upscale

Upscaling **P**roduct development **S**imulation **C**apabilities exploiting **A**rtificial inte**L**ligence for **E**lectrified vehicles

D 1.1 The Potential for Finite Volume solution acceleration in the context of Machine Learning

Authors

Jianbo Huang – Engys

Eugene de Villiers – Engys

Patrick Bangert - Algorithmica

Month Year

12/2019

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement Nº 824306.

Project Details

PROJECT TITLE	Upscaling product development simulation capabilities exploiting artificial intelligence for electrified vehicles
PROJECT ACRONYM	UPSCALE
GRANT AGREEMENT NUMBER	824306
INSTRUMENT	RIA
CALL	LC-GV-2018
STARTING DATE OF THE PROJECT	November, 1ST 2018
PROJECT DURATION	42 Months

The UPSCALE Project

The UPSCALE (Upscaling Product development Simulation Capabilities exploiting Artificial inteLligence for Electrified vehicles) goal is demonstrating the feasibility of using AI enhanced CAE methods in EV development processes, such as vehicle aerodynamics, battery thermal modelling and crash simulation and leading the deployment of AI tools for other CAE applications. UPSCALE is the first EU-project that has the specific goal to integrate artificial intelligence (AI) methods directly into traditional physics-based Computer Aided Engineering (CAE)-software and –methods. These CAE-tools are currently being used to develop road transportation not only in Europe but worldwide. The current focus of the project is to apply AI-methods to reduce the development time and increase the performance of electric vehicles (EVs) which are required by the automotive industry to reduce global emission levels. High performance computing (HPC) and CAE-software and –methods play a decisive role in vehicle development process. In order to make a significant impact on the development process, the two most HPC intensive CAE-applications have been chosen as use cases for the project: vehicle aero/thermal- and crash modelling. When considering total automotive HPC usage, approximately 20% is used for aero/thermal simulations and up to 50% of HPC resources are utilized for crash simulations. By improving the effectiveness of these two areas, great increases in efficiency will lead to a 20% reduction of product time to market. Other novel modelling approaches such as reduced order modelling will be coupled to the AI improved CAE-software and -methods to further reduce simulation time and ease the application of optimization tools needed to

improve product quality. Through the combined effort of universities, research laboratories, European automotive OEMs, software companies and an AI-SME specialized in machine learning (ML), the UPSCALE project will provide a unique and effective environment to produce novel AI-based CAE-software solutions to improve European automotive competitiveness.

The UPSCALE Consortium

PARTICIPANT Nº	PARTICIPANT ORGANISATION NAME	COUNTRY
1 (Coordinator)	**IDIADA AUTOMOTIVE TECHNOLOGY SA (IDIADA),**	Spain
2	VOLVO PERSONVAGNAR AB (Volvo Cars)	Sweden
3	VOLKSWAGEN AG (VW)	Germany
4	CENTRO RICERCHE FIAT SCPA (CRF)	Italy
5	ESI GROUP (ESI GROUP)	France
6	ENGYS LTD (ENGYS LTD)	United Kingdom
7	Kompetenzzentrum - Das Virtuelle Fahrzeug, Forschungsgesellschaft mbH (VIF)	Austria
8	VRIJE UNIVERSITEIT BRUSSEL (VUB)	Belgium
9	ECOLE NATIONALE SUPERIEURE D'ARTS ET METIERS (ENSAM PARISTECH)	France
10	ALGORITHMICA TECHNOLOGIES GMBH (ALGORITHMICA)	Germany
11	F INICIATIVAS I MAS D MAS I SL (F-INICIATIVAS)	Spain

Document Details

DELIVERABLE TYPE	Report
DELIVERABLE Nº	1.1
DELIVERABLE TITLE	The Potential for Finite Volume solution acceleration in the context of Machine Learning
NAME OF LEAD PARTNERS FOR THIS DELIVERABLE	ENGYS
VERSION	1
CONTRACTUAL DELIVERY DATE	M8
ACTUAL DELIVERY DATE	M14
DISSEMINATION LEVEL	Public

Revision History

The following table describes the main changes done in the document since it was created

REVISION	DATE	DESCRIPTION	AUTHOR (ORGANIZATION)
V.0	16/12/2019	First complete version of deliverable	Jianbo Huang (ENGYS)
V.0	30/12/2019	Reviewed	Enric Aramburu (IDIADA)

Disclaimer

Contents

Contents 5
1. Executive Summary 6
1.1. Attainment of objectives and deviations 8
2. ConvNet-based pressure projection method 9
2.1. Pressure projection method using machine learning 9
2.2. Analysis of the unsupervised machine learning approach 11
2.3. Fully supervised approach for acceleration of aerothermal simulations around vehicles 12
2.3.1. Description of the AI model 12
2.3.2. Model accuracy assessment of the fully supervised ConvNet architecture 13
2.3.3. Suitability assessment of the fully supervised ConvNet architecture 15
3. N-iteration-advancing fully supervised approach for acceleration of aerothermal simulations around vehicles 17
3.1. Model accuracy of the N-iteration advancing AI architecture 18
3.2. A use case for the N-iteration advancing AI architecture 18
3.3. Analysis of potential speed up of convergence for the N-iteration- advancing AI architecture 20
4. Smart mapping technology for speeding up aerothermal simulations 23
4.1. Background 23
4.2. Outline of the ‘smart mapping’ procedure 24
4.3. Outline of AI-based object-recognition method in the ‘smart mapping’ procedure 24
4.4. Accuracy of the ‘smart mapping’ approach 25
4.4.1. Map of flow fields around an aerofoil 25
4.5. Map of flow fields around two-dimensional square boxes 26
4.6. Map of flow fields around two-dimensional cars 27
5. Momentum-continuity coupled solver approach 31
5.1. Theoretical background of the momentum-continuity coupled solver approach 31
5.2. Comparison of results between the segregated and the coupled solver 32
6. Conclusions 35
6.1. Recommendations for future work 36
6.2. Risk assessment 36
7. Acknowledgements 38
8. References 39

1. Executive Summary

This report describes the work performed in Task 1.1.1 It is a part of work package 1, focusing on assessment of methodologies in accelerating aerothermal simulations of flow fields around cars. The main goal of this subject of research is to speed up the process, and to reduce the HPC-resource requirements for providing training dataset samples to be used in developing AI-based predictive models.

The main goal of the UPSCALE project is to develop AI-based predictive models for aerothermal simulation and crash simulation of electric vehicles. In order to achieve this, a large number of CFD simulations will be needed to generate training datasets, which are necessary for developing such AI-based predictive models.

CFD simulation of aerothermal fields around vehicles is a time-consuming process with a heavy demand on HPC resources. It is therefore of great interest to speed up the simulation, without sacrificing accuracy. This will not only shorten the development cycle of the AI-based models and software, but also significantly reduce the energy consumption of the CFD simulation process.

The objective of Task 1.1.1 is to identify or, if needed, to develop the potentially effective methodology for speeding up aerothermal CFD simulations around vehicles, and further to assess the effectiveness, as well as shortcomings of each method. Although the immediate beneficiary of such methodology is the UPSCALE project itself, such novel methods have far-reaching applications in the field of CFD and CAE.

After a comprehensive search, we found that, there are several potentially effective ways to speed-up the simulation process, they can be classified in four board categories:

1. AI-based method to speed up pressure projection process, which is typically reduced to solving a Poisson equation for the pressure field.
2. AI-based method to shorten the iterative process of the CFD simulation, which means using less iteration numbers to achieve the same accuracy.
3. Provide smart, near converging initial field, based on previously performed simulations around similar but not identical vehicles, so that much less iterations may be needed to achieve convergence of the simulation process.
4. Apply novel, more implicit solvers for the linearized large equation sets, which are derived from discretizing the governing momentum and continuity equations using finite-volume method.

Comprehensive efforts have been made into the above-mentioned four areas. The sections followed will describe our research work into each of the above areas. To summarize, we found that:

- The AI-based pressure projection method, as proposed by Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, Ken Perlin (1), offers an effective way of solving the Poisson equation for pressure projection by a ConvNet-based machine-learning model.
- We verified that, as stated in the authors paper, this method can lead to about 20% reduction in the simulation time, for the simulation cases they carried out. However, this method is aimed at solving inviscid, incompressible Euler equation. It will need

substantial work to re-implement the methodology they proposed, in the context of viscous, turbulent flow fields around vehicles.

- AI-based method to shorten the iterative process of the CFD simulation: we extended the method proposed by Tompson et al, to use supervised training methodology. While the original proposal is to replace the PCG solver for discretised continuity equation, we extended it by using AI-based predictive model to replace both momentum and continuity solver.
- In our extended approach, the training is constructed in such a way that, the input data are the pressure and velocity fields which correspond to a particular iteration. The target data are pressure and velocity fields a number of iterations (say 100) later. Models trained in this way are then capable of predicting flow field N iterations later, based on input data of the current iteration. The advantage of this method is that it can advance many iterations in the simulation process, thus significantly shortens the convergence time. However, since AI-prediction cannot be 100% match the target value, this method can introduce errors in both pressure and velocity fields, meaning that the continuity and momentum equations are not satisfied to the same level of a CFD solver. The error (noise field) needs to be filtered out by the CFD solver, after each AI predictions. We found that with this approach, a speed up in simulation by 30% to 50% can be achieved.
- An AI-based smart mapping method was developed by ENGYS. The technology and ideas derived directly from efforts of WP1 and WP2 to create automated workflows for machine learning data generation. With this approach, simulation of flow around a new geometrical configuration, such as a new design of a car, can be started from a converged flow field around a similar but not identical geometry. The computational grids of the two flow fields need not be the same. This approach uses an AI technique to find the closest flow configuration from a database of similar simulations, so that the differences of the new and the existing one in the database is minimized. Thus, the simulation has the potential to converge much quicker than with commonly used potential field initialization or point-to-point mapping. This approach can lead to large speed-ups (potentially in excess of 100%) compared to traditional flow initialization procedures.
- Compared to the first two AI-based simulation acceleration approaches, the smart mapping approach has the advantage of not needing previous training datasets. Since training is not required (or more accurately is performed on the fly) smart-mapping can be included into any training process without additional up-front cost. As long as there is an existing converged solution, it can be used for a new simulation to restart from. It is therefore particular useful for fast evaluation into the effect of small variation of design on the shape of a vehicle, and for generating training dataset for flows around a large number of similar but not identical shape designs.
- We identify that the coupled-solver technology, developed at ENGYS, provides another way of speeding up the CFD simulation. The solver couples the discretised momentum and pressure equations, so that the solution converges much faster. Typical speed up of 3 times can be achieved with the couple-solver approach. Another advantage of this approach is that it does not need any training dataset to achieve a significant performance gain.
- While the coupled solver approach is not strictly based on machine learning, it is a robust technology that provides significant synergies in the context of UPSCALE and will allow us to significantly exceed the original 20% speed up goal in CFD simulation.

We also anticipate that the coupled solver methodology will be compatible with all other ML based efforts.

1.1. Attainment of objectives and deviations

This deliverable constitutes a report on the viability or machine learning for accelerating finite volume CFD calculations. A broad range of methods that have the potential of fulfilling this goal have been investigated, including the explicitly specified “FluidNet” approach from Thompson et al. (1). In a technical sense therefore, all indicated objectives have been met. In terms of delivery timeline, staffing issues have resulted in a delay with respect to the delivery date relative to that originally specified. As mentioned in the original change request, we do not anticipate that this delay will materially impact the rest of the project.

2. ConvNet-based pressure projection method

This section discusses the methodology proposed by Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann and Ken Perlin (1). Their method uses a convolutional neural network (a.k.a. FluidNet) to train a model for solving the Poisson equation to predict the pressure field, using the velocity as input. The divergence-free condition is used to construct the objective function for the training process to minimize. No target pressure field is provided, so the training is unsupervised. The theoretical basis of the method is described in section 2.1. Some basic simulation results are presented that verify the claim made in the original paper. The extension technique of this method to cover the application of aerothermal flow simulations is also given in this section.

