Mesh Optimization for Improved Computational Fluid Dynamics Numerical Stability and Convergence Rate

Mohammad Zandsalimy* and Carl Ollivier-Gooch†
The University of British Columbia, Vancouver, British Columbia, V6T 1Z4

A novel mesh optimization approach is utilized in conjunction with the Ansys Fluent solver for numerical stability and convergence rate enhancement of computational fluid dynamics simulations. This method leverages the dynamic mode decomposition of solution update vectors for solution mode identification. Through this data reduction technique, the large-scale linear evolution system is mapped onto a smaller space with substantially fewer degrees of freedom for stability analysis at a negligible fraction of the overall computational cost. The eigenanalysis of the small-scale matrix facilitates the identification of dominant solution modes during the simulation. This mesh optimization technique leverages the gradients of the problematic solution modes with respect to local changes of the mesh to calculate proper modification vectors for a small collection of nodes. These modifications lead to the improved numerical stability of the simulation. Employing the Ansys Fluent CFD package as the primary finite-volume solver, our study demonstrates the complete non-invasiveness of the presented mesh optimization approach, requiring no access to the underlying software architecture. The results presented herein illustrate the feasibility and efficacy of this mesh optimization technique in improving numerical stability and convergence rate, showcasing its compatibility with third-party flow solvers.

I. Introduction

Despite over half a century of progress in Computational Fluid Dynamics (CFD), the scalability and effectiveness of contemporary industrial CFD packages continue to be impeded by numerical stability issues. Challenges such as slow converging solutions or unstable simulations may arise from flawed discretization schemes, inaccurate time integration, or suboptimal mesh quality, among other potential factors. Recent investigations have explored strategies to enhance numerical stability and convergence rate in CFD applications through targeted modifications to local mesh areas [1]. In this study, the authors employed eigenanalysis of the Jacobian matrix to identify problematic solution modes. By adjusting a small subset of nodes in the mesh, they successfully improved the numerical stability and convergence rate of their in-house finite-volume solver. Additionally, the mesh optimization approach required frequent human intervention for critical decision-making necessary for a successful outcome.

To tackle these challenges, Zandsalimy and Ollivier-Gooch [2, 3] introduced an innovative approach for identifying numerical solution modes, aiming to replace the computationally demanding eigenanalysis module. This method leveraged anomaly detection machine learning to analyze either the residual vector or the solution modes obtained through Principal Component Analysis (PCA) of solution update vectors. The authors proposed a new technique for generating synthetic vectors from anomalous residuals or PCA vectors, closely resembling the eigenvectors of the Jacobian matrix. By doing so, they successfully eliminated the need for the computationally intensive eigenanalysis of the Jacobian matrix in solution mode identification. Additionally, this new approach completely removed the requirement of human decision-making in the mesh optimization process.

Expanding upon this research, Zandsalimy and Ollivier-Gooch [4] introduced a novel, computationally cheaper, and reliable mesh optimization method compared to the previous approach. Their methodology involved leveraging Dynamic Mode Decomposition (DMD) on solution update vectors to identify numerical solution modes, a notably more efficient alternative to the computationally intensive eigenanalysis of the Jacobian matrix. The mesh optimization approach presented was also fully automatic without the need for human involvement during the process. They successfully tested this approach on the Advanced Numerical Simulation Library (ANSLib) finite-volume solver [5], and demonstrated improved numerical stability and convergence rate across diverse simulations.

*Ph.D., Mechanical Engineering, The Advanced Numerical Simulation Laboratory.
†Professor, Mechanical Engineering, The Advanced Numerical Simulation Laboratory, AIAA Associate Fellow.

Following the methodology introduced by [4], we adopt the dynamic mode decomposition of solution update vectors for solution mode identification and mesh optimization. This approach stands out as the preferred choice for optimizing finite-volume simulations, thanks to its unparalleled computational efficiency, reliability, and scalability, particularly in handling complex simulations. To underscore the applicability and user-friendly nature of this mesh optimization approach, we apply it alongside Ansys Fluent, an industrial solver widely embraced in scientific and engineering communities. As revealed in our findings, the presented approach treats the flow solver as a black box, requiring no access to the underlying software architecture. The results showcased underscore the feasibility and efficacy of this mesh optimization technique in large-scale CFD simulations, with an example case encompassing over 21.7 million degrees of freedom in the Fluent solver.

II. Background

A. Numerical Simulation Scheme

The present study employs Ansys Fluent version 2023 R2 as the finite-volume fluid flow simulation package. Fluent, a widely adopted industrial solver, provides a robust graphical user interface that facilitates the examination of intricate flow characteristics and behavior. The incorporation of User Defined Functions (UDF) enables software customization, significantly improving workflow flexibility. The software employs a finite-volume numerical discretization scheme with second-order accuracy to solve the Partial Differential Equations (PDE). Fluent typically restricts access to its underlying software architecture under normal circumstances. Consequently, our mesh optimization technique must operate as a fully non-invasive process, devoid of the need for such permissions. As demonstrated in this study, the presented mesh optimization technique seamlessly integrates with an external solver, necessitating only access to the solution vector and partial access to the Jacobian matrix, both of which do not mandate access to the underlying software architecture.

B. Stability Improvement

We have developed a comprehensive numerical stability improvement approach with a versatile framework suitable for diverse applications. This framework comprises distinct modules, each offering flexibility for employing various methods of user's choice. An overview of this algorithm is depicted in Figure 1. An initial crucial step in implementing this algorithm is determining the appropriate solution iteration for initiating the stability improvement process. It is noteworthy that unstable solution modes arising from initial conditions tend to dissipate naturally within the early iterations of the solver, provided a suitable time integration method is employed. Consequently, we advise against prematurely applying the stability improvement approach in the simulation process to conserve computational resources. This decision-making process can be either user-driven, allowing manual intervention, or entirely automated, eliminating the need for human involvement.

