International Forum on Aeroelasticity and Structural Dynamics
IFASD 2019
9-13 June 2019, Savannah, Georgia, USA

SELF ADAPTIVE POD BASED ROM 3D AEROELASTIC SIMULATIONS

Rubén Moreno-Ramos[1], Fernando Varas[2], José M. Vega[2]

[1]Altran Innovación, Aeronautics Space and Defence Division
C/ Campezo, 1, E-28022, Madrid, Spain
ruben.morenoramos@altran.com

[2]E.T.S.I. Aeronáutica y del Espacio, Universidad Politécnica de Madrid
Pza. Cardenal Cisneros, E-28040, Madrid, Spain

Keywords: Reduced Order Models, Computational Aeroelasticity

Abstract: A self adaptive POD based ROM capable of speeding up computational aeroelastic simulations, the so called "POD on the Fly", was presented during IFASD 2017 in Como, Italy. The previously presented method couples together a 3 d.o.f. airfoil structural model with a CFD solver (Euler or RANS equations) and during the simulation a POD based ROM is created, where the governing equations are proyected only on a subset of selected points from the computational domain, acting the remaining as slaves. Once the ROM is deemed to be accurate enough, it takes over the simulation until the ROM accuracy is degraded (mainly due to truncation instabilities), moment at which the simulation is continued by the CFD, updating the ROM and repeating the process. In the previously presented simulations, significant speed up was achieved for subsonic simulations, to the point that the solution would not have to revert back to the CFD solver after a given point. This paper extends the ROM capabilities to 3D simulations, solving issues like the non-trivial selection of master cells and the computational issues of ROMs created over multiple computer nodes, as it is typical of industrial 3D CFD simulations.

1 INTRODUCTION

The prediction of steady forces over lifting surfaces and aerodynamic bodies by means of computational fluid dynamics (CFD) is becoming a standard in the aerospace industry. It allows for accurately predicting the vehicle lift, drag, loads and handling qualities, among other quantities. Moreover, this is done with an affordable computational cost and the support of a decreasing number of testing campaigns, needed to tune some solver parameters.

The use of CFD to predict unsteady aerodynamic forces is also intended as an alternative to current methods used in production tasks, such as the doublet lattice method (DLM), which is based on potential theory and does not capture nonlinear/viscous effects appearing in detached flows and shock waves. For dynamic aeroelastic simulations, however, unsteady CFD requires a large computational effort and its accuracy is not guaranteed by the large majority of current solvers given the complexity of the flow, as highlighted in recent meetings on the subject [1]. The research community is making a tremendous effort on improving the unsteady CFD solvers available to the aeroelastic engineers. In some cases, good correlation with wind tunnel tests has been reported [2], but the computational cost still prevents them from being implemented

in industry due to the large number of simulations that are needed to account for different combinations of the parameters defining the vehicle configuration and flight conditions.

The previous work from the authors, briefly reintroduced here, has shown the so called "POD on the Fly" method to have very good potential to significantly reduce the cost of unsteady CFD used in fluid structure interaction simulations. It does so by using the full numerical solver to create a ROM based on POD modes, that is able to take over the simulation until loses its validity, moment at which reverts back to the full numerical solver to continue the simulation and update itself, as in Figure 1.

The method has shown significant accelerations for the subsonic fluid structure interaction case of a 3 d.o.f. airfoil, to the point that after a given point the ROM is capable of taking over the full simulation, reverting seldom to the full solver. In the transonic case the method has to deal with the motion of the strong shocks present on the airfoil, which imposes a shorter validity of the ROM, but the ROM is still capable of significant performance increases.

This paper extends the method to more industry related three-dimensional simulations, where typically the computational effort is spread over a set of computational nodes. In particular, while for the 3 d.o.f. airfoil example the master cells selection was performed ad-hoc focusing on including enough points in the airfoil wake and around the airfoil, it is not clear how such a selection would be performed for a wing, since the risk of including unnecessary points could be much higher, and so would be the computational cost penalty.

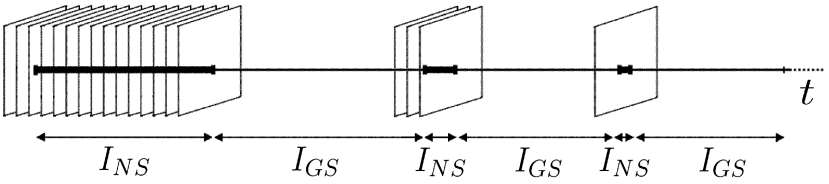

Figure 1: Sketch showing the ROM and full numerical solver time intervals

2 "POD ON THE FLY"

2.1 Truncated POD modes basis

In this method, the integration of the fluid governing equations starts using the full numerical solver. Then, after a prescribed amount of time steps, for a time T_o^{CFD}, the flow state variables $\mathcal{Q}$ for each time step can be concatenated into a snapshot matrix, such as

$$\boldsymbol{S} = \begin{bmatrix} \mathcal{Q}_1^1 & \mathcal{Q}_1^2 & \cdots & \cdots & \mathcal{Q}_1^T \\ \mathcal{Q}_2^1 & \mathcal{Q}_2^2 & \cdots & \cdots & \mathcal{Q}_2^T \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathcal{Q}_M^1 & \mathcal{Q}_M^2 & \cdots & \cdots & \mathcal{Q}_M^T \end{bmatrix}. \tag{1}$$

The POD modes are calculated from this snapshot matrix. Traditionally, POD modes have been calculated by obtaining the covariance matrix of the snapshot matrix as $[R_{ij}] = \langle S_i, S_j \rangle$ for a given inner product $\langle \cdot, \cdot \rangle$. It is customary expressing the eigenvalue decomposition of $\boldsymbol{R}$ as

$$\boldsymbol{R}\mathbf{A} = \mathbf{A}\mathbf{D}^2, \tag{2}$$

where the columns of $\mathbf{A}$ are the orthonormal eigenvectors of $\boldsymbol{R}$ and $\mathbf{D}^2$ is the diagonal matrix of the eigenvalues of $\boldsymbol{R}$, whose square roots are the singular values. Then, the POD modes are the columns of the following matrix

$$\boldsymbol{\Phi} = \boldsymbol{S}\mathbf{A}\mathbf{D}^{-1}, \tag{3}$$

so that they are orthonormal with the considered inner product.

However, due to the fact that the covariance matrix requires the multiplication by the inner product of the snapshots, the POD modes associated with singular values such that $(D_{jj})/(D_{11}) < 10^{-8}$ are not valid [3] with the standard double precision binary floating point format.