2.1. Pressure projection method using machine learning

This sub-section describes the theoretical basis of the AI-based methodology for solving Poisson equation to get the pressure field, as proposed in (1)

The basic idea of Tompson et al.'s proposed method is to solve a linear equation system using a data-driven method, in particular, using a ConvNet-based deep-learning architecture. The target equation they want to solve is the pressure projection equation for inviscid, incompressible flow with a buoyancy force. The momentum equation for such a flow problem takes the following form:

$$\frac{\partial \boldsymbol{u}}{\partial t} = -\boldsymbol{u} \cdot \nabla \boldsymbol{u} - \frac{1}{\rho}\nabla p + \boldsymbol{f} \qquad (1)$$

where $\boldsymbol{f}$ is the body force. In the case they investigated, it is the buoyancy force. Equation (1) is subjected to the divergence-free constraint for the velocity field at any point in the computational domain:

$$\nabla \cdot \boldsymbol{u} = 0 \qquad (2)$$

Using an operator-splitting, Tompson et al. derived the Poisson equation for pressure:

$$\nabla^2 p = \frac{1}{\Delta t}\nabla \cdot \boldsymbol{u}^* \qquad (3)$$

where $\boldsymbol{u}^*$ is calculated from the previous time step, by ignoring the effect of pressure gradient. Correcting the velocity $\boldsymbol{u}^*$ by taking into account the effect of pressure gradient, results in:

$$\boldsymbol{u} = \boldsymbol{u}^* - \frac{1}{\rho}\nabla p \qquad (4)$$

Instead of using a conventional linear equations solver, such as the pre-conditioned conjugate gradient (PCG) solver, Tompson et al. propose to use unsupervised machine-learning to solve the Poisson equation (3). They developed a ConvNet architecture, using MSE loss function derived from the divergence-free constraint (2).

The computational domain is a uniform 2D or 3D grid, depending on the flow field to be solved. Velocity obtained from equation (4) should satisfy the constraint expressed in equation (2) at each of the grid node. Tompson et al. define an objective function and formulate the inference solution as an unsupervised machine learning task. The loss function is given by:

$$f_{obj} = \sum_i w_i \{\nabla \cdot \boldsymbol{u}\}_i^2 = \sum_i w_i \{\nabla \cdot \left(\boldsymbol{u}^* - \frac{1}{\rho}\nabla p\right)\}_i^2 \qquad (5)$$

Where w_i is the weighting factor at node *i*. The machine-learning task is to minimize the objective function to infer pressure. Equation (5) states that we are going to train a model to derive an optimum set of weighting factors, so that the error in satisfying divergence-free constraint becomes a minimum.

Antonio Alguacil (2) re-implemented Tompson et al.'s method using PyTorch. He expands the original unsupervised learning approach to a hybrid of supervised-unsupervised learning, in which the loss function is a linear combination of MSE-norm and L1 norm,

$$f_{obj} = \beta \sum_i w_i \{\nabla \cdot \boldsymbol{u}\}_i^2 + (1-\beta) \sum_i |p_p - p_{tgt}| \qquad (6)$$

Where β is a pre-defined weighting factor. p_p and p_{tgt} are the predicted pressure and the target pressure. The latter is produced by a CFD solver.

The above form as described in equation (6) has an advantage that the target value from CFD solver should already satisfy both continuity constraint and boundary condition at the object surface. By setting $\beta = 0$, the learning procedure becomes fully supervised, and by setting $\beta = 1$, it becomes fully unsupervised. Unsupervised training is less costly than supervised training but the resulting model is more likely to produce unphysical solutions.

Figure 1 shows two video clips from a 2D demo case, which is generated from a trained model, using the FluidNet architecture.

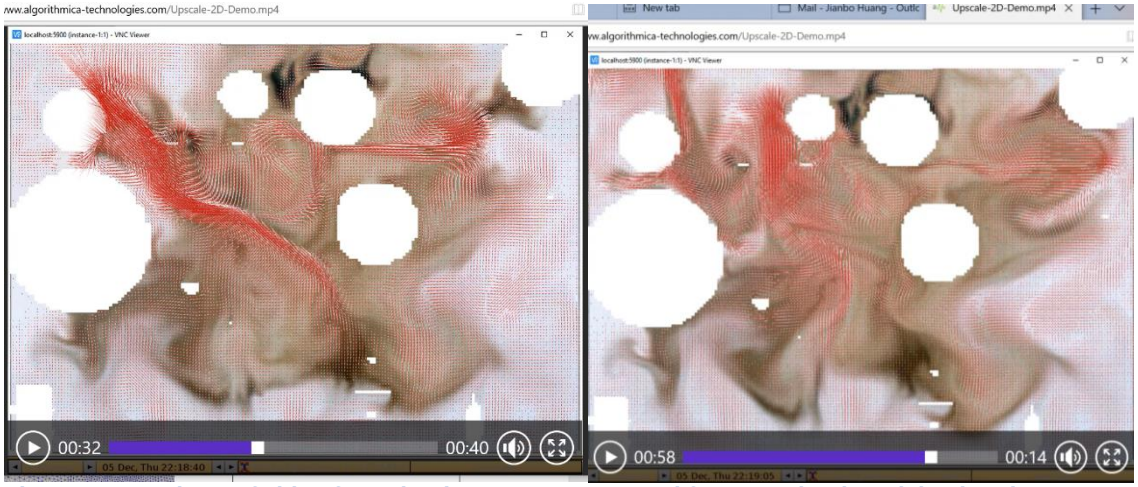


Figure 1 Snapshots of video from the demo case generated from a trained model using the FluidNet architecture

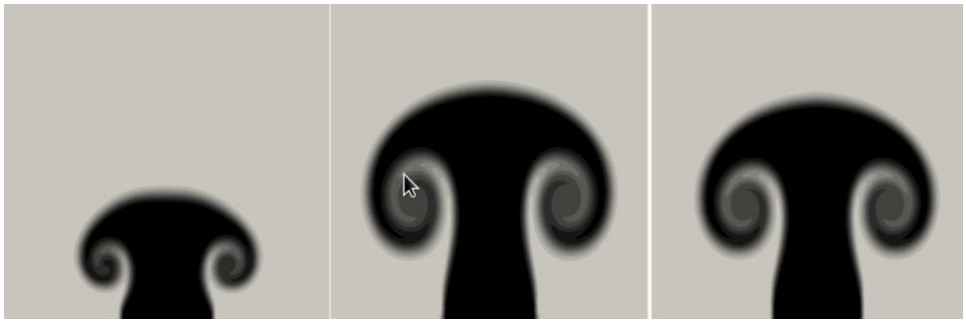


Figure 2 Results for the Rayleigh-Taylor bubble instability problem, generated from a trained model using the FluidNet architecture

Figure 2 shows the AI-model prediction for the Rayleigh-Taylor bubble instability problem. The AI model is trained using the FluidNet architecture as described in (1).

2.2. Analysis of the unsupervised machine learning approach

The method presented in section 2.1 is general enough to be adopted in the UPSCALE project. However, modifications would be needed to suit the purpose for aerothermal simulations of flow around vehicles.

The first important difference is that the machine-learning architecture we are developing must handle viscos flow with various turbulence models, therefore the Poisson equation will be of a slightly different form. In the context of steady-state, incompressible, turbulent flow, the pressure projection equation will take the following form:

$$\nabla(\alpha\nabla p) = \nabla \cdot \boldsymbol{u}^* \qquad (7)$$

Where α is a field derived from solving momentum equation with both laminar and turbulence viscosities.

Another important issue is that in a machine-learning approach, the optimization of loss is based on uniform grids, which are not fitted to the boundary surfaces. This has two important consequences:

- when deriving pressure from equation (5) the boundary condition (zero transpiration, no-slip) on the surface of solid objects in the flow field is not strictly satisfied.
- it is not possible to use the typical high aspect ratio near wall layer cells that are typically required for accurate CFD prediction of vehicle flows.

In order for this method to be used productively for vehicle aerodynamics, both these issues have to be overcome. As mentioned previously, the expected performance gain using the FluidNet method (or its derivatives) is of the order of 20%.

2.3. Fully supervised approach for acceleration of aerothermal simulations around vehicles

2.3.1. Description of the AI model

Based on experiences with the FluidNet architecture and geometry based flow prediction in general, we propose a new fully supervised approach for deriving both pressure and velocity fields, using a ConvNet architecture. The new approach uses the current velocity and pressure fields as input and then attempt to predict the entire flow field at some point in the future. The input and target training data are obtained from a CFD solver, so significant upfront data generation is required. In the current work, we use the steady incompressible flow module in the HELYX CFD package provided by ENGYS.

The task of the machine-learning procedure proposed here is to train a model to infer both pressure and velocity fields, so that the total absolute difference between model-predicted values and the CFD-simulated values are minimized. The objective function to minimize is therefore as follows:

$$f_{obj} = \sum_i |\boldsymbol{U}_p - \boldsymbol{U}_{tgt}| + \sum_i |p_p - p_{tgt}| \qquad (8)$$

Where $\boldsymbol{U}_p$ is velocity vector predicted by the trained model, and $\boldsymbol{U}_{tgt}$ is the velocity vector obtained by the CFD solver. The approach we proposed has the following advantages:

1. The trained model has the capability of replacing both pressure and velocity solvers, therefore is likely to yield bigger gain in flow prediction efficiency.
2. Since the target value already satisfies the boundary conditions, the model predicted value will also satisfy the boundary condition, as long as the prediction accuracy is good enough. In Tompson et al.'s method, boundary conditions have to be integrated into the training process.
3. The training process will be faster. The unsupervised approach and hybrid approach both involving calculating the gradients of velocity and pressure during training, thus adding extra overhead to the training process.

The U-net model, as described in (3), is used to build the training network. The architecture is shown in Figure 3.

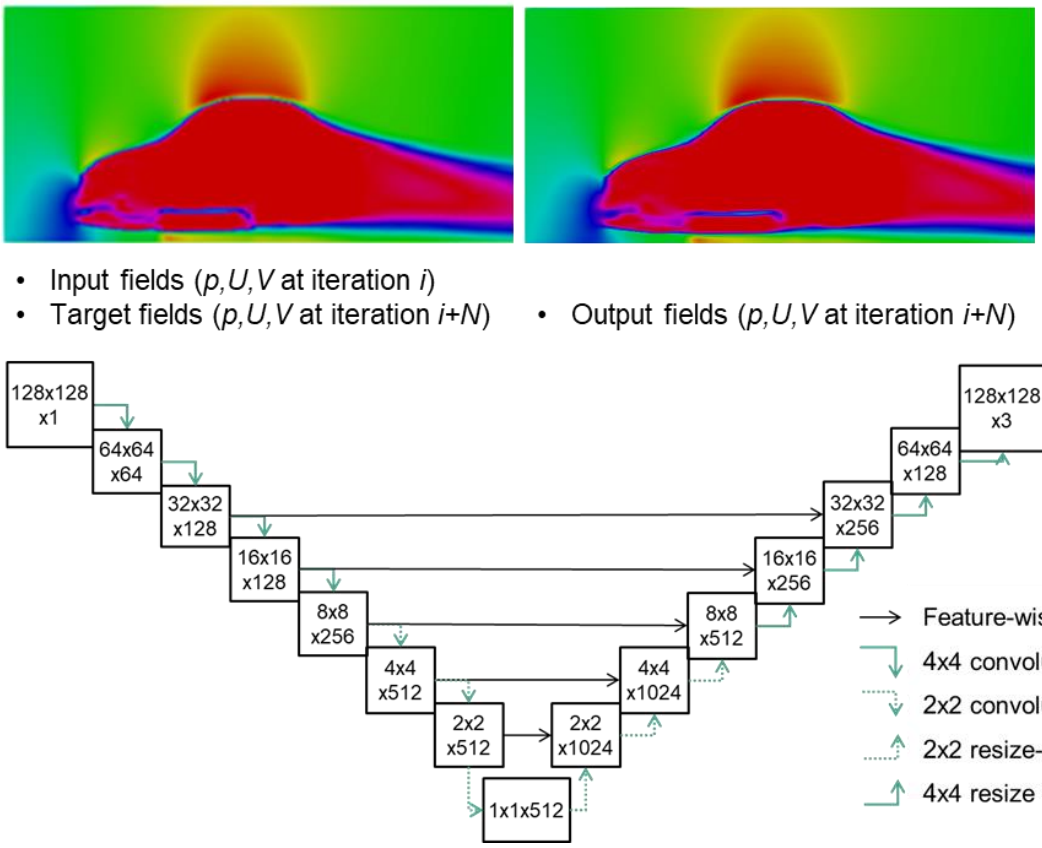


Figure 3 U-shaped ConvNet architecture for CFD regression applications

2.3.2. Model accuracy assessment of the fully supervised ConvNet architecture

To validate the ConvNet model proposed in sub-section 2.3.1, we consider a use case of aerodynamic flow around an aerofoil. The shape code of the aerofoil is e857 from the UIUC aerofoil database (3). The shape of the aerofoil and the CFD mesh around it are shown in Figure 4. Figure 5 provides the relative error for pressure prediction and Figure 6 presents the relative error for velocity prediction.