In the subsequent stage, a proper solution mode analysis tool can be utilized to identify the dominant solution modes. Various techniques can be employed for this purpose, including full or partial eigenanalysis of the Jacobian matrix [1, 6], residual vector analysis [2], principal component analysis [3], and dynamic mode decomposition of solution update vectors [4, 7]. All of these methods have undergone rigorous examination and testing, and have been successfully integrated into this novel stability improvement approach. The dominant solution modes carry vital information regarding the dynamic response of the linear system including magnitude growth rate and oscillation frequency. Consequently, it becomes possible to pinpoint unstable or slowly converging solution modes for targeted remediation in subsequent steps of the algorithm. The objective is to strategically modify an independent parameter within the simulation, exerting sufficient influence on the identified mode. This modification aims to enhance the mode's numerical stability and convergence behavior. In the context of linear systems resulting from the discretization of partial differential equations, examples of such influential parameters include the discretization scheme, reconstruction method, time step size, mesh vertex locations, and mesh refinement.

Subsequently, the gradients of the identified solution modes are computed with respect to variations in the independent parameter. Essentially, these gradients enable us to quantify changes to the solution modes corresponding to changes to the specified independent parameter. Consequently, we derive modification vectors that facilitate favorable adjustments to the problematic solution modes in the simulation. Implementing these calculated modifications to the independent parameter enhances the numerical stability of the linear operator at this stage. However, it is imperative to note that this method primarily enhances the local stability of the problem. Given the inherent non-linearity of substan-

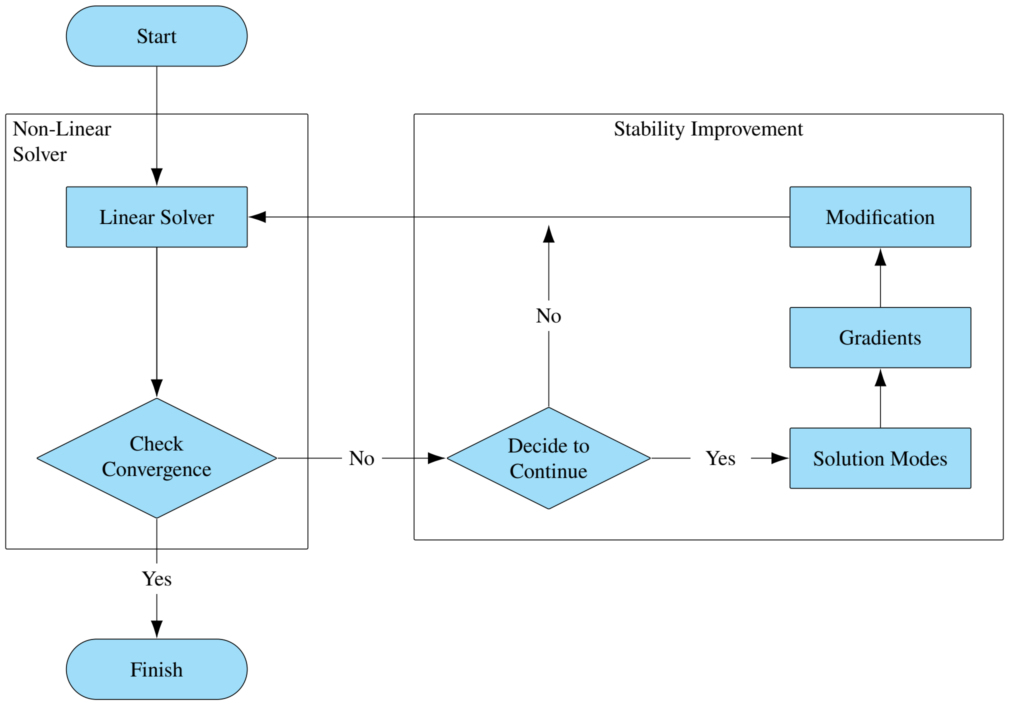


Fig. 1 The general stability improvement approach coupled with the CFD solver

tial CFD simulations, this process may need to be repeated a few times before achieving complete non-linear numerical stability.

III. Methodology

The general mesh optimization approach for improved stability and convergence rate of CFD simulations presented in Section II.B is highly flexible, providing diverse opportunities for customization to different applications. Zandsalimy and Ollivier-Gooch [4] exemplified a mesh optimization technique based on this general approach, showcasing remarkable computational efficiency and scalability in handling complex simulations. This specific approach employs dynamic mode decomposition of solution update vectors for solution mode identification. The working details of this approach are presented as follows.

A. Dynamic Mode Decomposition

Dynamic mode decomposition is a data reduction method to quantify the dynamic solution behavior from a selection of solution snapshots [8]. Let $\boldsymbol{x}_i \in \mathbb{R}^m$ be the solution vector at iteration i in which m is the number of degrees of freedom. To extract the time dynamics, DMD assumes a time-invariant linear mapping $\boldsymbol{A}$ between two subsequent solution snapshots as,

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k \tag{1}$$

We form an order-n Krylov subspace of $\boldsymbol{A}$ generated by x_1 as,

$$\mathcal{K}_1^n(\boldsymbol{A}, \boldsymbol{x}_1) = \{\boldsymbol{A}^0\boldsymbol{x}_1, \boldsymbol{A}^1\boldsymbol{x}_1, \boldsymbol{A}^2\boldsymbol{x}_1, \dots, \boldsymbol{A}^{n-1}\boldsymbol{x}_1\} \tag{2}$$

Forming a different order-n Krylov subspace $\mathcal{K}_2^n(\boldsymbol{A}, \boldsymbol{x}_2)$, we can write the following equation,

$$\mathcal{K}_2^n(\boldsymbol{A}, \boldsymbol{x}_2) = \{\boldsymbol{x}_2, \boldsymbol{x}_3, \dots, \boldsymbol{x}_{n+1}\} = \boldsymbol{A}\mathcal{K}_1^n = \mathcal{K}_1^n\boldsymbol{B} + \boldsymbol{r} \tag{3}$$