For the particular, and typical, case in which the inner product can be represented as

$$\langle S_i, S_j \rangle = {S_i}^T \boldsymbol{\mathcal{G}} S_j, \tag{4}$$

where $\boldsymbol{\mathcal{G}}$ is a diagonal matrix, the accuracy of the POD modes is increased by avoiding the creation of the covariance matrix. In a first step, SVD decomposition is applied to

$$\tilde{\boldsymbol{S}} = \sqrt{\boldsymbol{\mathcal{G}}}\boldsymbol{S}, \tag{5}$$

so that

$$\tilde{\boldsymbol{S}} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{6}$$

and therefore

$$\boldsymbol{S} = \sqrt{\boldsymbol{\mathcal{G}}^{-1}}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T. \tag{7}$$

Substituting these into the definition of the covariance matrix, and taking into account that D and $\mathcal{G}$ are diagonal matrices, it follows that

$$\boldsymbol{R} = \boldsymbol{S}^T\boldsymbol{\mathcal{G}}\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{U}^T\sqrt{\boldsymbol{\mathcal{G}}^{-1}}\boldsymbol{\mathcal{G}}\sqrt{\boldsymbol{\mathcal{G}}^{-1}}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = \boldsymbol{V}\boldsymbol{\Sigma}^2\boldsymbol{V}^T. \tag{8}$$

Finally, comparing (2) and (8) it is trivial to show that

$$\boldsymbol{\Phi} = \boldsymbol{S}\boldsymbol{V}\boldsymbol{\Sigma}^{-1} = \sqrt{\boldsymbol{\mathcal{G}}^{-1}}\boldsymbol{U}. \tag{9}$$

An important property of the POD modes is that, if the snapshot matrix is approximated as S^N making use of the first N POD modes, the relative root mean square error (RRMSE) can be estimated as

$$RRMSE_N^T = \sqrt{\frac{\sum_{i=N+1}^{T}(D_{ii})^2}{\sum_{i=1}^{T}(D_{ii})^2}}. \tag{10}$$

In this way, only the modes such that $RRMSE_N^T < \varepsilon_{POD}$ are retained.

2.2 POD modes based on a subset mesh

In order to reduce the computational overhead associated with the construction of the POD modes, the inner product is defined only in a subset of the cells that form the computational domain, such that the evolution of the fluid variables for the non-selected cells is made slave to the selected ones. Rearranging the vectors such that the M selected subset cells are at the top, the $\mathcal{G}$ matrix is created such that

$$\boldsymbol{\mathcal{G}} = diag\left(\mathcal{G}_1, \ldots, \mathcal{G}_M, 0, \ldots, 0\right). \tag{11}$$

This is equivalent to applying standard SVD to the reduced snapshot matrix $\widehat{\boldsymbol{S}}$, which is defined by considering in the original snapshot matrix $\boldsymbol{S}$ only those rows and columns corresponding to the selected cells. It follows that

$$\widehat{\boldsymbol{S}} = \sqrt{\widehat{\boldsymbol{\mathcal{G}}}^{-1}}\widehat{\boldsymbol{U}}\,\widehat{\boldsymbol{\Sigma}}\,\overline{\widehat{\boldsymbol{V}}}^{\top}, \quad \text{with } \overline{\widehat{\boldsymbol{V}}}^{\top}\widehat{\boldsymbol{V}} = \overline{\widehat{\boldsymbol{U}}}^{\top}\widehat{\boldsymbol{U}} = M \times M\text{-unit matrix}, \tag{12}$$

where for convenience we are recalling that the columns of the matrices $\widehat{\boldsymbol{U}}$ and $\widehat{\boldsymbol{V}}$ are orthonormal.

In order to recover the POD modes at all mesh points, we first note that, using (12), the reduced mode matrix $\widehat{\boldsymbol{U}}$ can also be written as

$$\widehat{\boldsymbol{U}} = \sqrt{\widehat{\boldsymbol{\mathcal{G}}}}\widehat{\boldsymbol{S}}\,\widehat{\boldsymbol{V}}\,\widehat{\boldsymbol{\Sigma}}^{-1}. \tag{13}$$

This formula produces slightly non-orthonormal modes due to round-off errors when very small singular values are retained, which makes the matrix $\widehat{\boldsymbol{\Sigma}}$ ill-conditioned. Thus, the reduced modes are re-orthonormalized using the QR decomposition, as $\widehat{\boldsymbol{U}} = \widehat{\boldsymbol{Q}}\widehat{\boldsymbol{R}}$, which corrects (13) as

$$\widehat{\boldsymbol{U}} = \sqrt{\widehat{\boldsymbol{\mathcal{G}}}}\widehat{\boldsymbol{S}}\,\widehat{\boldsymbol{V}}\,\widehat{\boldsymbol{\Sigma}}^{-1}\widehat{\boldsymbol{R}}^{-1}, \tag{14}$$

Now, taking into account the meaning of the various matrices appearing in (14), this equation yields the components of the POD modes corresponding to the collocation points (namely, the columns of $\widehat{\boldsymbol{U}}$) as linear combinations of their counterparts for the selected snapshots (namely, the columns of $\widehat{\boldsymbol{S}}$). In an equivalent way, the the full mode matrix can be calculated as

$$\boldsymbol{\Phi} = \boldsymbol{S}\,\widehat{\boldsymbol{V}}\,\widehat{\boldsymbol{\Sigma}}^{-1}\widehat{\boldsymbol{R}}^{-1}. \tag{15}$$

The truncation of the POD basis follows the procedure presented above. Additionally, the residual equations for the rows corresponding to the non selected cells do not play any role in the minimization, so the cost can be further reduced by avoiding its calculation.

2.3 Time evolution and POD basis update

At this point, the integration is switched to the ROM solver, and the system evolved in time, until the ROM loses its validity due to changes in system dynamics or truncation instabilities. It is therefore paramount to introduce in the time evolution the ability to detect this phenomenon. This is accomplished by monitoring the residual reduction the ROM is able to achieve. Being

$$\mathcal{Q} = \mathcal{Q}_o + \boldsymbol{\Phi}\xi, \tag{16}$$

the ROM integration will finish if

$$E_{res} = \frac{\left\| \sum_{j=1}^{N} \Phi_{ij} \left(\frac{\mathcal{V}_i^{n+1}\xi_j^{n+1} - \mathcal{V}_i^n \xi_j^n}{\Delta t} \right) + \mathcal{R}_i(\mathcal{Q}_o + \Phi\xi^{n+1}) \right\|}{\left\| \mathcal{R}_i(\mathcal{Q}_o + \Phi\xi^n) \right\|} > \frac{1}{K}, \tag{17}$$

where K is a tunable parameter and $\| \cdot \|$ is associated with a previously defined inner product.

After switching back to the numerical solver, the system is evolved for a time ΔT^{CFD}. It must be noted that the selection of ΔT^{CFD} needs to consider the balance between acquiring the new system dynamics and the efficiency of the overall integration scheme.

