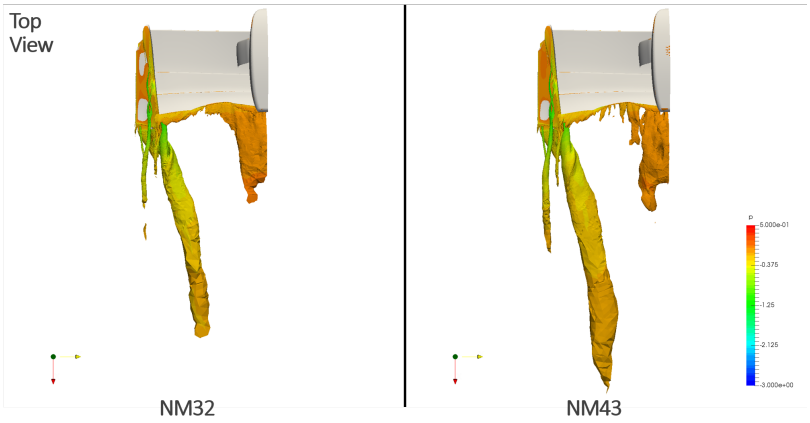


Figure 42: Comparison between Iso-surfaces of time averaged (over one convective time unit) total pressure coefficient value of 0.0 for 3^{rd} (NM32 on the left) and 4^{th} (NM43 on the right) order simulations

4.4 Run Time

A summary of the simulation times for one convective units is found on table 6. At this point, for Nektar++ simulations, a conservative setup in terms of stability and precision was adopted, aiming to match the CFL conditions and correct flow solution, in order to understand and attain high-fidelity of the solution. There is a potential enhancement in terms of improving the computational time, which is already being applied without losing the accuracy and will improve the simulation time. Such approaches are the convective formulation, optimal quadrature ordering and consideration of variable polynomial order.

Table 6: Simulation time for one convective length using Nektar++, considering 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) order polynomial expansion and LES

Load Case	CPUs	Time per Iteration (s)	Time Step	Length	CPU hours (h)	Total Wall Time (h)
NM32	2048	14	5e-5	0.25	38912	19
NM43	4096	13	1e-5	0.25	368640	90
NM54	2048	19	5e-6	0.25	540672	264
LES	4096	50	1.25e-4	0.25	114688	28

However, we can conclude from table 1 that the Nektar++ NM32 case is cheaper compared to FV LES and going from FV LES at $2^{nd}/3^{rd}$ order to 5^{th} order is around five times more expensive. Also, high polynomial runs are primarily influenced by reduced time step, so greater development on optimisation of semi-implicit formulation may help reduce this cost for those runs.

4.5 Conclusions

The study has demonstrated the use of one of our academic tools, Nektar++, on an industrial test case by technical experts within our industrial partners McLaren. The initial tests presented here show that we have a good agreement on the flow topology on the front wing endplate, and subsequently good agreement on the resulting secondary structures when moving downstream. However the main vortex is slightly under-predicted in both FV LES and in Nektar++ simulations, which come from an under-prediction of the front wing load. On-going simulations with a finer boundary layer may help demonstrate if we have the correct near wall resolution in our setup, and the simulations reported here have been extended as the time averaged solution was shown to be still evolving.

Next Steps

In order to improve correlation, it is necessary to run the simulations for longer and let the effect of the increased load on the front wing propagate downstream.

Once we obtain better agreement on the averaged quantities, the second order moments will be investigated and compared to experimental values. One example of the RMS field for the horizontal velocity at $x = -24mm$ is presented in Figure 43.

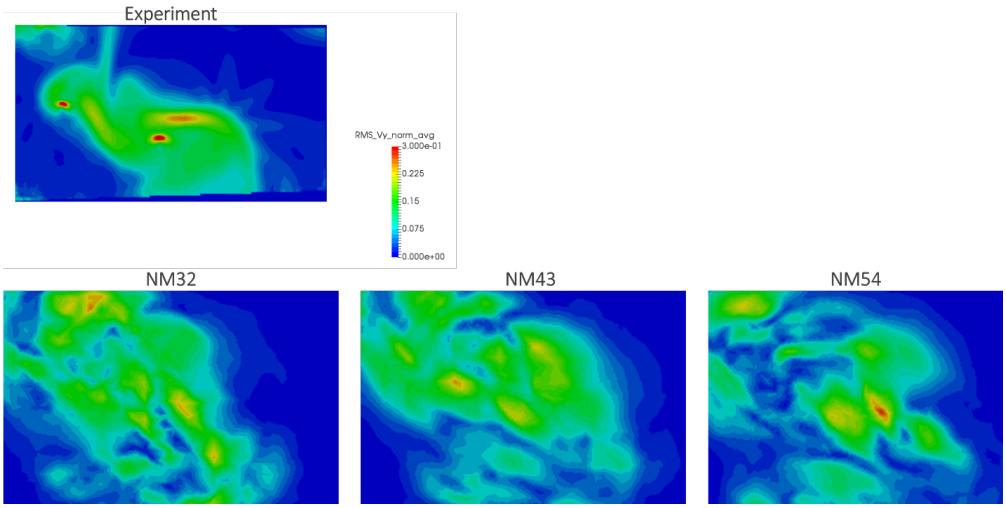


Figure 43: Normalized RMS contour for V_y comparing Experiment and Nektar++ for 3^{rd} (NM32), 4^{th} (NM43) and 5^{th} (NM54) polynomial order

5 Overall Conclusions

This deliverable has presented the flagship calculations designed to showcase the benefits brought about by the ExaFLOW algorithmic developments. Improvements in terms of scaling, run-time reduction, and energy efficiency have been achieved, all of which are important issues surrounding exascale computing. Highlights include:

1. Efficient implementation of AMR by KTH with the parallel performance of the conforming and nonconforming solvers is very similar despite the increased complexity of the nonconforming solver.