Here, $\boldsymbol{r}$ is a residual vector. The matrix $\boldsymbol{B}$, which is an approximation of the linear mapping projected onto a much smaller space, can be determined by minimizing the Frobenius norm of the residual as,

$$\boldsymbol{B} = \arg\min_{\hat{\boldsymbol{B}}} \|\mathcal{K}_2 - \mathcal{K}_1\hat{\boldsymbol{B}}\| \tag{4}$$

There exists a different method of computing a modified form of $\boldsymbol{B}$ instead of this optimization problem. The singular value decomposition of the matrix $\mathcal{K}_1$ can be used for this purpose. In this approach, $\mathcal{K}_1$ is decomposed as,

$$\mathcal{K}_1 = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^H \tag{5}$$

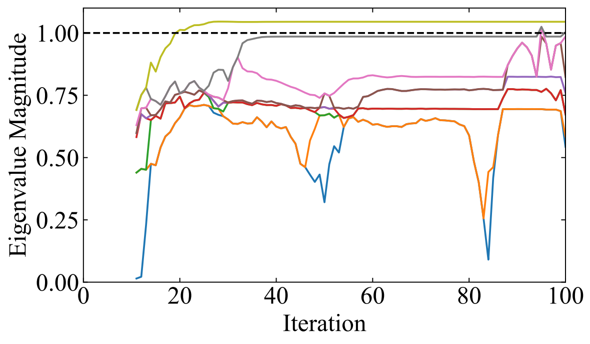
in which, $\boldsymbol{\Sigma}$ is a diagonal matrix containing the singular values while $\boldsymbol{U}$ and $\boldsymbol{V}$ contain the left and right singular vectors, respectively. Here, $(\cdot)^H$ denotes the Hermitian transposed of a matrix. To generate the modified small-scale representation of the linear operator $\tilde{\boldsymbol{B}} = \boldsymbol{U}^H\boldsymbol{A}\boldsymbol{U}$, we can substitute Equation 5 in Equation 3 and rearrange to get,

$$\tilde{\boldsymbol{B}} = \boldsymbol{U}^H\mathcal{K}_2\boldsymbol{V}\boldsymbol{\Sigma}^{-1} \tag{6}$$

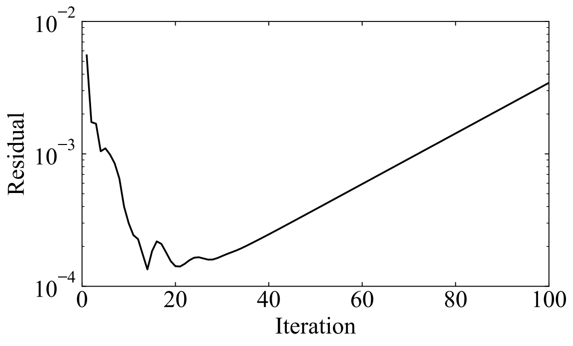
The eigenvalues of $\tilde{\boldsymbol{B}}$ and $\boldsymbol{A}$ are the same and the eigenvectors of the original linear mapping $\boldsymbol{w_A}$ can be computed using the eigenvectors of the projected space $\boldsymbol{w_{\tilde{B}}}$ as,

$$\boldsymbol{w_A} = \mathcal{K}_2\boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{w_{\tilde{B}}} \tag{7}$$

DMD provides highly accurate time dynamics of solution modes when applied to a small selection of solution update vectors. This makes DMD a desirable approach from both accuracy as well as computational requirement points of view. To test this, the 3-dimensional Advection problem is solved on a $3 \times 1 \times 1$ square channel with 754 tetrahedral control volumes. DMD is performed on the 10 most recent solution update vectors. The DMD eigenvalue magnitude history is shown in Figure 2a in which each color depicts the history of a different eigenvalue with solution iteration. As seen, there is a single unstable DMD mode with a magnitude larger than 1.0 at iteration 20 and onward. Before this solution iteration, non-linear effects resulting mainly from the initial conditions dominate the solution which is why the unstable solution mode takes a few iterations to show up in the DMD analysis. In other words, a few iterations are required for the numerical solution to dissipate the effects of the initial conditions and arrive at the correct solution orbit. The residual history of this problem shown in Figure 2b depicts a log-linear growth rate after iteration 30 of this unstable problem. In the divergence section, the residual ratio at two subsequent iterations is 1.04495747. The magnitude of the sole unstable DMD mode after iteration 30 is 1.04498849. These two values have a relative difference of 0.003% which shows an accurate residual growth rate prediction by the DMD analysis on the solution update vectors.



(a) DMD Eigenvalue Magnitude



(b) Residual History

Fig. 2 DMD eigenvalue magnitude predicting residual growth rate in a 3D Advection problem

In another test for two different simulations, DMD is performed on the 10 most recent solution update vectors and the magnitudes of $\tilde{\boldsymbol{B}}$ eigenvalues are shown in Figure 3 in which the changes to each eigenvalue are presented with a different color. Figure 3a depicts the results for a stable Burgers problem where the magnitudes of $\tilde{\boldsymbol{B}}$ eigenvalues stay smaller than 1.0. Figure 3b shows the results of an unstable Burgers problem in which some eigenvalues have a magnitude larger than 1.0. DMD eigenvalue magnitude indicates the behavior of the dominant solution modes which can be used to automatically start the mesh optimization module without the requirement for human intervention that was a prerequisite for the approach in [1].