At the end of the period ΔT^{CFD}, the new system dynamics have been acquired, and a new POD basis ($\mathbf{\Phi}$) has been calculated from the corresponding matrix of snapshots. Since some of the previous system dynamics are most likely part of the present system evolution, both $\mathbf{\Phi}$ and $\hat{\mathbf{\Phi}}$ (the previous set of modes) need to be combined to create the new updated ROM. This is done by applying POD to the new snapshot matrix $\tilde{\boldsymbol{S}}$ formed as

$$\tilde{\boldsymbol{S}} = \{\hat{\nu}_1\hat{\Phi}_1, \cdots, \hat{\nu}_{N_{old}}\hat{\Phi}_{N_{old}}, \nu_1\Phi_1 \cdots, \nu_{N_{new}}\Phi_{N_{new}}\} \tag{18}$$

where the weights $\hat{\nu}_j$ and ν_j are defined as

$$\hat{\nu}_j = min \left\{ \frac{\hat{D}_{jj}}{\sqrt{\sum_{i=1}^{N_{old}} (D_{ii})^2}}, \frac{\xi^j}{\sqrt{\sum_{i=1}^{N_{old}} (\xi^j)^2}} \right\}, \nu_j = \frac{D_{jj}}{\sqrt{\sum_{i=1}^{N_{new}} (D_{ii})^2}}. \tag{19}$$

In this way, those modes with higher relative "energy" (based on the inner product), and with higher relative contribution to the flow field, are retained in the newly created ROM, and the process is ready to start over again.

3 ENABLERS TO PERFORM 3D SIMULATIONS

As mentioned briefly before, the "POD on the Fly" method has shown very good potential to reduce the cost of aeroelastic simulations. The cost reduction comes from the extraction of patterns present in the flow field, and its use to project the flow field and minimize the residual, but for this cost to be much lower than the full CFD integration, only a subset of the computational mesh must be used. This selection, that is more or less straight forward in 2D simulations, is far from trivial in 3D simulations, at risk of increasing the computational cost if too many cells are chosen. Thus, one of the key enablers to be able to apply the method to such 3D simulations, is the automatic selection of the master cells that will be discussed in the following subsections. Additionally, 3D simulations are typically carried over multiple worker nodes, most likely not even sharing common memory, such that the full flow field vector is rarely assembled together. The new method must adapt to such feature, so that the operations do not require computationally expensive data transfer among nodes or intensive use of memory.

3.1 Automatic master cells selection method

The selection of the master cells, also known as collocation, can be performed in various ways, e.g., equispacedly [4] or concentrated in those spatial regions that are known to be most significant [5] [6]. Such simple selections require to take N somewhat large compared to M (say,

$N = 3M$, [4, 5]). This strategy was selected in the previous paper on the method by the author [7]. As mentioned before, when extending the method to 3D simulations, performed not necessarily on regular meshes, the selection is not evident, and is one of the the main objectives of the work presented in this paper.

More efficient selection methods exist, such as those implemented in the so-called missing point estimation [8], empirical interpolation [9], discrete empirical interpolation [10], and hyper-reduction [11], which use a specific sampling method. Such sampling methods improve the quality of the snapshots by enforcing the selection of spatially and temporally localized patterns.

A new collocation method, intended to be effective, simple and computationally efficient was presented in [12]. The collocation points and snapshots selection is performed by applying to the snapshot matrix $\boldsymbol{S}$ (whose columns are the given snapshots) the LU decomposition (namely, Gauss elimination) with pivoting, based on linear combinations of the columns of $\boldsymbol{S}$. If the snapshots are strongly correlated, the LU decomposition selects a (limited) number of rows of $\boldsymbol{S}$ and a limited number of columns, such that the whole set of matrix columns (i.e., the snapshots) can be reconstructed as linear combinations of the selected columns (i.e., the selected snapshots) using information only from the selected rows, which define the collocation points. The very essence of the performed LU decomposition implies that the reconstruction of the whole set of snapshots using only linear combinations of the selected snapshots is consistent with the expansions (16), while the fact that such reconstructions are based only on the collocation points is consistent with the inner product (4). Since, in the end, the method consists in combining LU decomposition and POD, it will be referred to hereinafter as the LUPOD method.

A set of collocation points, $j_1, \ldots, j_N$, and a set of snapshots, $k_1, \ldots, k_N$, are selected by applying the LU decomposition to the snapshot matrix $\boldsymbol{S}$ in 4 steps, as follows:

1. That element, $S_{j_1 k_1}$, of the matrix $\boldsymbol{S} = [S_{jk}]$ exhibiting the largest absolute value is identified. Using $S_{j_1 k_1}$ as pivot, convenient linear combinations of the columns of $\boldsymbol{S}$ are performed to set to zero the remaining elements of the j_1-th row of $\boldsymbol{S}$.
2. The indexes j_1 and k_1 define the first collocation point and the first selected snapshot, respectively. The k_1-th snapshot is removed from the matrix $\boldsymbol{S}$, obtaining a first modified snapshot matrix $\boldsymbol{S}_1$. Note that the j_1-th row of $\boldsymbol{S}_1$ identically vanishes.
3. Steps 1 and 2 are applied to the matrix $\boldsymbol{S}_1$, which defines a second collocation point, $j_2 \neq j_1$, and a second snapshot, $k_2 \neq k_1$, which is removed from $\boldsymbol{S}_1$, yielding a new modified matrix $\boldsymbol{S}_2$ whose j_1-th and j_2-th rows identically vanish.
4. The procedure iteratively continues, yielding new collocation points, $j_3, \ldots$, new snapshots, $k_3, \ldots$, and new modified matrices, $\boldsymbol{S}_3, \ldots$, until the first iteration step at which

$$\|\boldsymbol{S}_N\|_{\text{Fro}} \,/\, \|\boldsymbol{S}\|_{\text{Fro}} < \varepsilon_{LU}, \tag{20}$$

   for some tunable threshold ε_{LU}. This defines the (common) number of collocation points and selected snapshots, N.

Condition (20) is consistent with (17) and means that the relative RMS error of the approximation of the non-selected snapshots as linear combinations of the selected snapshots is smaller than ε_{LU}.

Once the sets of collocation points and selected snapshots have been identified, truncated POD is applied to the set of selected snapshots, using the inner product (4), based on the collocation points. This is performed by applying standard SVD to the reduced snapshot matrix $\widehat{\boldsymbol{S}}$,

which is defined by considering in the original snapshot matrix $\boldsymbol{S}$ only those rows and columns corresponding to the collocation points and the selected snapshots, respectively.

3.2 ROM updates

One key point of the "POD on the Fly" method is the ability to update the POD modes during the simulation as new system dynamics appear, or simply to reinforce the validity of the POD modes for the current dynamics. However, if the set of master cells is not constant between snapshot sets, a new procedure must be put in place.

For this, the procedure starts by assembling the POD modes matrix, properly weighted as in (18)(19), and then, instead of performing the SVD on the matrix based on the selected master cells, since there is no necessarily a common set of master cells, a new gaussian elimination is performed on the $\tilde{\boldsymbol{S}}$ matrix. This will result in a new set of rows (master cells), and columns (POD modes instead of time steps), on which the SVD would be performed to obtained the new set of mixed POD modes, associated to a new set of master cells valid, in principle, only for the subsequent ROM integration.