2. Using AMR gives 77% reduction in the number of elements used for the wing simulation compared to a simulation without AMR

3. Using oct-tree refinement fixes element aspect ratio for AMR simulations.

4. Improvements of 40% reduction in runtime are obtained using low memory code generated automatically with OpenSBLI by SOTON.

5. An error estimator, coupled with the multi-block grid generator greatly aided the set-up of the largest simulations with SBLI on HazelHen.

6. We have demonstrated Nektar++ application on an industrial test case which has highlighted the potential for greater off body resolution which is relevant to the demanding aerodynamics performance in F1.

7. These simulations have required developments in high Reynolds number stabilisation, improvement in high order meshing pipeline and improvement in parallel input/setup.

References

[1] I.H. Abbott and A.E. von Doenhoff. *Theory of wing sections*. Dover Publications, 2^{nd} edition, 1959.

[2] S. Dong, G. E. Karniadakis, and C. Chryssostomidis. A robust and accurate outflow boundary condition for incompressible flow simulations on severely-truncated unbounded domains. *J. Comput. Phys.*, 261:83–105, 2014.

[3] P. F. Fischer, J. W. Lottes, and S. G. Kerkemeier. NEK5000: Open source spectral element CFD solver. *Available at: `http://nek5000.mcs.anl.gov`*, 2008.

[4] C. T. Jacobs, M. Zauner, N. De Tullio, S. P Jammy, D. J. Lusher, and N. D. Sandham. An error indicator for finite difference methods using spectral techniques with application to aerofoil simulation. *Computers & Fluids*, 168:67 – 72, 2018.

[5] S. P. Jammy, C. T. Jacobs, and N. D. Sandham. Performance evaluation of explicit finite difference algorithms with varying amounts of computational and memory intensity. *Journal of Computational Science*, In Press, 2016. doi: 10.1016/j.jocs.2016.10.015.

[6] F. R. Menter. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA J.*, 32:1598–1605, 1994.

[7] Ketan Mittal and Paul Fischer. Mesh smoothing for the spectral element method. *Journal of Scientific Computing*, Aug 2018. ISSN 1573-7691. doi: 10.1007/s10915-018-0812-9. URL `https://doi.org/10.1007/s10915-018-0812-9`.

[8] D Moxey, MD Green, SJ Sherwin, and J Peiró. An isoparametric approach to high-order curvilinear boundary-layer meshing. *Computer Methods in Applied Mechanics and Engineering*, 283:636–650, 2015.

[9] Nicolas Offermans, O. Marin, M. Schanen, Jing Gong, P. Fischer, and Philipp Schlatter. On the strong scaling of the spectral element solver nek5000 on petascale systems. In *Proceedings of the 2016 Exascale Applications and Software Conference (EASC2016) : April 25-29 2016, Stockholm, Sweden*, ACM International Conference Proceeding Series, 2016. ISBN 9781450341226. doi: 10.1145/2938615.2938617. QC 20170814.

[10] Jonathan Mark Pegrum. *Experimental study of the vortex system generated by a Formula 1 front wing*. PhD thesis, Imperial College London, 2007.

[11] R. Placek, M. Miller, and P. Ruchala. The Roughness Position Influence on Laminar Aerofoil Aerodynamic Characteristic in Transonic Flow Regime. In *30th Congress of the International Council of the Aeronautical Sciences 2016, pp. 26*, 2018.

[12] P. Schlatter and R. Örlü. Turbulent boundary layers at moderate Reynolds numbers: inflow length and tripping effects. *J. Fluid Mech.*, 710:5–34, 2012.

[13] R. Vinuesa, P. S. Negi, M. Atzori, A. Hanifi, D. S. Henningson, and P. Schlatter. Turbulent boundary layers around wing sections up to $re_c = 1,000,000$. *Int. J. Heat Fluid Flow*, 72:86–99, 2018.

[14] M. Zauner and N. D. Sandham. Multiblock structured grids for direct numerical simulations of transonic wing sections. In *Tenth International Conference on Computational Fluid Dynamics (ICCFD10), Barcelona,Spain, July 9-13*, 2018.

[15] M. Zauner, C.T Jacobs, and N. D. Sandham. Grid Refinement Using Spectral Error Indicators With Application To Airfoil DNS. In *6th European Conference on Computational Mechanics (ECCM 6) 7th European Conference on Computational Fluid Dynamics (ECFD 7), Glasgow,UK, June 11-15*, 2018.

[16] M. Zauner, N. De Tullio, and N. D. Sandham. Direct numerical simulations of transonic flow around an airfoil at moderate Reynolds numbers. In *AIAA conference*, 6.2018-2911.