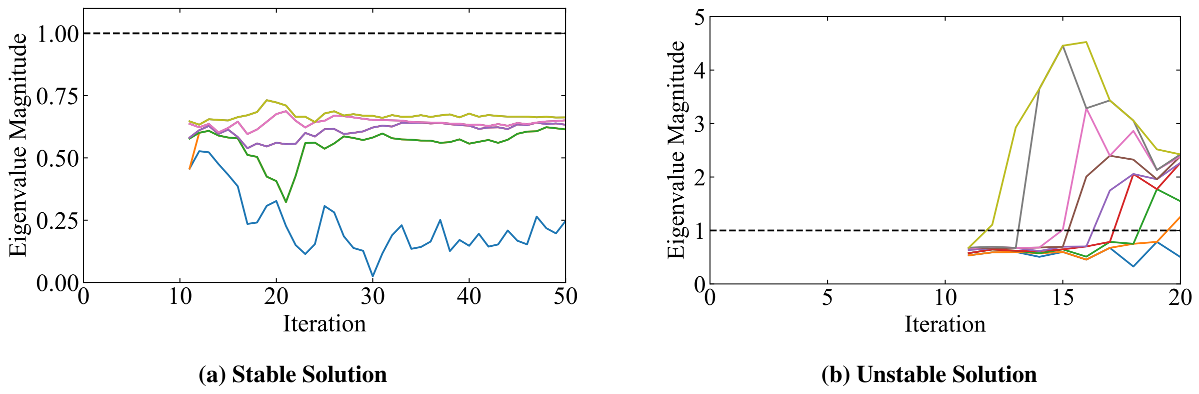


Fig. 3 Eigenvalue magnitude of $\tilde{B}$ for two example Burgers problems [4]

The possibility of large computational savings is another important aspect of using DMD in the mesh optimization algorithm. The eigenanalysis module was the most computationally expensive component of the approach presented by [1] with a cost of $O(m^3)$ for a simulation with m degrees of freedom. Conversely, using a small number of solution update vectors ($n \ll m$) in the DMD approach, we get a computational cost of $O(mn^2)$ which is a substantial improvement over the previous approach.

Performing DMD on a collection of the most recent solution update vectors provides the dominant eigenmodes of the simulation. The eigenvalue magnitude shows the growth rate for each solution mode. As a result, the unstable modes can be identified with a magnitude greater than 1.0 and selected for further analysis during the mesh optimization procedure. However, in some cases, the simulation can include stable eigenmodes with a magnitude close to 1.0 that would dominate the simulations with a slow converging behavior. In such cases, the slow converging modes that have a magnitude smaller than 1.0 can also be selected as eigenvalues of interest for modification. Examples of unstable solution modes during DMD analysis of different problems are presented in Figures 2 and 3. Further, the mesh optimization module can be started as soon as eigenmodes of interest are detected for a fully automated mesh optimization procedure.

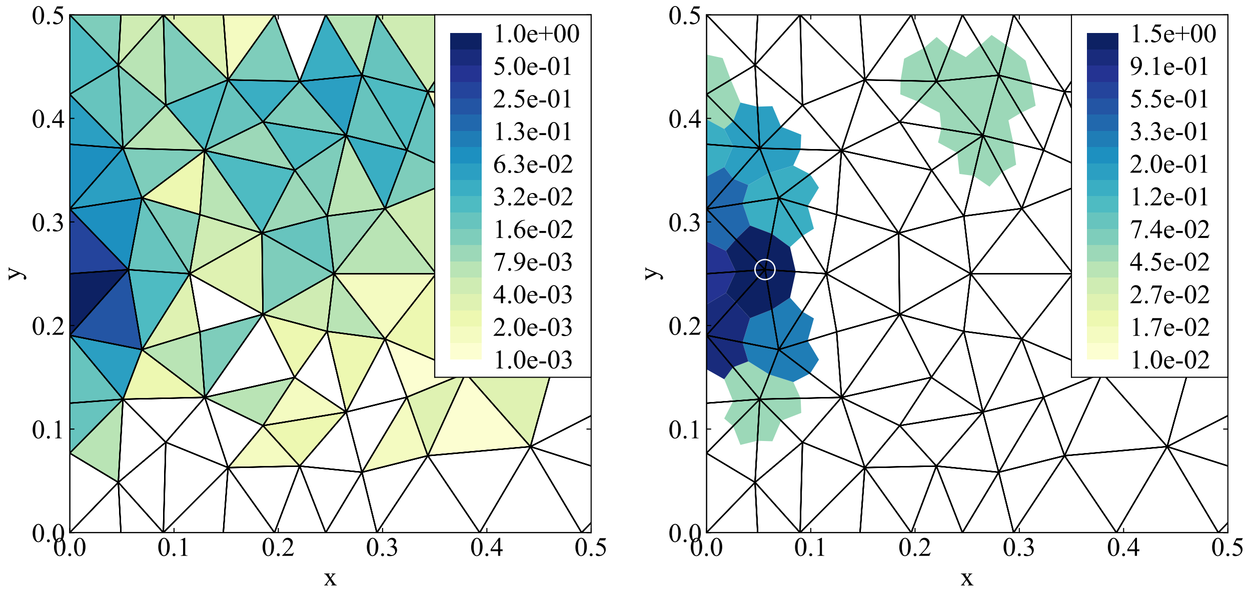
B. Cell and Vertex Selection

After identifying the unstable solution modes in the simulation, we can change one or more independent variables in the simulation with enough influence on the selected eigenmodes to improve the stability and convergence behavior of the numerical solution. The physics, boundary conditions, reconstruction method, time-step size, mesh topology, refinement, and vertex locations all can have considerable changes to a given numerical mode. In the current study, we narrow our focus to mesh vertex locations for unstable solution mode modification. In this approach, a collection of vertices on the mesh are selected whose locations have the highest possible effect on the eigenmodes in question. Through calculated changes in vertex locations, we can induce favorable changes to the selected eigenmodes for better stability and convergence behavior.

Similar to the approach by [1], we select a single vertex on the mesh to stabilize each unstable solution mode. Proper vertex selection involves calculating the gradients of each eigenmode with respect to the movement of all vertices on the mesh. The vertex with the largest gradient is the correct choice to make favorable changes to each unstable eigenmode. However, there is usually a handful of vertices with large enough gradients that can be used to modify each eigenmode. This can be convenient when we have already selected a vertex for a different mode or when vertex movement is limited beyond the calculated gradients for mesh modification. This vertex selection method will require the solution to the eigenproblem and can be computationally demanding. As a result, we adopt a different approach to vertex selection presented by [1] which avoids any unnecessary numerical overhead by cleverly employing the unstable numerical eigenvectors in the solution.