3.3 Distributed algorithm

As mentioned previously, the new algorithm must take into account the fact that the full CFD simulation might, and most likely will, be performed distributed among multiple computer nodes. As such, no full fluid state vector should be assembled and the algorithm must be applied avoiding as much communication overhead as possible. This could be achieved in multiple ways, that could be more efficient and will be the target of further work, but the algorithm presented here tries to stick to the original process as much as possible. For this, the following steps are followed:

1. Each worker node assembles the snapshot matrix $\boldsymbol{S}$
2. The gaussian elimination is then performed as a distributed algorithm for which:
   (a) Each worker sends a tuple formed by the maximum absolute value of its snapshot matrix, and the row and the column corresponding to this position, to a coordinator node, say worker_0
   (b) Worker_0 receives the tuple and selects the global maximum, so it sends the tuple with the maximum absolute value to all the workers
   (c) Each worker performs locally the column operations associated with the gaussian elimination, since for this only the column at which the maximum occurs and the maximum absolute value (and its sign) is required
   (d) The worker at which the maximum occurs appends the corresponding cell to a list containing the selected master cells
   (e) The process continues until the overall Frobenius norm is below a threshold value
3. Each worker assembles then the $\hat{\boldsymbol{S}}$ matrix, based on the selected snapshots and the selected cells. Then this matrix, which is much smaller than $\boldsymbol{S}$, is sent to the worker_0
4. Worker_0 concatenates all the $\hat{\boldsymbol{S}}$ matrices, forming a global $\hat{\boldsymbol{S}}$ matrix.
5. Worker_0 performs the POD decomposition on this matrix, performs the mode selection based on the singular values and sends the matrices S and V to each worker
6. Each worker creates the local POD modes $\boldsymbol{\Phi}$
7. The ROM time integration is carried over by worker_0, for which it asks each worker for the necessary mesh information. At each time step the flow field is reconstructed in each

worker to perform the integration of the forces over the body and prepare the solution, should it be necessary to revert the solution to the flow solver

The combination of modes is carried over in a similar way as in point 2.

4 PRACTICAL APPLICATION

The method presented above has been applied first to a NACA 0012 airfoil with 2 degrees of freedom, and then to the AGARD 445.6 wing. Only a reduced set of results is available at the time this paper is written due to a major malfunction of the HPC cluster at which the jobs were run. The following subsections show the corresponding results

4.1 NACA 0012 Airfoil

Let us focus now on the aeroelastic response of a NACA 0012 airfoil, extracted as a section from the B0012 NASA benchmark [13], which has two flexible modes, one consisting in a vertical translation at a frequency of 3.4 Hz, and one consisting in a rotation around the mid point at 5.2 Hz.
The subcritical and supercritical aeroelastic response to an impulse on the torsion mode has been calculated with PYFLOW, an in-house python based parallel CFD solver capable of solving Euler and RANS fluid structure interaction problems, that has embedded the "LUPOD on the Fly" ROM method.
The mesh is composed of approximately 20,000 cells. It can be seen, together with a close-out detail of the airfoil in Figure 2. The fluid has been taken as inviscid, and hence governed by the Euler equations. The loosely coupled problem has been solved with $\Delta t = 0.001s$, $T_o^{CFD} = \Delta T^{CFD} = 0.125s$, $\varepsilon_{POD} = 1E-6$, $K = 100.0$, $\varepsilon_{LU} = 1E-6$.

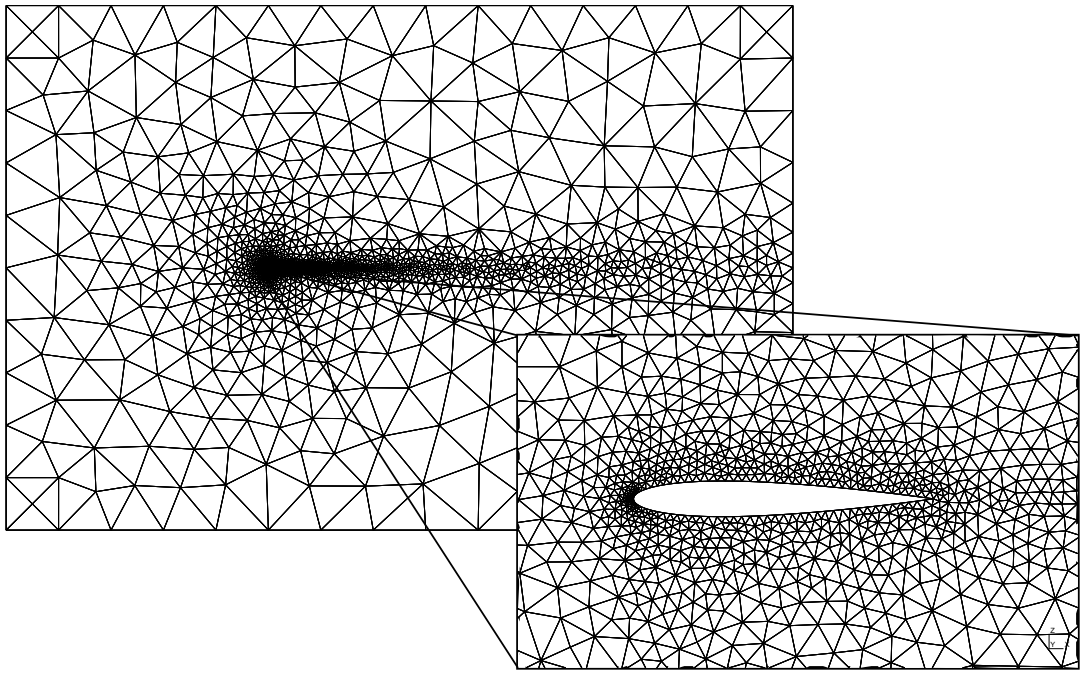

Figure 2: NACA 0012 computational mesh

4.1.1 Simulation at M=0.670

The aeroelastic response using the CFD solver and the "LUPOD on the Fly" based ROM for the case at $M = 0.670$ can be seen in figures 3-4 for a subcritical and supercritical case, respectively. As it can be seen, the ROM is capable of evolving the simulation without having to revert back to the full solver and with minimum response error. The method selects for the last iteration of the subcritical simulation 244 cells and 242 modes, while for the supercritical simulation selects 207 cells and 193 modes. The selected cells for this last case can can be seen in Figure 7.

It must be noted that the deviation from the CFD for the "POD on the Fly" results [7] were smaller, but using a much higher amount of points on the airfoil surface. It is remarkable the fact that, after 1 second of simulation time, it is unclear whether the problem would be stable or unstable, but the ROM is capable of evolving the solution either way without issue. Some updating issues are visible in the subcritical simulation since the ROM is deemed invalid after approximately 2.5 seconds, and further updates do not improve the situation.