Figure 7 depicts a typical comparison between AI prediction and CFD simulation for velocity field around the e857 aerofoil from the UIUC database. The prediction and simulation advances from iteration 1545 to 1546. The overall accuracy for the 181 test cases is 99.72%, which is approaching the accuracy a CFD solver would normally achieve. Figure 8 compares the pressure field predicted by the AI model and by the CFD solver. The mean overall accuracy is not as good as velocity, but still respectable at 97.22%. Figure 9 shows the distribution of errors in the predictions of the AI model for velocity and pressure. The biggest error is in the vicinity of the aerofoil surface.

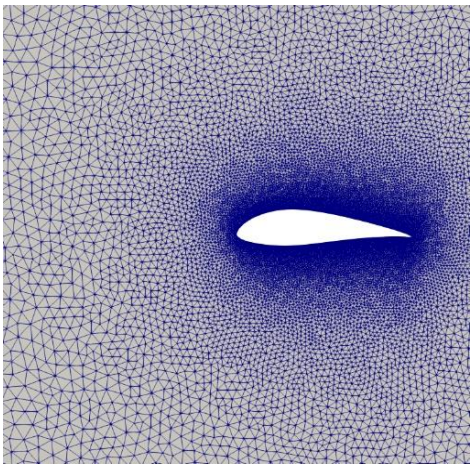


Figure 4 Geometry and mesh grid around the e857 aerofoil from UIUC database

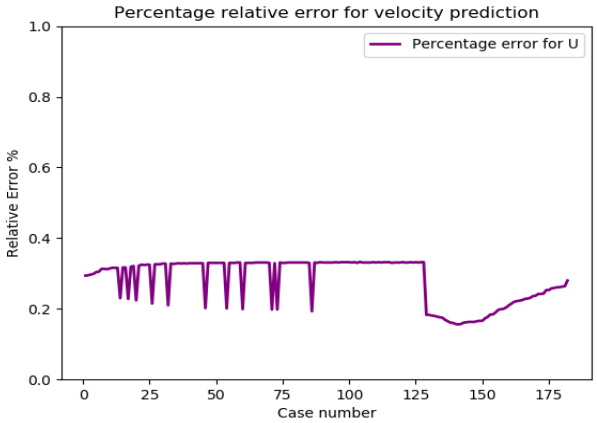


Figure 5 Relative error for velocity prediction. The overall accuracy for 181 predictions is 99.72% (relative error 0.28%)

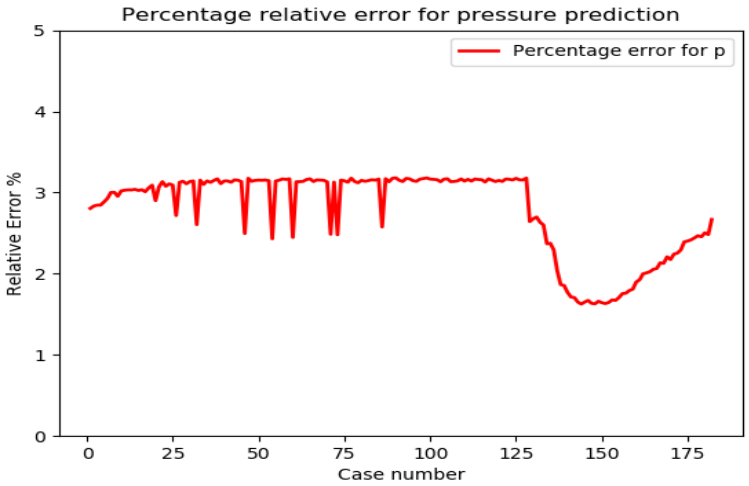


Figure 6 Relative error for pressure prediction. The overall accuracy for 181 predictions is 97.22% (relative error 2.78%)

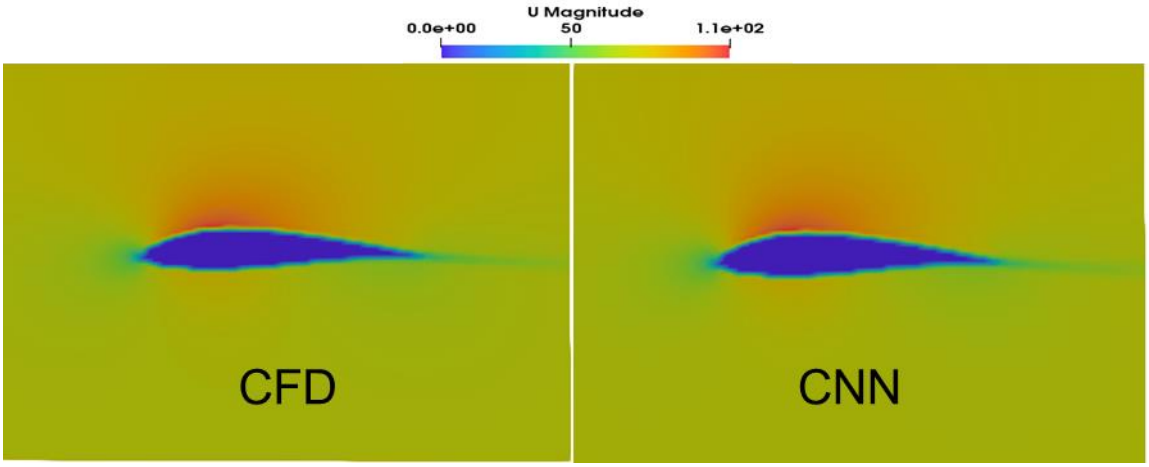


Figure 7 Comparison between AI prediction and CFD simulation for velocity field around the e857 aerofoil from the UIUC database. The prediction and simulation advances from iteration 1545 to 1546

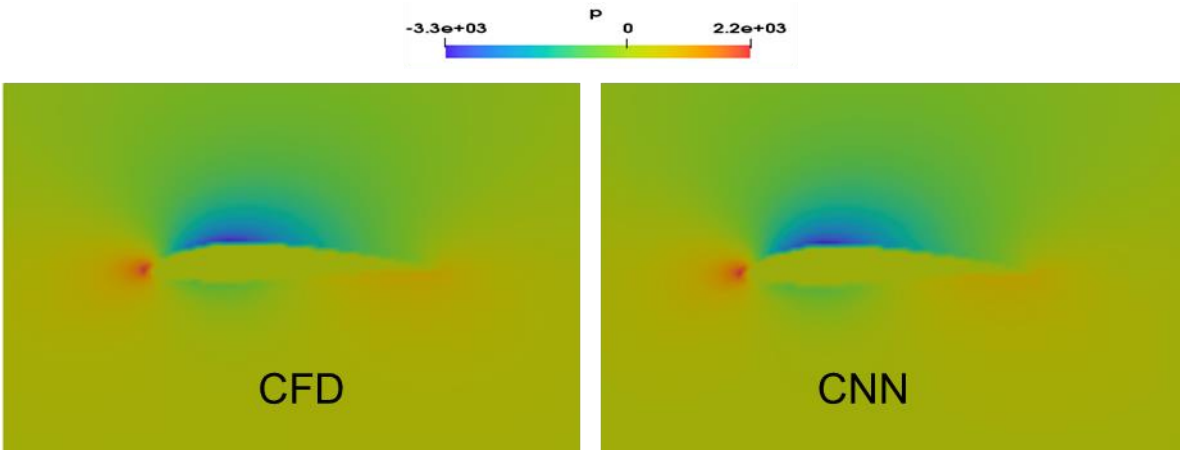


Figure 8 Comparison between AI prediction and CFD simulation for pressure field around the e857 aerofoil from the UIUC database. The prediction and simulation advances from iteration 1545 to 1546

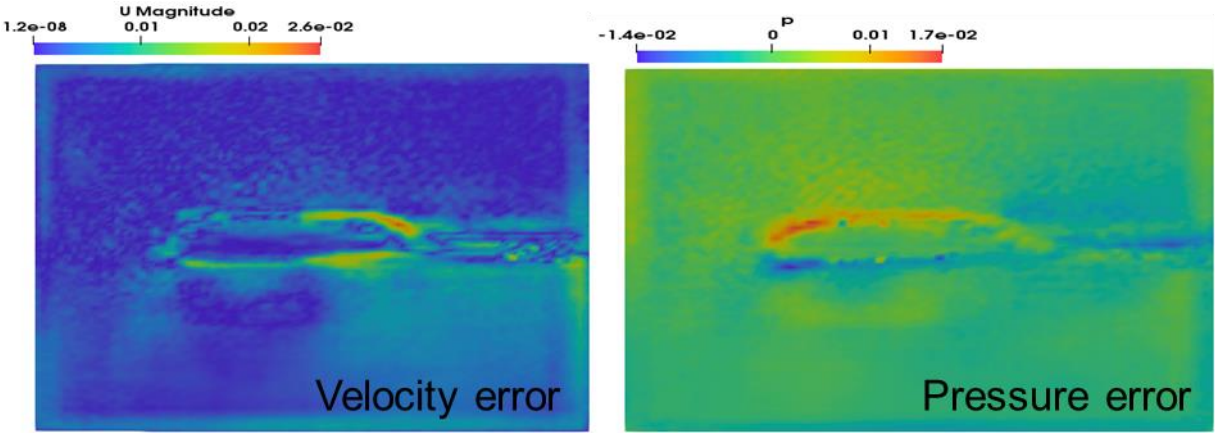


Figure 9 Difference between AI prediction and CFD simulation

2.3.3. Suitability assessment of the fully supervised ConvNet architecture

In this sub-section, we will assess the suitability of the ConvNet architecture described in sub-section 2.3.1, to be used as a tool for speeding up aerothermal flow simulations.

The initial investigation indicates that an AI model, given an input field from CFD simulation at a particular iteration, can predict the output at the next iteration relatively well (with error below 3% for pressure, and 0.5% for velocity). However, if we use the model prediction as input field to advance into further iterations, the solution quickly becomes unstable.

The reason for the above-mentioned characteristics is that the AI prediction does not strictly satisfy the continuity condition. The error accumulates with further predictions, based on the non-divergence-free fields of previous prediction results.

In order to make the system work, we need to correct the AI prediction results, filtering out the error by applying a CFD solver. If this is done on an iteration-by-iteration basis, the AI model will not in practice accelerate the solution or might do so only marginally.

With this assessment, we conclude that advance by 1 iteration at a time in the AI model, is not a viable approach. Instead, we propose to train a model, which is able to predict the solution N iterations ahead, where N is a pre-defined parameter, say N=100.

The N-iteration-advancing approach, of course requires a number of iterations from the CFD solver to filter out the ‘noise field’ in the AI predicted solution. It is hoped that the number of iterations required to filter the noise from the AI prediction will be much less than the number of iterations advanced by the AI model, so that an overall speed-up of the hybrid solver can be achieved.

3. N-iteration-advancing fully supervised approach for acceleration of aerothermal simulations around vehicles

In this section, we will discuss the new AI-based solution acceleration approach developed at ENGYS in the context of UPSCALE. The methodology of this approach is similar to the one described in sub-section 2.3.1, except that we train a model, with input value of pressure field *p*, velocity field *U* at a given time step. We want the model to predict the flow field N iterations in the future, with N is a pre-defined number. To test the potential of the method in the context of the report, we choose N=100.

The same U-shaped ConvNet architecture as described in section 2.3 is used, with the major difference that the model is trained to project the solution 100 iteration ahead rather than 1.

The argument behind this approach is that no matter how good an AI model is, it will not in general be 100% accurate. Since the CFD simulation results are used as target values to train a model, the predicted field will always have a small perturbation from the target one.

Let us assume that the AI-model predicted pressure and velocity are denoted by $p^*, \boldsymbol{U}^*$ respectively, while the true values are p, U respectively. We can express the predicted field as the true field plus a noise field:

$$p^* = p \pm \varepsilon_p, \qquad U^* = U \pm \varepsilon_u \tag{9}$$

Where $\varepsilon_p, \varepsilon_u$ are random variables with uniform distribution in such a way that their mean value equals to the mean error in the model prediction.

The $\pm$ sign is chosen randomly, assuming a Bernoulli distribution of probability 0.5 for each outcome. let k be a random number with Bernoulli distribution in (0,1) with equal probability 0.5 to take either value. If k==1, we take + sign, otherwise we take – sign.