The unstable numerical eigenvectors are highly local to certain areas in the mesh with only a few non-zero values [1]. As a result, these vectors are pointing to local areas on the mesh in which the unstable mode is growing out of control and resulting in solution divergence. This fact can be used for vertex selection on the mesh at a small computational cost. In this approach, we lay out the absolute values of the unstable numerical eigenvector on the mesh and for each vertex, the sum of values in the adjacent cells is computed as a selection proxy. Our past experiments [1] have shown that this vertex-centered vector usually points directly to the vertex that has the largest eigenmode gradient. This approach results in the selection of a single vertex to modify each unstable solution mode.

The vertex selection procedure is depicted for an example Burger problem in Figure 4. Figure 4a shows the absolute value of the sole unstable DMD eigenvector in the simulation. This vector is highly local to a certain area on the mesh in which the solution mode experiences the largest growth rate. The summation of DMD vector values in the adjacent cells of each vertex is shown in Figure 4b. The maximum value of this new vector is pointing to a vertex shown with a white circle which is selected for modification in the next steps of the presented methodology. In fact, this vertex has the largest eigenvalue gradient for the eigenmode in question. Note that in calculating the vertex-centered vector of Figure 4b we ignore the small components in the DMD vector (magnitude smaller than 5% of the maximum entry).

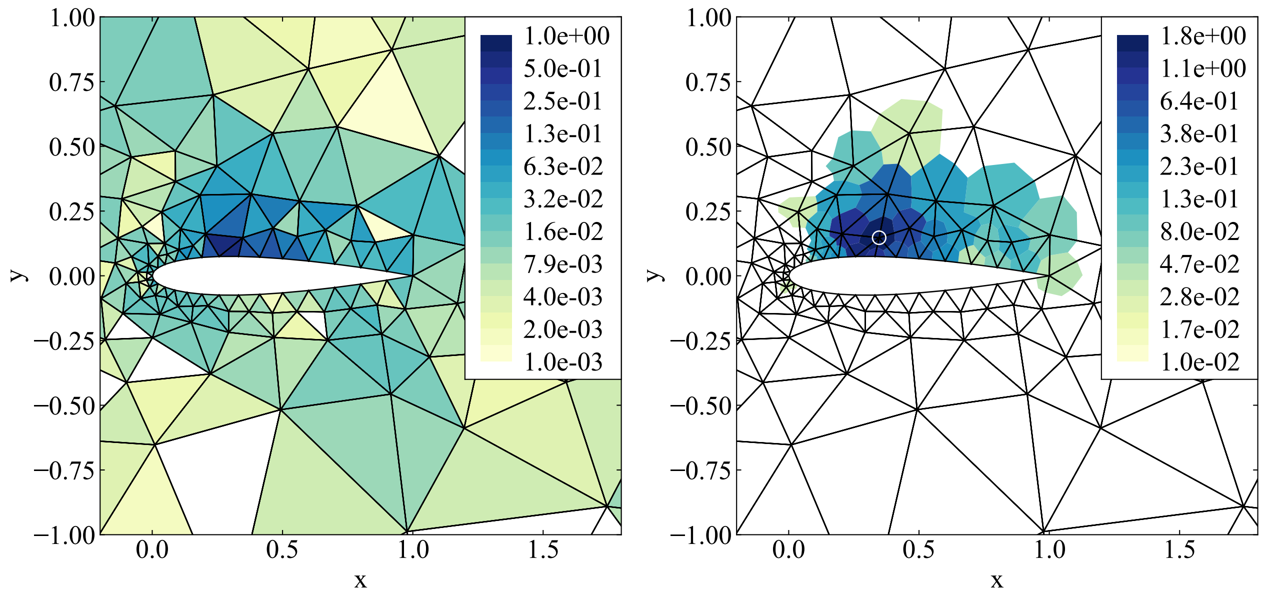


(a) Absolute value of DMD eigenvector

(b) Selection weight measure

Fig. 4 Vertex selection in a Burgers problem (note the logarithmic colormap scale) [4]

In the next test, the absolute value of the DMD eigenvector corresponding to an unstable mode in an Euler problem is shown in Figure 5a. Once again, the selected solution mode is highly localized on the mesh pointing to the area where numerical instabilities are emerging. The vertex-centered selection weight measure is computed for the significant entries of the DMD vector and shown in Figure 5b. The vertex indicated with a white circle is selected for modification in the mesh optimization process.



(a) Absolute value of DMD eigenvector **(b) Selection weight measure**

Fig. 5 Vertex selection in an Euler problem (note the logarithmic colormap scale) [4]

C. Movement Vector Calculation

After the vertex selection stage, proper modification vectors need to be calculated to adjust the mesh for favorable changes to the eigenmodes in question. Prior studies utilized the gradients of eigenvalues with respect to vertex movement for this purpose which was computationally expensive, requiring eigenvectors and partial recalculation of the Jacobian matrix. In the current study, however, we utilize the gradients of the Jacobian matrix entries directly to find proper modification vectors. According to the Gershgorin circle theorem [9], the set of eigenvalues of a given matrix is located inside the union of a collection of specific discs in the complex plane. This theorem associates a disc to each row of the matrix centered at the diagonal entry with a radius equal to the sum of absolute values of the off-diagonal entries. Equation 8 presents a simple Jacobian matrix $\boldsymbol{E}$, the Gershgorin circles of which are presented in Figure 6 in cyan with centers depicted by black Xs. Without any knowledge of the exact eigenvalues of this matrix and only looking at the union of the circles, we conclude that this matrix may have some eigenvalues with positive real parts. Such eigenvalues correspond to unstable solution modes with positive exponential growth. Figure 6 also depicts the eigenvalues of the example matrix $\boldsymbol{E}$ with red stars. As seen, the eigenvalues are located in the union of the Gershgorin circles.