4.1.2 Simulation at M=0.800

The aeroelastic response using the CFD solver, and the "LUPOD on the Fly" based ROM for the case at $M = 0.800$ can be seen in figures 5-6 for a subcritical and supercritical case, respectively. Note that for this Mach number, the airfoil exhibits one strong shock both in the upper and the lower surface. The method selects for the last iteration of the subcritical simulation 146 cells and 155 modes, while for the supercritical simulation selects 214 cells and 219 modes. The selected cells for this last case can can be seen in Figure 8. It is interesting to see how the majority of the cells are either on the wake or on the shock location, and almost none on the airfoil surface.

As it occured with the "POD on the Fly" method [7], the shock motion jeopardizes the validity of the ROM, but it still does a good job in adapting and carrying forward the simulation.

4.2 AGARD 445.6 Wing

Next, we focus on the aeroelastic response of the AGARD 445.6 wing, weakened-3 model. The model has 4 flexible modes, namely first wing bending (9.6 Hz), first wing torsion (38.17Hz), second wing bending (48.4 Hz) and second wing torsion (91.5Hz). For more details on the model, please refer to [14]. The supercritical aeroelastic response to an impulse on the first wing torsion mode has been calculated with PYFLOW. The mesh is composed of approximately 200,000 cells. A rather coarse wake refinement has been applied to be able to run the case, given the hardware issues mentioned above. The mesh can be seen in Figure 9. The fluid has been taken as inviscid, and hence governed by the Euler equations. The loosely coupled problem has been solved with $\Delta t = 0.0002s$, ${T_o}^{CFD} = \Delta T^{CFD} = 0.025s$, $\varepsilon_{POD} = 1E-6$, $K = 100.0$, $\varepsilon_{LU} = 1E-6$.

4.2.1 Simulation at M=0.678

The aeroelastic response using the CFD solver, and the "LUPOD on the Fly" based ROM can be seen in Figure 10. The method selects for the last iteration of the simulation 1074 cells and 1040 modes. The selected cells for this last case can can be seen in Figure 11.

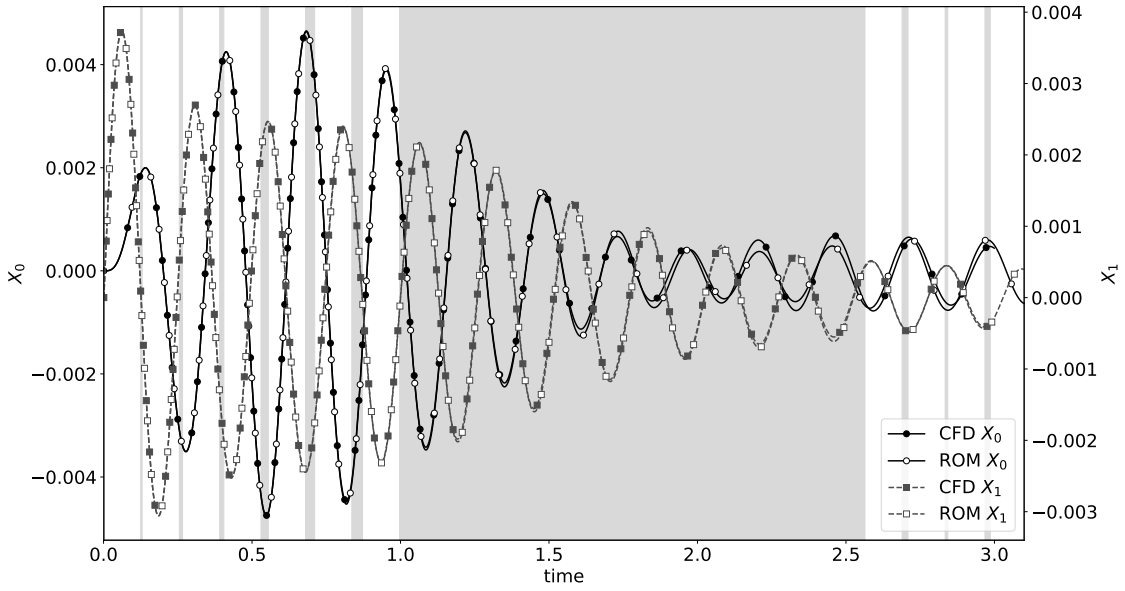

Figure 3: N0012 subcritical aeroelastic response at $M = 0.670$. Gray areas correspond to intervals where the ROM has been taken over the simulation

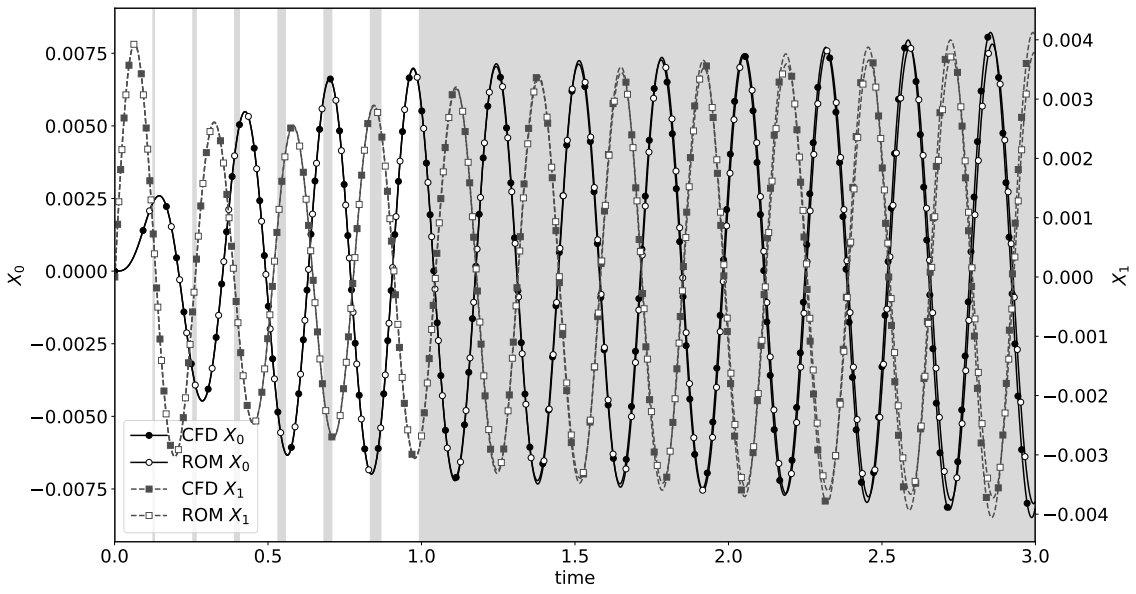

Figure 4: N0012 supercritical aeroelastic response at $M = 0.670$. Gray areas correspond to intervals where the ROM has been taken over the simulation