The presence of a noise field makes the pressure field no longer satisfy the continuity equation to the required accuracy. In order to make the subsequent simulation to converge, we need to filter out this noise field by applying a CFD solver. This means that the AI-predicted values for pressure and velocity will be used as initial field value, and apply the momentum and pressure solver in the following fashion:

Let r_p, r_u be the residual for pressure and velocity respectively, which are the measurement of how well the continuity and the momentum equations are satisfied. Further, let tol_p, tol_u be the tolerance for the pressure and velocity solution error, then we can establish the following momentum-continuity coupling pseudo-code procedure:

Let *cnt*=0

While ($r_p > tol_p$ or $r_u > tol_u$
{
    Solve momentum equation
    Solve continuity equation

Get new residual r_p, r_u
cnt++
}

By using this approach, it is anticipated that the iteration numbers count *cnt*, needed to make the corrected field satisfy both continuity and momentum equations to the required accuracy, will be much less than the 100 iterations the AI-based model projects the solution, so we will be able to reduce the number of full cost iterations in the a given *N* iteration period by by (*N-cnt*) iterations.

3.1. Model accuracy of the N-iteration advancing AI architecture

In this section, we will analyse the model accuracy of the N-iteration-advancing AI architecture. Here model accuracy is defined as the overall difference between the predicted field and the target field. With this definition, the relative error for pressure prediction and for velocity prediction can be expressed as:

$$err_p = \sum_{i=1}^{Nc}(p_{pred}\text{-}p_{tgt})/\sum_{i=1}^{Nc}|p_{tgt}| \qquad (10)$$

Where Nc is the total number of cells in the computational domain, p_{pred} is the pressure predicted by machine-learning model, p_{tgt} is the target pressure value coming from the CFD simulation (the correct value).

$$err_U = \sum_{i=1}^{Nc}(U_{pred}\text{-}U_{tgt})/\sum_{i=1}^{Nc}|U_{tgt}| \qquad (11)$$

Where U_{pred} is the velocity vector predicted by the machine-learning model, and U_{tgt} is the velocity vector from CFD simulation.

3.2. A use case for the N-iteration advancing AI architecture

As a use case, we consider the aerofoil cases as investigated by N. Thuerey et al (3).

The training dataset in our use case contain aerofoil models from the UIUC database. Altogether, there are 1498 different shapes in the database. The training and test shapes are randomly chosen.

The inlet flow is set into freestream, with velocity component *U* as a random variable uniformly distributed between (0,100] m/s. The *V* and *W* components are set into zero.

The computational domain is bounded by a box from point (-5,-5,0) to point (5,5,1), with length unit being meter.The sampling region is bounded by a box from point (-0.5,-1,0) to point (1.5,1,1). The sampling resolution is 128x128x1.

Each training sample consists of a tensor of (6,128,128), where the first dimension 6 is the number of channels at each sampling point, which consists of 3 input channel (one pressure, one velocity U, one velocity V, at a given iteration number *i* in the list

[100,200,300,400,500,600,700,800,900].

The output channel contains target values for pressure *p*, velocity component *U*, velocity component *V* at iteration number *i*+100. The objective of the training system is to minimize the L1 loss between the normalized pressure and velocity prediction and the corresponding normalized target values.

In the normalization process, the velocity is normalized by the inlet velocity U_0 and pressure normalized by U_0^2. The objective function is defined as:

$$f_{obj} = \sum |\acute{p}_{pred} - \acute{p}_{tgt}| + \sum |\acute{U}_{pred} - \acute{U}_{tgt}| \qquad (12)$$

The training dataset contains 3990 samples and the test dataset contains 363 samples.

The controlling parameters for the model training process are listed in Table 1.

Table 1 Parameters for the N-iteration-advancing neural network

Item name	Item value
Number of iterations	60000
Batch size	12
Learning rate	0.0006
Learning rate decay	true
Sampling grid density	128x128x1
Dropout rate	0
Weight decay rate	0
Optimization scheme	Adam

In this use case, the best accuracy for the trained models we obtained is 91.47% for pressure, and 99.4% for velocity.

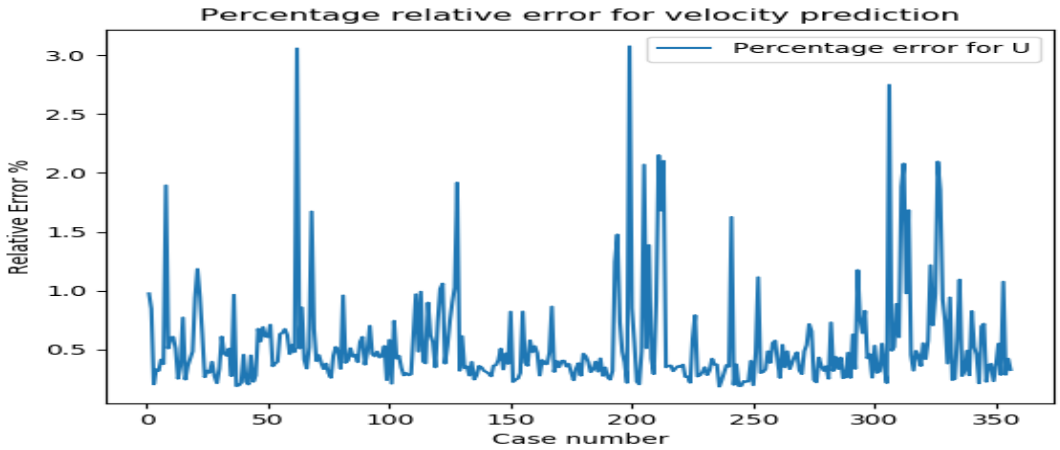


Figure 10 Relative error in velocity prediction by the AI model for each individual test case. The test cases are not in the training loop to ensure reliability

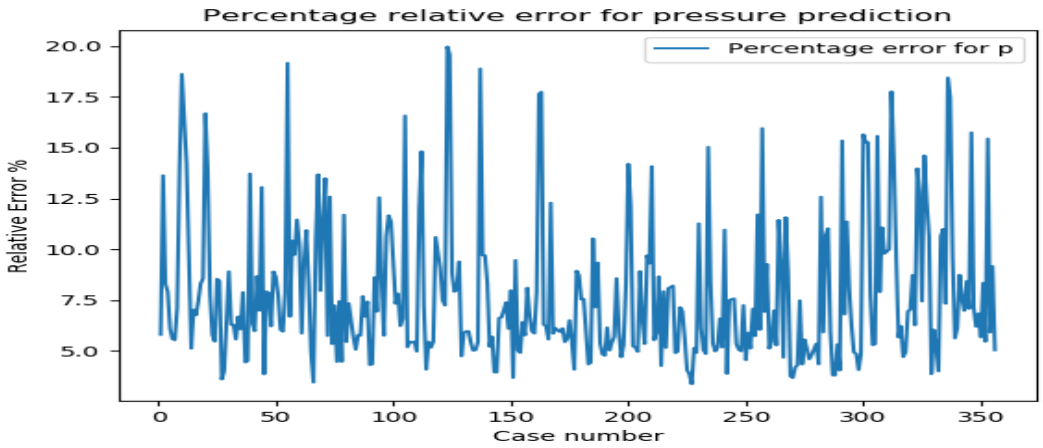


Figure 11 Relative error in pressure prediction by the AI model for each individual test case. The test cases are not in the training loop to ensure reliability

Figure 10 and Figure 11 show the relative error with regard to each individual test case for velocity and pressure predicted by the AI model. As can be seen, the error distribution is quite random, due to the random choice of test cases. The accuracy for velocity is above 97% for all the 364 test cases, with average accuracy around 99.4%. The minimum accuracy for pressure prediction relative to the CFD is above 80%, with the average accuracy being 91.47%.

3.3. Analysis of potential speed up of convergence for the N-iteration- advancing AI architecture

In order to analyze the potential speed up for the simulation convergence, we choose some of the test cases, and combine the AI prediction with CFD simulation.
The prediction time is very short comparing to the advance of a CFD iteration, especially for large cases. To make the analysis easier, we neglect the time that AI prediction takes. The CPU time taken for advancing each iteration in CFD simulation is roughly the same, therefore the speed up ratio can be roughly judged by the iteration numbers a CFD simulation takes.

In our use case, an AI prediction advances the solution by 100 CFD iterations. However, the AI prediction is not accurate enough as input for the AI model to advance further. A number of CFD iterations will be needed to filter out the noise field an AI prediction introduced.

Assume that *m* iterations is needed to bring the residual to the same level when the simulation is advanced by CFD procedure. Suppose m<100, then the percentage gain during this one AI prediction step is:

$$R_{speed} = 100\,(N - m)/N \qquad (13)$$

Where R_{speed} is the percentage speed up of 1 AI prediction step. To simplify our analysis process, we further assume that the error introduced by AI prediction is uniformly distributed in the sampling region.

Since the residual tolerance for pressure is typically more difficult to reduce than that for velocity, we use this as the primary judge of convergence quality to assess the impact of the AI prediction steps. The restarted results from an AI prediction must reach the same level of a non-restart, iteration-by-iteration CFD solver for the filtering process to be deemed complete.

To illustrate the typical impact of the method, Table 2 show a random set of test samples from the cases investigated in section 3.2:

Table 2 Potential speed up of N-step-advancing AI prediction model

Case Name	Relative error in p with AI prediction	Relative error in U with AI prediction	Restart iterations	P-residual should be	p-residual reached	Potential Speed up
1087-11-73_100	0.137	0.0167	33	0.00856	0.00844	67%
1087-11-73_400	0.0795	0.00067	58	0.00289	0.00288	42%
1000-57-44_100	0.136	0.00845	46	0.011	0.0109	54%
1000-57-44_300	0.0829	0.002	67	0.00302	0.00306	33%
1000-57-44_600	0.079	0.0033	96	2.48e-4	4.89e-4	0%
0-45-87_700	0.058	0.00976	93	0.00224	0.0023	7%
101-49-95_300	0.0559	0.004	65	0.00272	0.00263	35%
1013-34-11_200	0.186	0.006	58	0.00644	0.0064	42%
1014-94-85_100	0.142	0.005	58	0.0128	0.0128	42%
1018-17-1_200	0.0703	0.0077	45	0.0076	0.00754	55%
1029-14-63_300	0.057	0.006	51	0.0041	0.00402	49%
1029-14-63_700	0.054	0.0026	96	7.0e-4	9.0e-4	0%

In Table 2, the naming of the cases follows a few simple rules described below.
Using case **1018-17-1_200** as an example, the first digit **1018** means it is the 1018th sample in the dataset, **17-1** means the free stream flow velocity is 17.1 m/s (with decimal point replaced by – to avoid conflict with the file extension). The last 200 after the underscore means the AI prediction starts from iteration number 200 and advances the solution to iteration 300.

From Table 2, we can see that significant speed up can be achieved during the early and middle stages of the simulation, when the simulation residual is relatively high.

When the simulation residual is already low (i.e., when the case is near-converging), using multi-step-advancing AI prediction is no longer advantageous. Under such conditions AI projection may not speed up the simulation and may even slow down further convergence. Based on our observations the solution projection technique becomes counterproductive once the continuity residual drops below 0.002. The proposed method is therefore best suited to early stage solution acceleration, while the latter stages of the simulation will have to be treated in the normal manner. Based on the limited statistics collected so far, we expect the method to achieve roughly 50% speed-up in the first 500 iterations and none thereafter. Depending on the complexity and details of the case this could result in overall speed-up of the simulation of between 10 and 25%.

4. Smart mapping technology for speeding up aerothermal simulations

4.1. Background

One of the most time- and resource-intensive processes in UPSCALE is to generate datasets to train the various AI models. A distinguishing feature of these data sets is that the training samples share similar flow pattern, differing only by small amounts in some specific regions near the vehicle's surfaces.

To make use of the above-mentioned property, it would be advantages to use the converged solution of a previous run as an effective initial field for a new simulation, which is of similar but not identical geometrical configuration. This approach could potentially speed up the new simulation considerably, relative to the typical potential flow initialization method that is commonly used today.

Although techniques such as point-to-point mapping are available for mapping solution fields from one case to another, they do not take into account variations in the geometry and will end up mapping solid regions to fluid regions and vice versa. Therefore, unless the two cases have exactly the same geometry, there will be parts of the mapped solution that are very far from the converged solution. This will result in slower convergence, and can even result in the mapped field taking longer to converge than starting from a potential field initialization.

To overcome the problem of a flow field point being mapped into a solid object, or a point inside solid object being mapped into flow field, we propose a 'smart mapping' approach.

The idea behind this approach is that, instead of finding the point-by-point correspondence of the target field and the source field based on the geometric location, we map the two fields based on the point's relative location with regard to its distance from the solid object inside the flow. This will avoid a point inside solid object being mapped into a flow field, or vice versa.