$$\boldsymbol{E} = \begin{bmatrix} -11 & -2 & 0 & 1 \\ 0.5 & -7 & 0.5 & 0.5 \\ 0 & 1 & -2 & 0 \\ -1 & -1 & -1 & 5 \end{bmatrix} \tag{8}$$

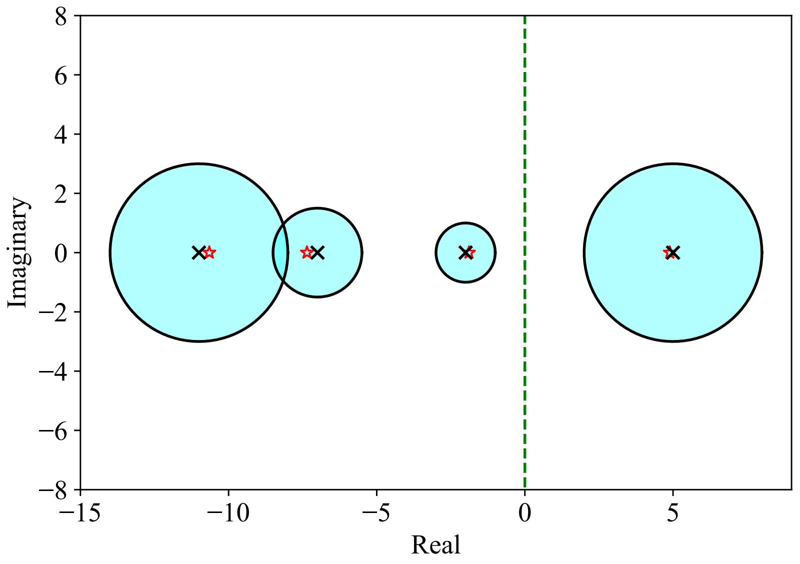


Fig. 6 Example Gershgorin circles

In the novel movement vector calculation scheme, we aim to improve the eigenspectrum of the Jacobian matrix by making favorable changes to the Gershgorin circles associated with the problematic cells. In other words, proper changes to vertex location are calculated to push the circles on the right side of the spectrum as far to the left as possible. This approach does not guarantee that all the problematic eigenmodes are going to lie in the left half-plane after mesh modification but it is a positive step toward numerical stability. Significant computational savings can be achieved through this novel approach which is an improvement over the previous studies. This technique essentially boils down to changing the mesh in order to make the Jacobian matrix more diagonally dominant. Reducing the time-step size is a more traditional approach to fixing numerical instabilities which also results in better diagonal dominance of the linear system of equations.

The current approach involves computing the gradients of the diagonal elements of the Jacobian matrix with respect to vertex movement on the problematic rows. This method only requires the computation of a few entries in the Jacobian matrix which makes the approach highly computationally efficient and avoids forming the entire matrix. To calculate the gradient of the (i, j) entry in the Jacobian with respect to mesh movement, $\frac{dJ_{ij}}{d\zeta}$, we use the finite difference method as,

$$\frac{dJ_{ij}}{d\zeta} \approx \frac{\partial J_{ij}}{\partial \zeta} = \frac{J_{ij}(\zeta + \delta\zeta) - J_{ij}(\zeta)}{\delta\zeta} \tag{9}$$

D. Vertex Modification

At this point, the required information for proper vertex modification to improve the numerical stability or convergence behavior of the simulation is available. However, it is crucial to establish limits on mesh vertex movements to prevent the formation of defective cells with negative volumes. To address this, a characteristic length scale, such as the shortest edge incident on each vertex, can be defined and utilized to restrict vertex displacement. This concern becomes particularly pertinent when dealing with meshes containing high aspect ratio and highly skewed control volumes, as observed in boundary layer and turbulence problems. These applications often feature cells with high aspect ratios, posing challenges in mesh modification. In such cases, there exists a threshold beyond which a single vertex cannot be moved without resulting in a defective control volume.

Moreover, in adaptive mesh modification approaches, error minimization techniques are employed to adjust elements [10, 11], which may yield conflicting vertex movement vectors when compared to optimization methods. Fortunately, in most scenarios, movement vectors derived from mesh optimization are considerably smaller than those from adaptive methods. This discrepancy allows for seamless optimization without hindering other modification techniques, ensuring a delicate balance between stability and precision in numerical simulations.