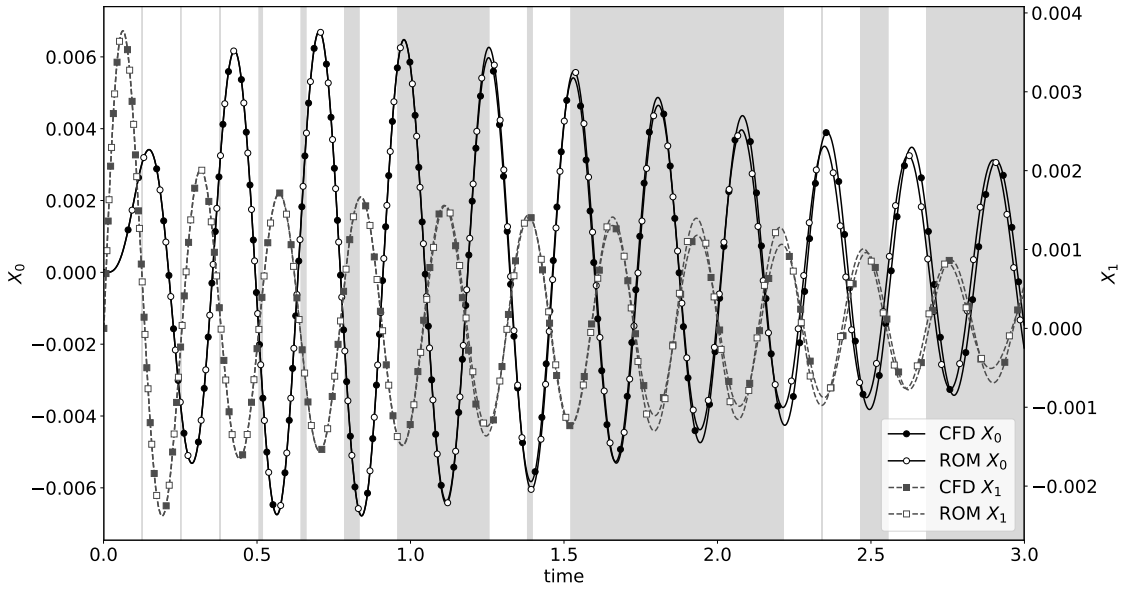

Figure 5: N0012 subcritical aeroelastic response at $M = 0.800$. Gray areas correspond to intervals where the ROM has been taken over the simulation

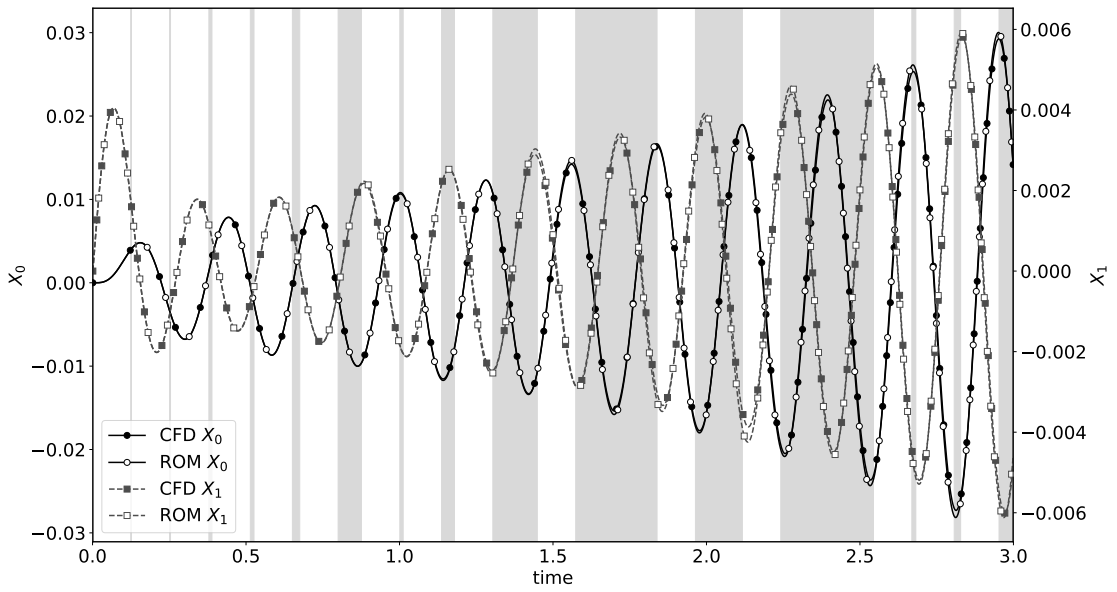

Figure 6: N0012 supercritical aeroelastic response at $M = 0.800$. Gray areas correspond to intervals where the ROM has been taken over the simulation

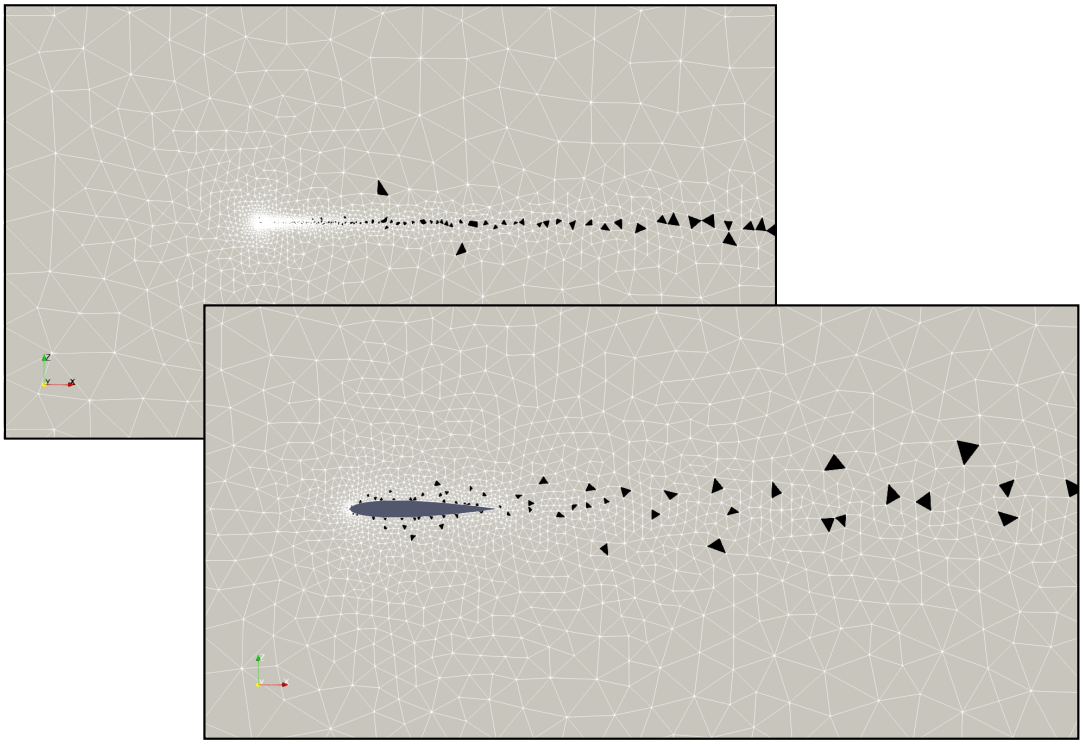

Figure 7: N0012 selected cells for the $M = 0.670$ supercritical case, in black. Global view and airfoild close-out