An issue with the wall distance approach is that multiple points in the flow field may have the same wall distance. In order to make the mapping unique, there must be a one-to-one mapping between the source and the target fields.

To achieve a one-to-one mapping, we combine the wall distance approach with the physical-location approach. This means that cell centers of the two fields are first grouped according to their wall-distance values. When we do the mapping, we only map points of the two fields which are in the same wall-distance group. Within the group, the individual points are mapped according to their Cartesian coordinates.

To ensure that domains with different sizes can be mapped, we normalize both the wall distance and the Cartesian coordinates by the bounding box of the flow domain:

$$\acute{d}_w(x,y,z) = d_w\,(x,y,z)/d_{max} \tag{14}$$

$$\overline{x} = (x - x_{min})/(x_{max} - x_{min}) \tag{15}$$

$$\bar{\bar{y}} = (y - y_{min})/(y_{max} - y_{min}) \quad (16)$$

$$\bar{z} = (z - z_{min})/(z_{max} - z_{min}) \quad (17)$$

Where $d_w(x, y, z)$, is the wall distance at location (x, y, z), d_{max} is the maximum wall distance of the flow field. $x_{min}, x_{max,} y_{min}, y_{max,} z_{min}, z_{max,}$ are co-ordinates which define the bounding box of the flow domain. The result is that the source and target point always have a similar wall distance which results in much more realistic flow distributions where the target and source geometries do not match.

4.2. Outline of the 'smart mapping' procedure

The basic procedure of applying the 'smart mapping' approach in speeding up the simulation involves the following steps:

- Start from a converged solution of a different geometry and mesh configuration
- Map the flow fields (p, U, μ_t etc) at cell-centres using kdTree-based fast searching algorithm, which possesses the property of time complexity strictly log(N), where N is total number of samples
- Use wall-distance-based categorization method to pre-group the flow fields, in such a way that cell-centres with similar wall distances are in the same group
- When mapping a source field to a target field, first search the candidates from the same wall-distance group, then search the 'right' candidate based on physical distance between source and target points, use k-nearest-neighbour algorithm
- This approach prevents a cell centre point inside the flow from being mapped into a point inside a solid object

4.3. Outline of AI-based object-recognition method in the 'smart mapping' procedure

One of the first steps in the 'smart mapping' approach described in section 4.1 and section 4.2 is to identify the closest flow solution in the database to the one in question. During an object optimization process, engineers may have run many simulations with different object shapes. Some of the shapes may be quite close to each other, some may be quite different.

To make best use of the smart mapping feature, one wishes to pick up the existing flow field with the closest geometry to the one currently under consideration.

The k-nearest neighbour-based machine-learning method is best suited for this purpose. Steps in the kNN-based object recognition model are:

- Divide the domain around the object into uniform grids, with the same dimensions for all training samples
- Generate object masks by assigning a value of 1 if a grid point is inside a solid object, and 0 if it is not

- Search the database to find the best match use a log(N)-complexity fast kNN algorithm. Our validation test has found that if there is an identical geometry in the database, the kNN search algorithm will find it as the best match. If there is no identical match in the database, the search algorithm will find the closest one.

4.4. Accuracy of the 'smart mapping' approach

We have validated the 'smart mapping' approach against various use cases of different geometry, and found that:

1. When the source field and the target field are of identical configuration (not necessary the same geometry scale), we will obtain identical mapping without artificial error being introduced in the smart mapping process.
2. When the source and target fields are of different configuration, we can guarantee that the points inside flow field will not be mapped into a solid object, and a point inside a solid will not be mapped into flow point.
3. The kNN search algorithm is very efficient for both median-sized and large cases.

The following sub-sections present the results for mapping flows around a few different geometrical objects.

4.4.1. Map of flow fields around an aerofoil

This use case concerns the mapping of a flow field around an aerofoil to another aerofoil.

The source field is flow around aerofoil of shape ID geo397 in the UIUC database. The target field is of shape ID goe393. The CFD mesh grids for the two aerofoils are shown in Figure 12 and Figure 13 respectively. As one can see, the two mesh grids are both completely unstructured. There is no one-to-once correspondence with the two grids.

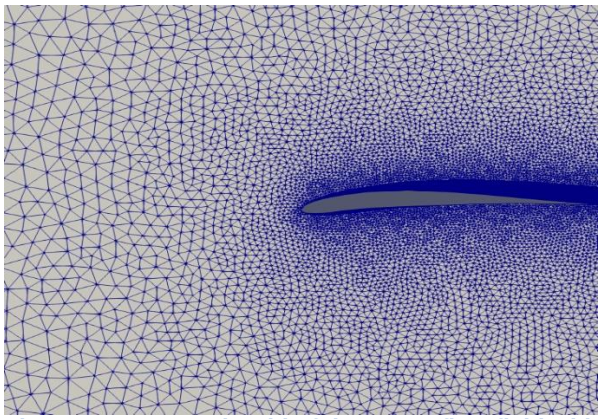


Figure 12 CFD mesh grid of the target flow field with aerofoil shape ID geo393

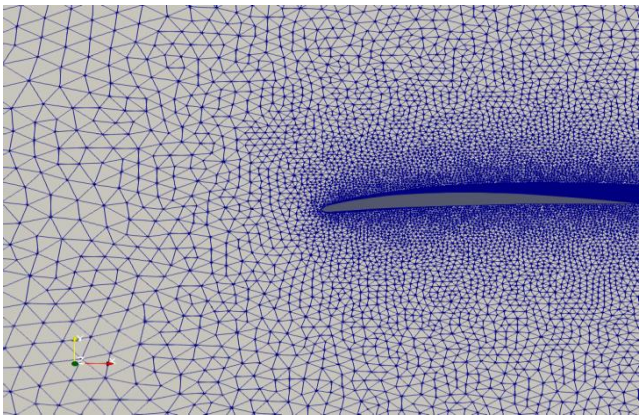


Figure 13 CFD mesh grid of the source flow field with aerofoil shape ID geo397

Figure 14 and Figure 15 compares the velocity around aerofoil in the source case, and the mapped velocity field around a similar but different aerofoil. The shape id for the source aerofoil is geo397 while for the target aerofoil is geo393.

Figure 16 and Figure 17 compare the pressure field in the source case and the mapped pressure field in the target case. As one can see, the relative locations of the two fields are mapped well, and the flow distribution in the target field is much more realistic than the potential field initialization.

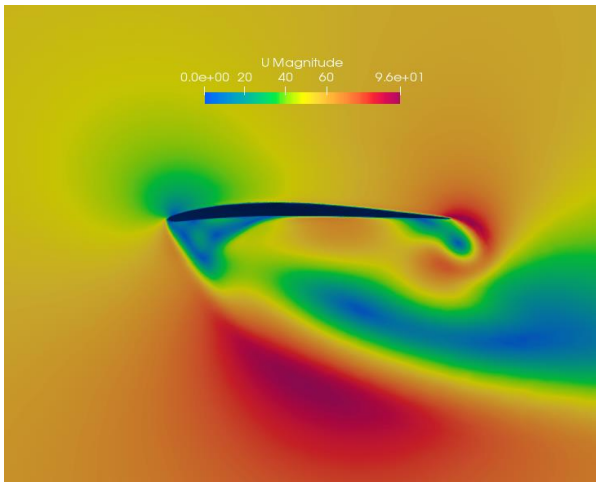


Figure 14 Velocity field around the aerofoil in the source case

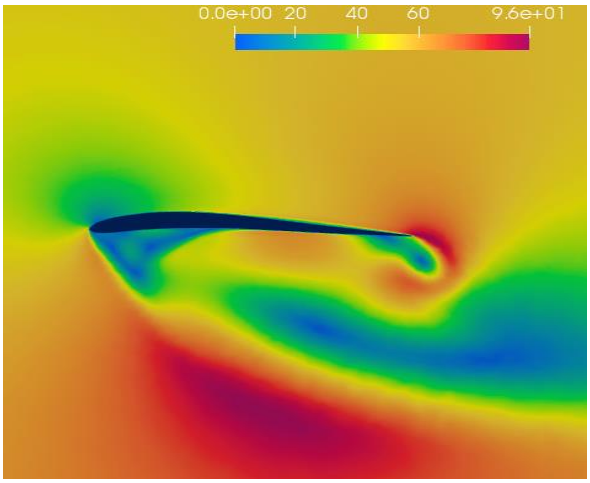


Figure 15 Mapped velocity field around the aerofoil in the target field

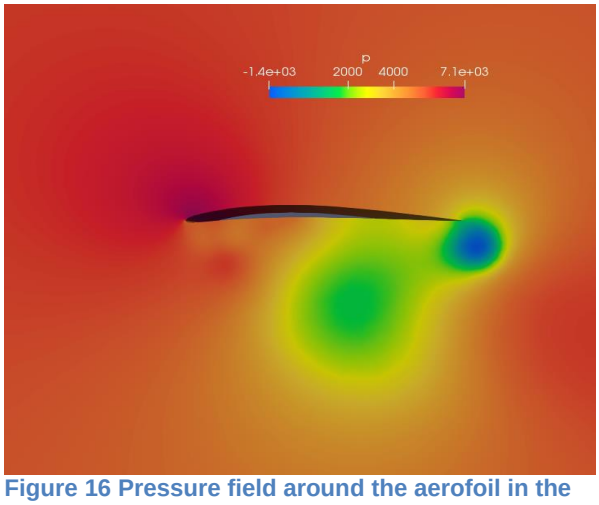


Figure 16 Pressure field around the aerofoil in the source case

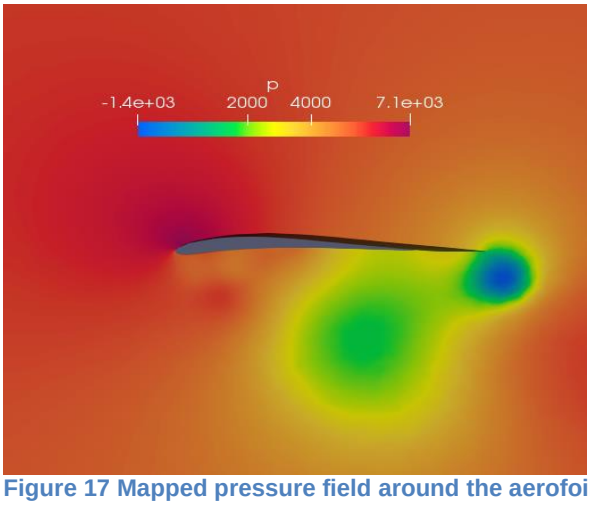


Figure 17 Mapped pressure field around the aerofoil in the target

4.5. Map of flow fields around two-dimensional square boxes

In this sub-section, we present the results in mapping the flow field around a two-dimensional square object, which is rotated round its centre, to a target flow field around a square with different angle of rotation. With the conventional mapping method, a significant part of the source field will be mapped into the solid object in the target field, and some regions inside the solid object will be mapped into a fluid region.

Figure 18 and Figure 19 show the source and the mapped pressure field around the 2d boxes, rotated around their centers by 16 degrees and 24 degrees anti-clockwise respectively. As we can see, even with these quite different geometry configurations, the mapped fields are relatively plausible (and certainly better than potential flow).

Figure 20 and Figure 21 present the source and the mapped velocity field around the 2d boxes rotated around their centers by 16 degrees and 24 degrees anti-clockwise respectively. The smart mapping approach gives initial fields, which are much more realistic than the one with potential flow initialization, or with the conventional mapping which is purely based on cell-center coordinates.