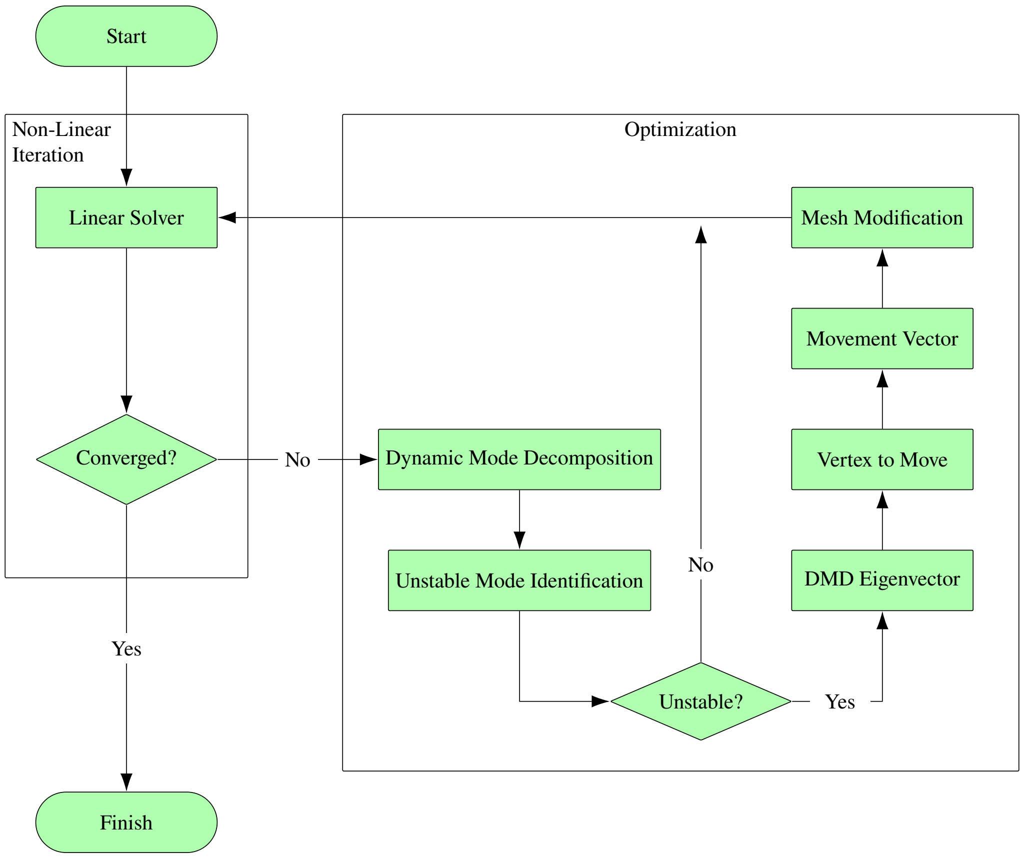


Fig. 7 Overview of the mesh optimization approach [4]

Guaranteeing an accurate numerical solution during the mesh optimization process is paramount, necessitating the preservation of the boundary geometry. Consequently, if a vertex located on the boundary is proposed for modification, the movement vector must be constrained to adhere to the boundary, a restriction that might limit the efficacy of boundary vertex adjustments within the optimization framework. This constraint holds even in highly improbable scenarios where the suggested vertex lies on the boundary, and the allowable vertex movement aligns normal to the optimal gradient. In such instances, the first recourse involves changing the movement vector direction so that the eigenmodes are pushed to the stable side of the spectrum with possibly different imaginary values. Alternatively, selecting other vertices for modification serves as a secondary solution to this challenge. The optimization method presented here prioritizes vertices that have the greatest influence on stability. Nevertheless, it is crucial to recognize that other vertices within a local selection also impact the stability of the specific eigenmode in question. Consequently, these alternate vertices can be considered for modifications, offering a comprehensive approach to improving solution stability.

E. Algorithm

An overview of the presented mesh optimization approach is shown in Figure 7. At each solution iteration, DMD is performed on a collection of the most recent solution update vectors. The number of vectors in use affects the computational complexity of the optimization application. However, according to our experiments, this number is not a function of the degrees of freedom in the solution. To show this, we have performed mesh optimization on large problems with around 21.7 million degrees of freedom using only the 10 most recent solution update vectors.

In the next step, unstable solution modes are automatically identified based on their magnitude. In the case of unstable modes, we continue the mesh optimization process by finding the DMD eigenvectors. Using the eigenvectors, certain vertices are identified on the mesh for alteration, and modification vectors are calculated. Lastly, using the previous computations we can perform mesh optimization and continue the solution or restart from the initial conditions.

IV. Ansys Fluent Integration

The presented stability improvement algorithm treats the flow solver as a black box without requiring any access to the underlying software architecture. This approach offers a significant advantage due to its non-invasive nature when integrated with external solvers. Our primary objective is to develop algorithms with seamless integration without necessitating any modifications to the internal flow solver architecture, which is achieved in the current study. Consequently, a natural course of action is to utilize the presented methodology along with an industrial flow solver to demonstrate its effectiveness and robustness. Ansys Fluent stands out as a preferred option, given its widespread utilization in both scientific research and industrial applications.

We have successfully implemented our DMD stability improvement method in conjunction with Ansys Fluent version 2023 R2, showcasing effortless integration. Notably, this process did not require access to the Fluent source code. In simpler terms, only a valid user license for Ansys Fluent software is required to replicate the results demonstrated in this work. This demonstrates the accessibility and usability of the presented method for a wide range of industrial applications without delving into complex code modifications. Note that Ansys Fluent specific nomenclature (e.g. boundary condition types and turbulence models) are used herein, which are explained in the software manual.

A. Approach

To seamlessly implement the presented stability improvement approach outlined in this study with any flow solver, the sole requisites are access to solution vectors, along with the Jacobian matrix. It can be reasonably assumed that most major flow solvers offer access to the solution during the numerical simulation. It is important to note that the Jacobian matrix does not need to be explicitly defined and only an implicit representation is sufficient for the stability improvement algorithms presented. The Ansys Fluent solver provides a matrix-free representation of the Jacobian matrix in the Adjoint solver subsystem which is utilized for the purpose of the current study.

PyFluent allows the use of Fluent software within a Python environment of the user's choice in conjunction with other Python libraries. PyFluent implements a client-server architecture and uses remote procedure calls to launch or connect with a running Fluent process as a server. However, the user is only required to interact with the Python interface. PyFluent version 0.18.2 is utilized to create, interact with, and control a Fluent session to implement the presented stability improvement methodology. To extract the required information from Fluent, we have developed a collection of UDFs that are called during the simulation from the Python interface. These UDFs provide access to the solution vector, the Jacobian matrix, and the ability to perform calculated modifications to the mesh. As we will show in the next section, during this project, we used Ansys Fluent to

- Successfully apply our novel stability improvement algorithm to an industrial solver
- Perform mesh optimization in large-scale 3D turbulent problems
- Utilize the turbulent variables for mesh optimization
- Demonstrate the effectiveness of the method in diverse scenarios

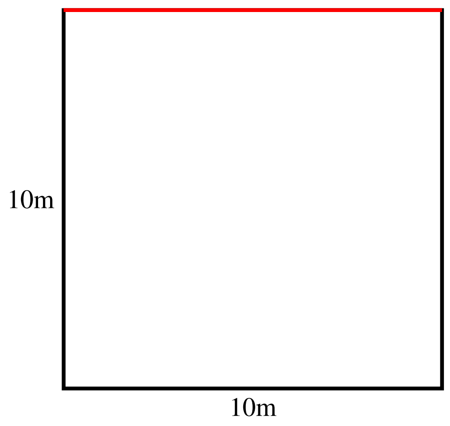
For this purposes, no access to the underlying software architecture is needed and the computational cost of the approach is negligible compared to each iteration of the flow solver.