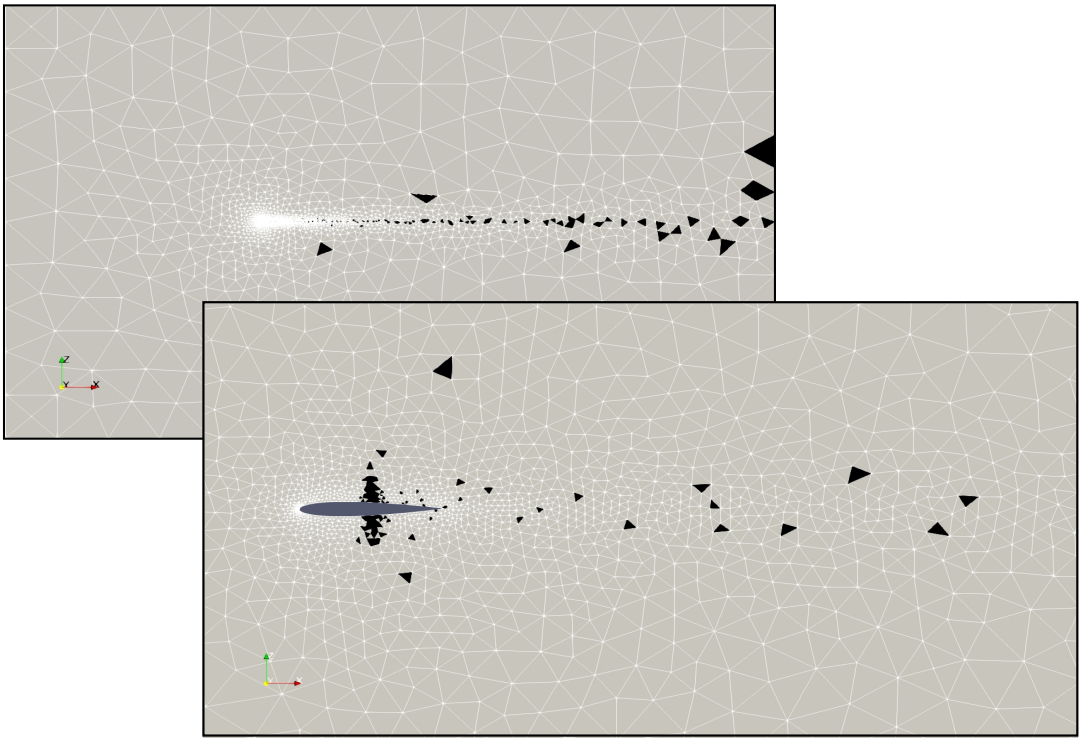

Figure 8: N0012 selected cells for the $M = 0.800$ supercritical case, in black. Global view and airfoild close-out

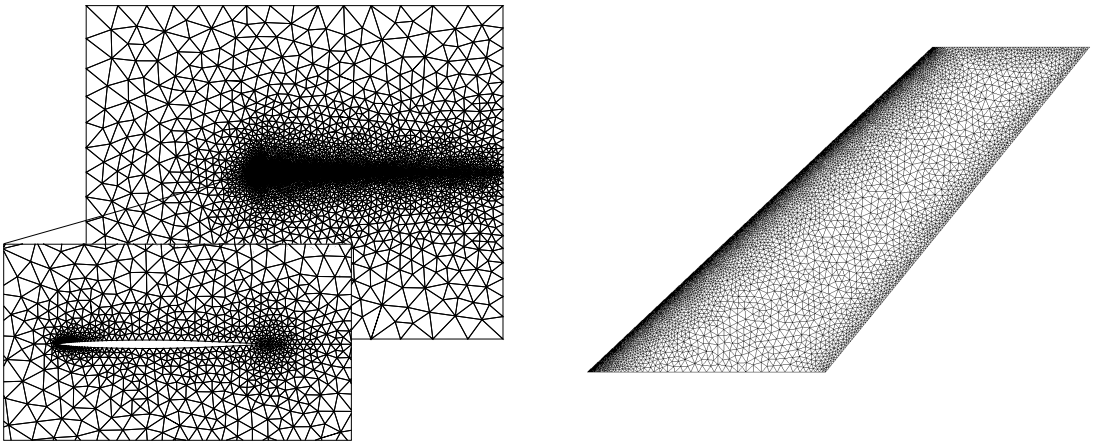

Figure 9: AGARD 445.6 computational mesh

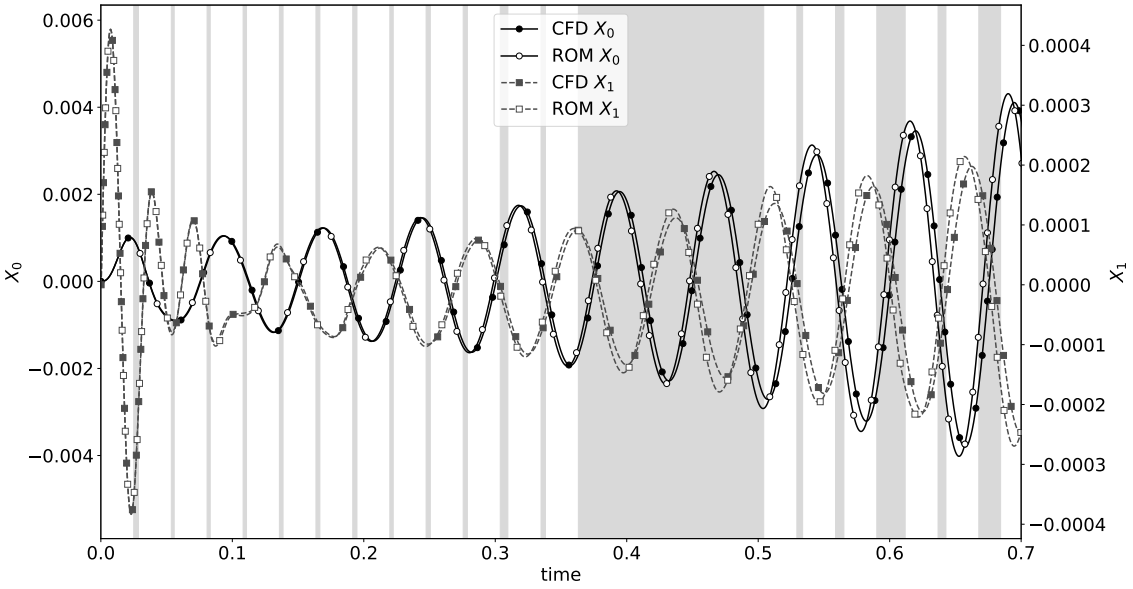

Figure 10: AGARD 445.6 supercritical aeroelastic response at $M = 0.678$. Gray areas correspond to intervals where the ROM has been taken over the simulation

For this case, the ROM extends its validity in time, up to a point at which the ROM is capable of evolving the simulation for a time much higher than ΔT^{CFD}, but once it reverts back to the full solver has problems with the mode updating. The ROM accuracy is lower than in the NACA 0012 case, most likely due to the lack of points over the wing surface. One plausible explanation for this is the coarseness of the employed mesh, since the cell size growth from the wing surface normal to the wing and along the wake prevents good convergence of the full solver. Nonetheless, the case would certainly benefit also from fine-tuning the method parameters, for which more simulations would be required.