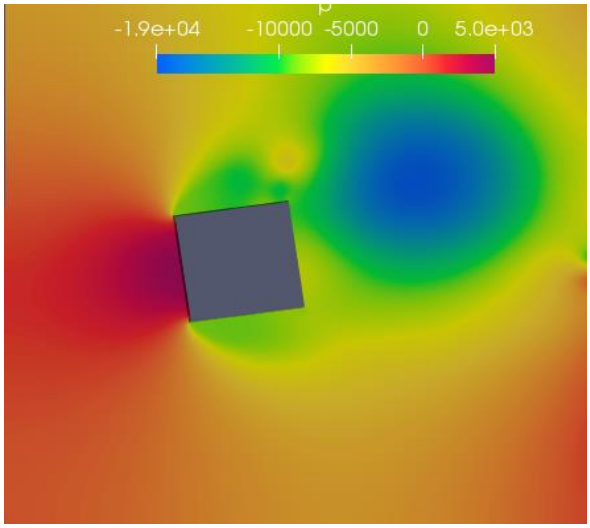


Figure 18 Source pressure field around a two-dimensional square, rotated about its centre by 16 degrees anti-clockwise

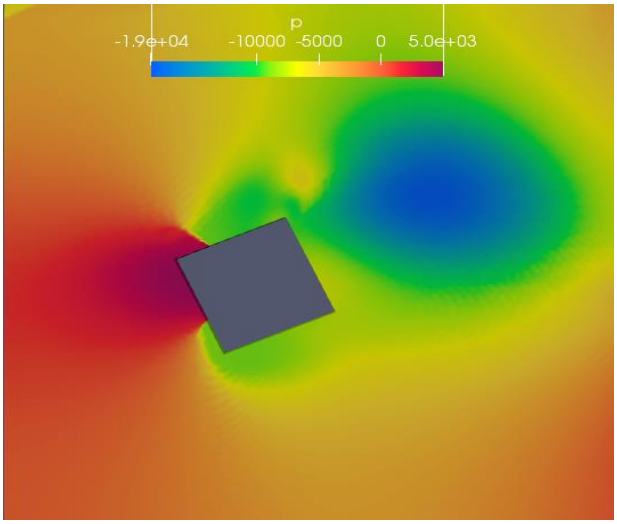


Figure 19 Mapped pressure field around a two-dimensional square, rotated about its centre by 24 degrees anti-clockwise

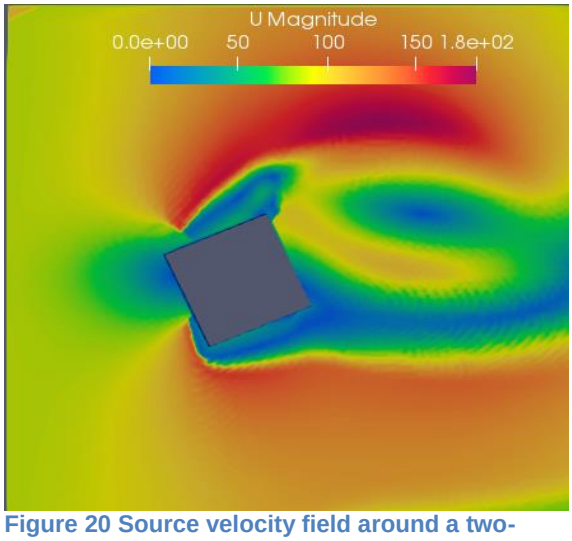


Figure 20 Source velocity field around a two-dimensional square, rotated about its centre by 16 degrees anti-clockwise

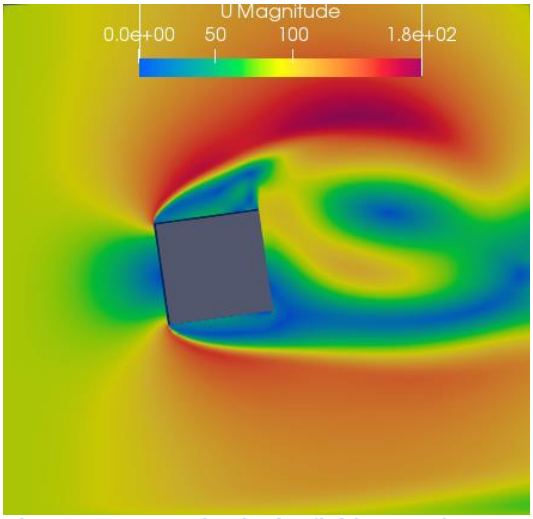


Figure 21 Mapped velocity field around a two-dimensional square, rotated about its centre by 24 degrees anti-clockwise.

4.6. Map of flow fields around two-dimensional cars

To demonstrate the potential of the baseline smart-mapping algorithm (excluding the machine-learning based source categorisation and selection), we compare various metrics from the execution of two cases initialised with smart-mapping, traditional mapping and a uniform initial flow field. The case in question is a 2D car (DrivAer (4) variant) which is the principle development platform for the UPSCALE AI systems and has been extensively parameterised to allow the rapid generation of variant cases (see UPSCALE D2.1 for details). For smart mapping, we randomly choose 3 variants of this model, specify one as the source/baseline and two as mapping targets. All cases are run to 2000 iterations and can be considered representative for comparison purposes. The computational grid of a typical car geometry is shown in Figure 22.

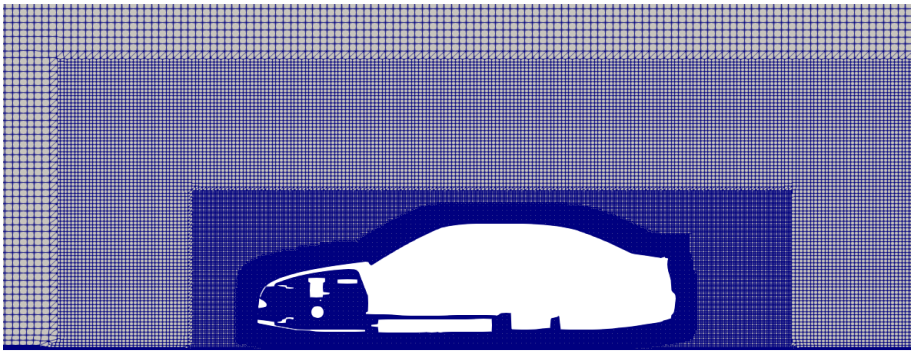
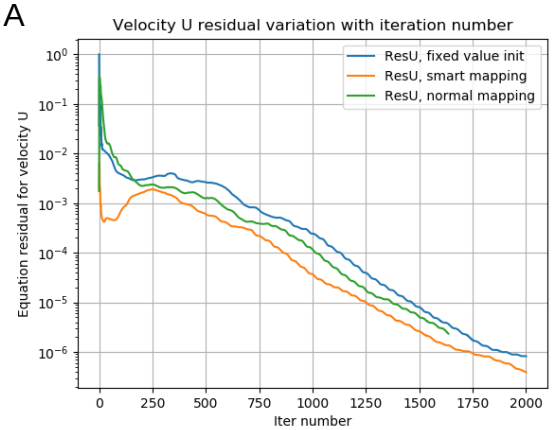


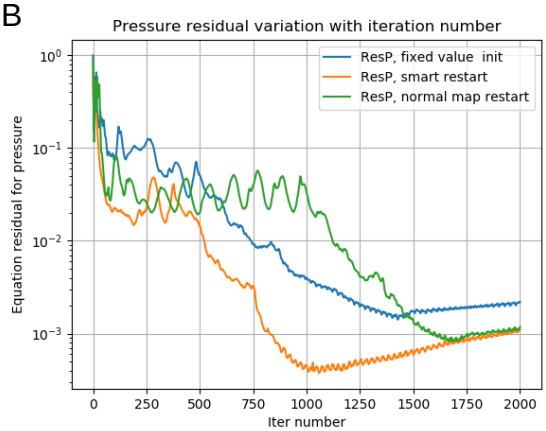
Figure 22 Geometry and mesh grid of the 2d car cases

Figure 23 shows residuals for velocity and pressure for both target cases (A – left column & B – right column). The focus here is purely on the statistics and not on individual features of the two random cases. In all plots the yellow line, representing the variant initialised with smart-mapping, has a lower residual that the case initialised with direct mapping (green) or with no initialisation (blue), indicating that the equation systems converge faster than with traditional mapping. Surprisingly, for case B it appears that the traditional mapping performs worse than the uniform flow initialisation. We will not explore the mechanism for this unexpected performance in this instance, but it serves to highlight that flow simulation performance can be very sensitive to the initialisation. It should also be noted that we do not expect one initialisation method to outperform all others for all cases, but we do expect the smart-mapping approach to outperform other methods on average.

A



B



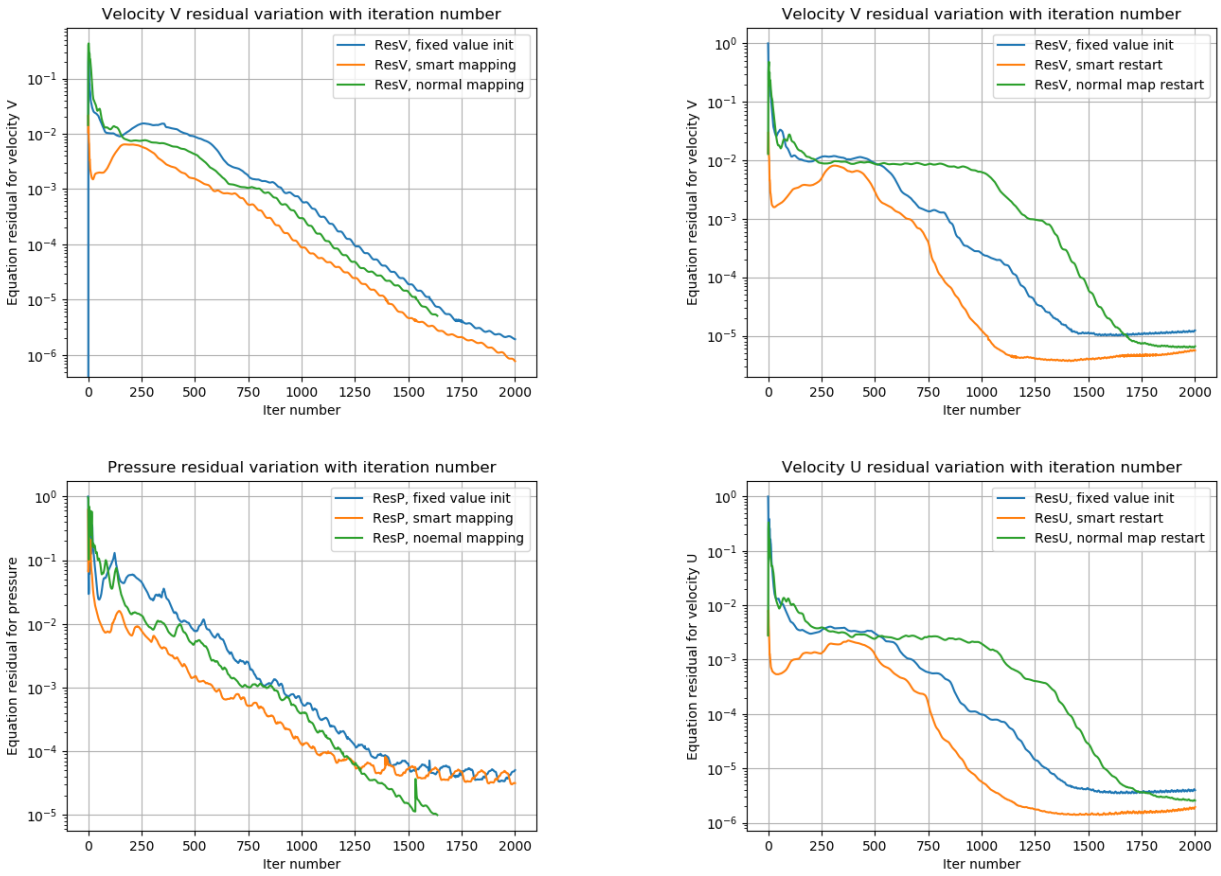


Figure 23 Residuals for two cases (A - left column & B - right column) initialised using different methods. Each row shows different residuals: row 1 = U, row 2 - V, row 3 - p

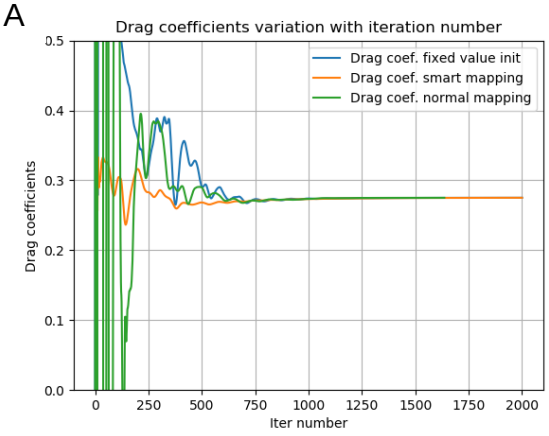
Figure 24 compares the development of the integral lift and drag forces over the vehicle as a function of solution iteration. Again, the two cases A & B are used to contrast the variable influence of different initialisation methods. Again, the smart-mapping initialisation performs consistently better than other methods and the inconsistent behaviour of direct mapping is mirrored. It is difficult to measure an exact level of acceleration provided by the smart mapping as it depends on how convergence is judged. The smart map force integrals are however generally less oscillatory and tend to reach the equilibrium state around 10-30 percent faster than the alternatives. More importantly, the smart-mapping approach is consistent unlike direct mapping which performs well in case A, but not in case B. We can thus (at least tentatively) conclude that smart-mapping will see even greater margins of performance gain relative to any single alternative method.