5 CONCLUSIONS AND FUTURE WORK

The paper presents a new method, called here "LUPOD on the Fly", that enables the application of the formerly presented "POD on the Fly" method [7] to 3D configurations, since it incorporates into it the automatic cell selections, as well as the possibility to work on simulations distributed over several computer nodes. The method performs remarkably well for the presented simulations on a NACA 0012 airfoil, with some small issues present on the mode

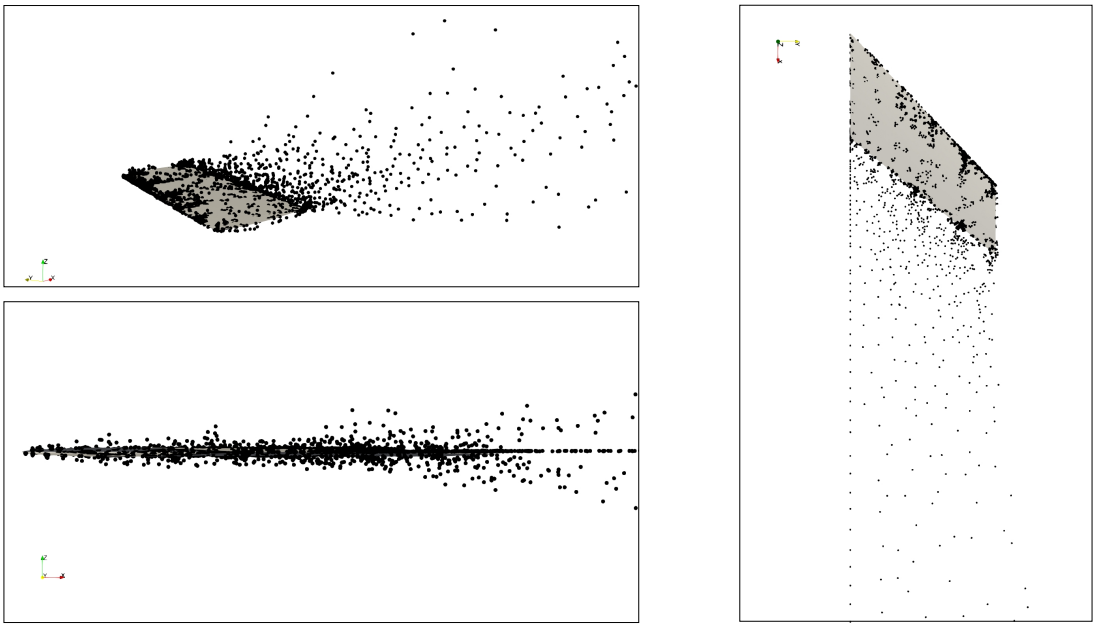

Figure 11: AGARD 445.6 selected cells for the $M = 0.678$ supercritical case, indicated as black points

updating that would require further work to be solved.

When applied to the AGARD 445.6 wing, the method behaves as expected, but the accuracy and mode updating is slightly degraded. This is the only simulation available thus far due to hardware issues, but it has shown enough potential to increase the efficiency of computational aeroelastic simulations to encourage further investigations, that will be the focus of future publications.

Apart from fine tuning the currently presented method, which includes the application to viscous simulations, the future work will aim at increasing its computational efficiency. This is due to the fact that the gaussian elimination computational cost keeps increasing as more modes need to be mixed, hence possibly benefiting for local (in each worker) down selection of the candidate collocation cells.

6 REFERENCES

[1] Heeg, J., Chwalowski, P., Florance, J. P., et al. (2013). Overview of the aeroelastic prediction workshop. In *51st AIAA Aerospace Sciences Meeting*. Grapevine, TX, United States.

[2] Moreno, R., Taylor, P., and Newsom, J. (2012). A rigid horizontal tail wind tunnel test for high transonic mach and high frequency unsteady pressure acquisition. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*. Honolulu, HI, United States.

[3] Rapún, M.-L., Terragni, F., and Vega, J. (2015). Adaptive pod-based low-dimensional modeling supported by residual estimates. *International Journal for Numerical Methods in Engineering*, 104(9), 844–868.

[4] Alonso, D., Vega, J., Velazquez, A., et al. (2012). Reduced-order modeling of three-dimensional external aerodynamic flows. *Journal of Aerospace Engineering*, 25, 588–599. doi:10.1061/(ASCE)AS.1943-5525.0000148.

[5] Terragni, F., Valero, E., and Vega, J. (2011). Local pod plus galerkin projection in the unsteady lid-driven cavity problem. *SIAM Journal on Scientific Computing*, 33, 3538–3561.

[6] Moreno-Ramos, R., Vega, J. M., and Varas, F. (2016). Computationally efficient simulation of unsteady aerodynamics using pod on the fly. *Fluid Dynamics Research*, 48(6), 061424.

[7] Moreno-Ramos, R., Vega, J. M., and Varas, F. (2017). Self adaptive pod based rom aeroelastic simulations. In *17th International Forum on Aeroelasticity and Structural Dynamics (IFASD 2017)*. International Forum on Aeroelasticity and Structural Dynamics, pp. 1487–1505.

[8] Astrid, P., Weiland, S., Willcox, K., et al. (2008). Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10), 2237–2251.

[9] Barrault, M., Maday, Y., Nguyen, N. C., et al. (2004). An empirical interpolation-method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9), 667–672.

[10] Chaturantabut, S. and Sorensen, D. C. (2010). Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5), 2737–2764.

[11] Ryckelynck, D. (2009). Hyper-reduction of mechanical models involving internal variables. *International Journal for Numerical Methods in Engineering*, 77, 75–89.

[12] Rapún, M.-L., Terragni, F., and Vega, J. (2017). Lupod: Collocation in pod via lu decomposition. *Journal of Computational Physics*, 335, 1–20.

[13] Bennett, R. M. (2000). Test cases for flutter of the benchmark models rectangular wings on the pitch and plunge apparatus. Tech. rep., NATIONAL AERONAUTICS AND SPACE ADMINISTRATION HAMPTON VA AEROELASTICITY BR .

[14] Yates Jr, E. C. (1987). Agard standard aeroelastic configurations for dynamic response. candidate configuration i.-wing 445.6.

COPYRIGHT STATEMENT