An additional point to note is that the new mapping approach derived from developments undertaken as part of Task 1.3 and WP2 and while it conforms to the general brief of ML-based acceleration it was not initially an explicit objective. Results are however promising enough that additional refinement and process integration is recommended.

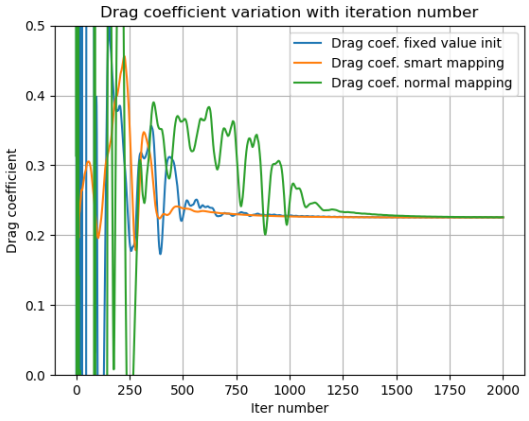
There is one specific aspect that bears further consideration: mapping methods in the context of finite volume in general only map cell centred values. Face based fields like the volume flux are not mapped or are mapped without maintaining their critical divergence-free characteristic. Starting any calculation from a flux field that is not divergence free will tend to have

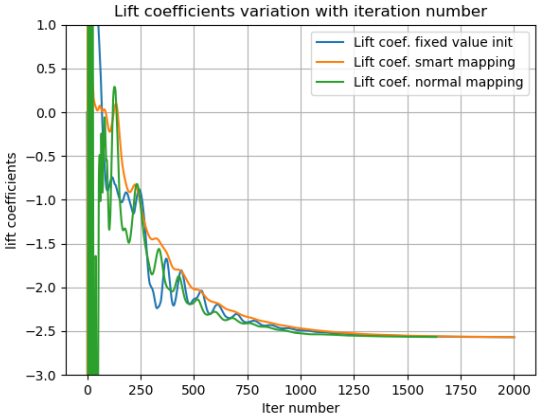
destabilising influence on the initial iterations. However, by using smart-mapping it is theoretically possible to rapidly recover this face flux field by additionally mapping momentum matrix coefficients and then performing a pressure solution pre-processing step. Testing out this proposed approach will be the main focus of future work on the topic of smart-mapping.

A



B





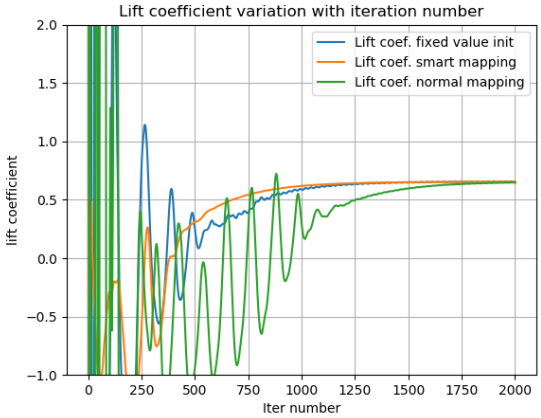


Figure 24 Lift and drag development for case A (left) and case B (right)

5. Momentum-continuity coupled solver approach

Another potential way to speed up the aerothermal simulation of external flow around vehicles is by using a momentum-continuity coupled solver approach. With this method, the momentum and continuity equations are fully coupled, significantly increasing the implicitness of the system.

It is commonly recognized fact that coupled implicit solvers can lead to faster convergence than explicit or segregated solvers, but it comes at the cost of significantly increased complexity in the underlying code. In this section, we will look into the feasibility of speeding up UPSCALE AI training by applying the coupled solver technology.

5.1. Theoretical background of the momentum-continuity coupled solver approach

In a tensor notation, the continuity and momentum equations for a steady-state flow simulation can be written in the following way:

$$\begin{aligned} R^p &= \frac{\partial v_j}{\partial x_j} = 0 \\ R_i^v &= v_j \frac{\partial v_i}{\partial x_j} + \frac{\partial p}{\partial x_i} - \frac{\partial}{\partial x_j}\left[\nu_{eff}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right)\right] = 0 \end{aligned} \quad (18)$$

Where v_j is the velocity component in j (j=x,y,z) direction. *p* is pressure.

In the segregated solver, A is a matrix with scalar coefficients (for every equation – v_{xyz}, p)
All equations are solved in a sequential mode, so that coupling between them is explicit. To make the system converge, an under-relaxation factor which has a default value of 0.7 for velocity, and 0.3 for pressure, has to be applied.

Under-relaxation improves the stiffness of the discretised equation system, but add more explicitness to the system, hence slows down convergence.

In a momentum-continuity-coupled block-solver approach, every element in A is a matrix 4x4 (v_i+p), therefore velocity and pressure are implicitly solved in a single system. The flux linearization constitutes the only explicit part of the system.

The discretized Incompressible steady-state Navier-Stokes equations can be expressed in the following matrix form:

$$\begin{bmatrix} A_x & & & \frac{\partial}{\partial x} \\ & A_y & & \frac{\partial}{\partial y} \\ & & A_z & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} & 0 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ p \end{bmatrix} = \begin{bmatrix} b_x \\ b_y \\ b_z \\ b^p \end{bmatrix} \quad (19)$$

Velocity at a face can be expressed by adopting the Rhie-Chow approach:

$$v_i^f = [v_i]^f - \left[\frac{1}{A}\right]\left(\frac{\partial p}{\partial x_i}\bigg|^f - \left[\frac{\partial p}{\partial x_i}\right]^f\right) \quad (20)$$

Where *f* denotes the face of a computational cell. The discretised continuity equation for a computational cell can be expressed as:

$$\sum_f v_i^f S_i^f = 0 \rightarrow \sum_f \left([v_i]^f S_i^f - \left[\frac{1}{A}\right]^f \left(\frac{\partial p}{\partial x_i}\bigg|^f - \left[\frac{\partial p}{\partial x_i}\right]^f\right) S_i^f\right) \quad (21)$$

Where summation is done for all faces enclosing a computational cell. The final form of the coupled momentum-continuity system can be expressed as:

$$\begin{bmatrix} A_x & & & a_{xp} \\ & A_y & & a_{yp} \\ & & A_z & a_{zp} \\ a_{px} & a_{py} & a_{pz} & A_p \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ p \end{bmatrix}\Bigg|_c + \sum_i^N \begin{bmatrix} A_x & & & a_{xp} \\ & A_y & & a_{yp} \\ & & A_z & a_{zp} \\ a_{px} & a_{py} & a_{pz} & A_p \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ p \end{bmatrix}\Bigg|_i = \begin{bmatrix} b_x \\ b_y \\ b_z \\ b_p \end{bmatrix}\Bigg|_c \quad (22)$$

5.2. Comparison of results between the segregated and the coupled solver

In this section, we present a comparison between the conventional segregated solver and the new coupled solver. We take the geometry of a two-dimensional car as an example. To make a like-for-like comparison, the same mesh and boundary conditions are used for both approaches.

Figure 5.1 compares the velocity field produced by the two different solvers. There is no visual difference between the results. Figure 5.2 compares the convergence history for the momentum and continuity equations. The coupled solver converges to a residual of 1.0e-6 for both momentum and continuity equations within around 100 seconds clock time, while the segregated solver can only reach a residual in the order 1.0e-3 for momentum and about 0.003 for continuity. After clock time 200 seconds, the residual does not change with the segregated solver, i.e. convergence stalls.

Figure 5.3 and Figure 5.4 present comparison of the lift and drag variation with clock time, obtained using the conventional segregated solver and the new coupled solver respectively. Judging from the solution graphs, the coupled solver reaches the converged results in about

50 seconds clock time, while the segregated solver requires about 150 seconds to reach the converged results. The is a small difference in prediction of lift coefficients by the two solvers, while there is almost no difference for the converged solution of drag coefficient by the two solvers.

Judging from the equation residuals, the coupled solver needs about 50 seconds to reach satisfactory residual tolerance, while the segregated one needs 200 seconds to reach a stable but much higher residual. In this case, the coupled solver shows much superior performance than the segregated counterpart in both efficiency and the capability to reach a lower residual tolerance. The coupled solver is 4 times faster than the segregated one for this particular case.

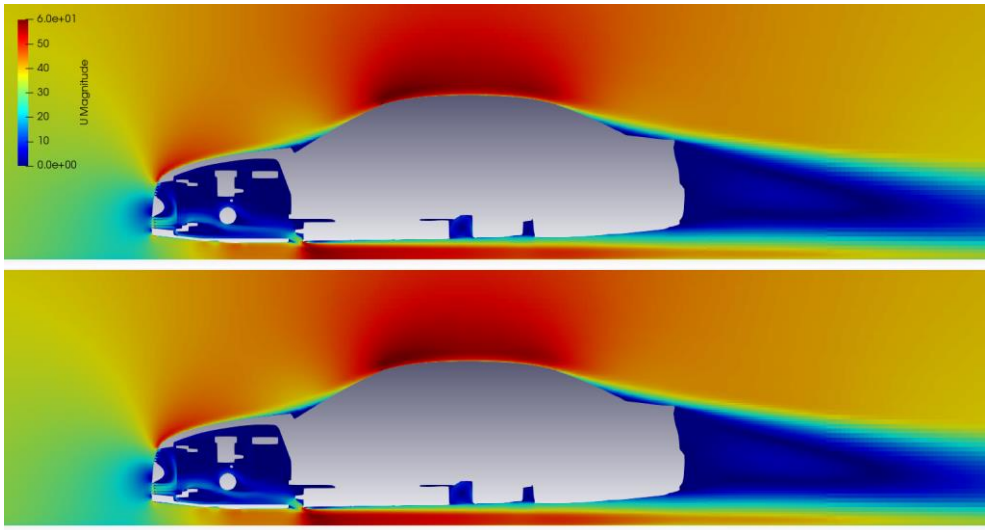


Figure 25 Comparison of velocity field around the two-dimensional car model. Top: solution from segregated solver, bottom: solution from the coupled solver

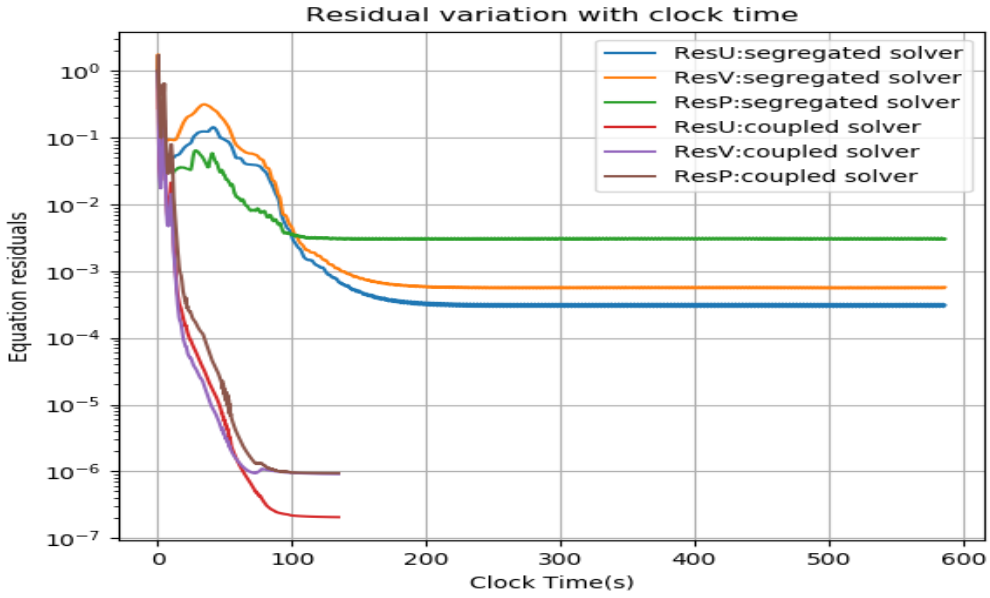


Figure 26 Comparison of equation residuals between the segregated solver and the coupled solver

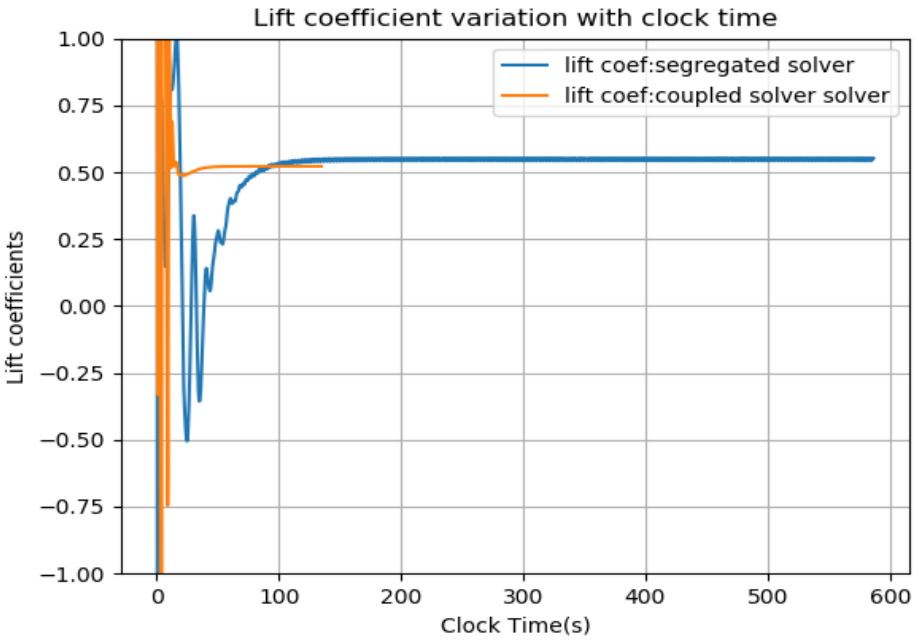


Figure 27 Comparison of lift coefficients obtained using the segregated and the coupled solver for the two-dimensional car model

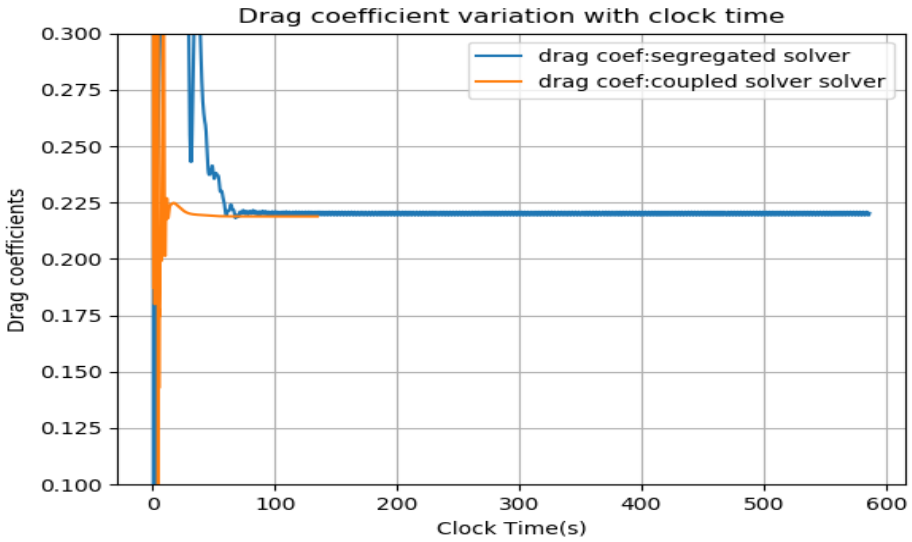


Figure 28 Comparison of drag coefficients obtained using the segregated and the coupled solver for the two-dimensional car model

6. Conclusions

This report presents a detailed analysis of some novel techniques in speeding up the process of aerothermal simulations. In particular, the prospects of using AI technology for speeding up incompressible steady-state CFD simulation is investigated. The following conclusions can be drawn from our investigations:

- It is found that the AI-based Poisson solver for pressure projection method, as proposed by Tompson et al, can be used as a means for speeding up incompressible aerothermal flow simulations around vehicles. The efficiency increase is around 20%, as shown in the original paper (1).
- The unsupervised AI method proposed in (1) was originally designed for inviscid flows, but can be extended to viscus and turbulent flows needed by the UPSCALE project
- We have proposed a supervised-training-based version of the AI methodology for speeding up incompressible steady-state CFD simulations, in which both momentum and continuity solver of the CFD method can be replaced by the proposed AI version. This version of AI-based flow prediction can speed up by 40% to 50% during the initial iterative stage when the residual is relatively high. This method becomes less effective as the system converges but can be expected to deliver performance benefits of the order of 20%.
- Providing reasonable initial fields can significantly speed up CFD calculations. The conventional coordinate-based mapping method cannot reliably be used for speeding up the simulation process and for some cases it can lead to slower convergence than the completely unphysical (but smooth) potential-flow-initialization method.
- An AI-based smart-mapping technique was proposed and demonstrated, by which a flow simulation can be started from a converged solution of a flow field around a similar but not identical object. The closest flow configuration to the current one in consideration can be picked up using k-nearest-neighbour-based machine-learning method, from a database of previously generated solutions. The mapping itself is then mediated by an additional distance field so that context is preserved between dissimilar geometries.
- The smart-mapping method has a potential to speed up simulation process by a significant margin (> 20%). For large DoE's, such as those used in training neural networks, it will progressively and automatically improve the time to solution of each new case as the database of source cases becomes more densely populated. In the context of using this technique for preparing training data for the UPSCALE project, this feature will be particularly useful. The fast categorisation-based training means that a previously completed sample case can be used immediately to initialise another training sample, often making smart-mapping more effective than a fully trained model in terms of performance gain.
- The momentum-continuity-coupling algorithm enhancement, developed by ENGYS, has proven to be a very effective way of speeding up the aerothermal simulation process. Speed-up ratios between 2 and 10 times have been demonstrated. This method can be combined with the smart-mapping method to further increase efficiency and has a plethora of potential benefits in related fields.

- By combining different techniques, we anticipate that the performance gain will exceed the project goal of 20% by a significant margin.

6.1. Recommendations for future work

This report has identified and demonstrated several methods, based on both on machine learning and complimentary technologies, that have the potential to speed up CFD for AI training calculations significantly. The next step is to make a more detailed assessment in terms of the integration cost of the different methods relative to their potential impact on the principle objective. Future efforts will be guided by this assessment. At present it looks like all the investigated methods have significant promise, our technical recommendations are thus as follows:

1. Two “FluidNet” variants for accelerating pressure-velocity solvers were explored. The supervised solution acceleration approach appears to have the most promise and is significantly easier to integrate with existing solver machinery. As such, the supervised solution acceleration approach will be the focus of continued investigations in this sector.
2. Although not explicitly covered in this work, additional application of the FluidNet concept in the context of running transient simulations with larger time intervals without unduly sacrificing accuracy was identified. This approach would synergise with both the original FluidNet work and Task 1.1.3 “Physics Informed Machine Learning”. We thus strongly recommend that it be pursued as gains in performance could far exceed those expected from the baseline FluidNet method.
3. The principle of smart mapping has been proven and its potential demonstrated. The next step is to properly quantify its benefits in the context of a practical design of experiments scenario to measure the aggregate impact of the methodology in a representative environment over a large range of cases. Also explore potential refinements to the smart mapping process to further improve performance gains.
4. The coupled pressure-velocity work builds on ENGYS foreground and has shown major performance benefits at minimal expense. The next step is to integrate the block coupled solver into the AI training framework. We also recommend exploring additional acceleration potential in the block solver framework made available by the implicit nature of the pressure-velocity system.

6.2. Risk assessment

Much of the work explored in the context of Task 1.1.1 is experimental in nature and therefore inherently carries high risk. At the same time, the overall impact of Task 1.1.1 on the UPSCALE project is small and can thus be mitigated by several conventional means (mainly resource reallocation). In addition, we have chosen to investigate a broad set of acceleration solutions focused on machine learning, but also including baseline solver algorithmic enhancements. By relying on a bigger set of potential acceleration targets we reduce the risk of any one element failing, but conversely, fewer resources are available for each specific approach. By choosing a limited number of methods of varying complexity and maturity we hope to strike the best balance between risk and positive outcomes.

Based on current progress, no substantial risks have been identified that would seriously impact the successful completion of task 1.1.1.

7. Acknowledgements

The author(s) would like to thank the partners in the project for their input, valuable comments on previous drafts and for performing the review.

Project partners:

PARTICIPANT Nº	PARTICIPANT ORGANISATION NAME	COUNTRY
1 (Coordinator)	IDIADA AUTOMOTIVE TECHNOLOGY SA (IDIADA),	Spain
2	VOLVO PERSONVAGNAR AB (Volvo Cars)	Sweden
3	VOLKSWAGEN AG (VW)	Germany
4	CENTRO RICERCHE FIAT SCPA (CRF)	Italy
5	ESI GROUP (ESI GROUP)	France
6	ENGYS LTD (ENGYS LTD)	United Kingdom
7	Kompetenzzentrum - Das Virtuelle Fahrzeug, Forschungsgesellschaft mbH (VIF)	Austria
8	VRIJE UNIVERSITEIT BRUSSEL (VUB)	Belgium
9	ECOLE NATIONALE SUPERIEURE D'ARTS ET METIERS (ENSAM PARISTECH)	France
10	ALGORITHMICA TECHNOLOGIES GMBH (ALGORITHMICA)	Germany
11	F INICIATIVAS I MAS D MAS I SL (F-INICIATIVAS)	Spain

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no. 824306

8. References

1. *Accelerating Eulerian Fluid Simulation with Convolutional Networks.* **J. Tompson, K. Schlachter, P. Sprechmann, K. Perlin.** ICLR : https://github.com/google/FluidNet, 2017.
2. **Alguacil, Antonio.** fluidnet_cxx: Accelerating Fluid Simulation with Convolutional Neural Networks. A PyTorch/ATen Implementation. *Github.* [Online] https://github.com/jolibrain/fluidnet_cxx.
3. *Deep Learning Methods for Reynolds-Averaged Navier-Stokes Simulations of Airfoil Flows.* **N. Thuerey, K. Weißenow, L. Prantl, Xiangyu Hu.** s.l. : Technical University of Munich.
4. *Introduction of a New Realistic Generic Car Model for Aerodynamic Investigations.* **A. Heft, T. Indinger, N. Adams.** Detroit, Michigan, USA : s.n., April 23-26, 2012. SAE 2012 World Congress. pp. 2012-01-0168.
5. *Adjoint methods for car aerodynamics.* **Othmer, C.** 1, s.l. : Journal of Mathematics in Industry, Vol. 4. 2190-5983.
6. *Searching for important factors in simulation models with many factors: Sequential bifurcation.* **Kleijnen, J. P. C., & Bettonvil, B. W. M.** 1, s.l. : European Journal of Operational Research, 1997, Vol. 96. 180-194.
7. *Reduced Order Modeling for Vehicle Aerodynamics via Proper Orthogonal Decomposition .* **M. Mrosek, C. Othmer, R. Radespiel.** s.l. : SAE International Journal of Passenger Cars: Mechanical Systems, 2019-09-21.
8. **Guo, X. and Li, W. and Iorio, F.** Convolutional Neural Networks for Steady Flow Approximation. *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* New York, NY, USA : s.n., 2016, pp. 481-490.
9. *On Active Subspaces in Car Aerodynamics.* **C. Othmer, T. Lukaczyk,P. G. Constantine, J. J. Alonso.** Washington D.C. : AIAA Aviation forum, 2019. 17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.
10. *Application of POD techniques in CFD optimization of vehicle aerodymamics.* **L. Miretti, E. M. Ribaldone,.** 2016.
11. *Towards Real-time Vehicle Aerodynamic Design via Multi-fidelity Data-driven Reduced Order Modeling.* **A. Bertram, C. Othmer, R. Zimmermann.** Florida, USA : AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2018.
12. *Free Form Deformation, mesh morphing and reduced order methods: enablers.* **F. Salmoiraghia, A. Scardiglibc, H. Telibb and G. Rozza.** s.l. : International Journal of Computational Fluid Dynamics, 2018-03-15. Vol. 32, pp. 233-247.
13. *Deep Learning Methods for Reynolds-Averaged Navier–Stokes Simulations of Airfoil Flows.* **Nils Thuerey, Konstantin Weißenow, Lukas Prantl, and Xiangyu Hu.** 0, s.l. : AIAA, 2019, AIAA Journal, Vol. 0.
14. *A Physics Informed Machine Learning Approach for Reconstructing Reynolds Stress Modeling Discrepancies Based on DNS Data.* **Jian-Xun Wang, Jin-Long Wu, Heng Xiao.** 2017, Physical Review Fluids 2, Vol. 034603.
15. *Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty.* **Templeton, J. Ling and J.** 085103, 2015, Physics of Fluids, Vol. 27.