V. Results

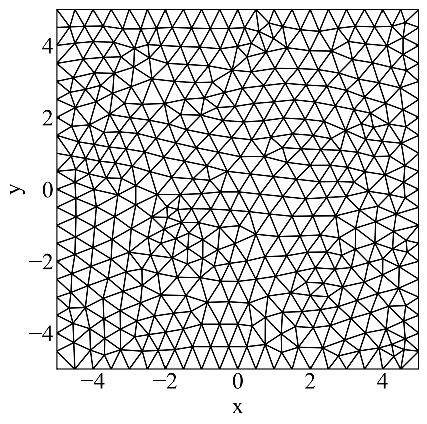
We have demonstrated the efficiency and effectiveness of the algorithm by implementing mesh optimization on diverse test cases in Ansys Fluent, highlighting the improvements in residual history. Furthermore, this end was achieved without access to the source code, preserving the software's original architecture. The results underscore the strength and viability of this algorithm when integrated with third-party flow solvers. Additionally, our study showcases its effectiveness in 3D turbulent simulations.

A. 2D Cavity

The lid-driven cavity is chosen as the first test case. Figure 8a shows the 10m $\times$ 10m solution domain of this simulation in which the black lines are set to be stationary no-slip walls and the red line is set to be a moving no-slip wall. Figure 8b depicts a typical triangular mesh that was used for the solution. The cavity lid is moving at a constant velocity from left to right. Fluid density and viscosity are both constant, set to $1.225\,\frac{\mathrm{kg}}{\mathrm{m}^3}$ and $1.7894 \times 10^{-5}\,\frac{\mathrm{kg}}{\mathrm{m}\cdot\mathrm{s}}$, respectively.



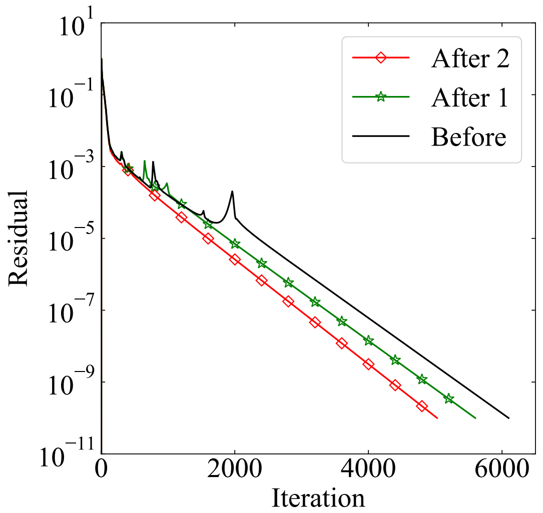
(a) The physical domain



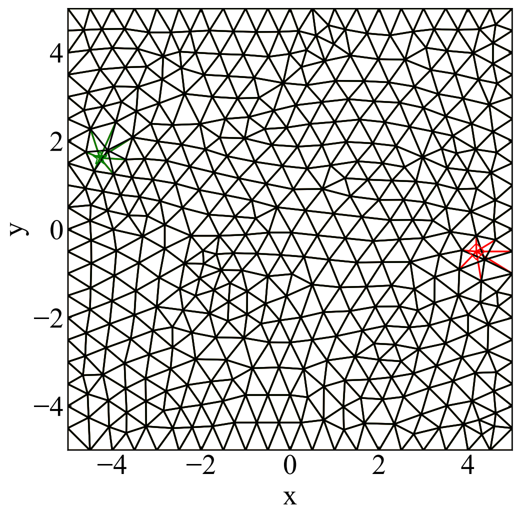
(b) An example of the mesh

Fig. 8 Solution domain and mesh for a lid-driven cavity problem

In the first experiment, the top wall moves at a velocity of $1\frac{\mathrm{m}}{\mathrm{s}}$ to the right, giving a Reynolds number of 685,000. The viscosity is set to laminar which is not a correct physical simulation given the Reynolds number. In this case, the mesh includes 814 triangular cells, 1261 faces, and 448 nodes generated with the Ansys Meshing software. As seen in Figure 9a presented in black, the original solution is converging. DMD is performed on the latest 10 solution update vectors to identify solution modes. In the first optimization iteration, the second dominant DMD mode is used to select a single vertex for modification at iteration 200 of the solver. We should note that, in this case, the dominant mode at iteration 200 has a physical behavior that makes it infeasible for vertex selection and optimization. This optimization iteration results in the modification of the single green vertex in Figure 9b with the improved residual history depicted in Figure 9a. In the next optimization iteration, the second dominant DMD mode is used for vertex selection at iteration 360 of the new solution. This process results in the modification of the red vertex presented in Figure 9b with the improved residual history depicted in Figure 9a.



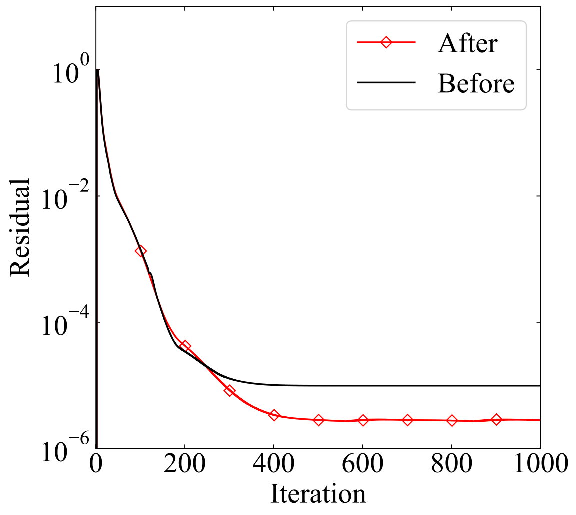
(a) The residual history



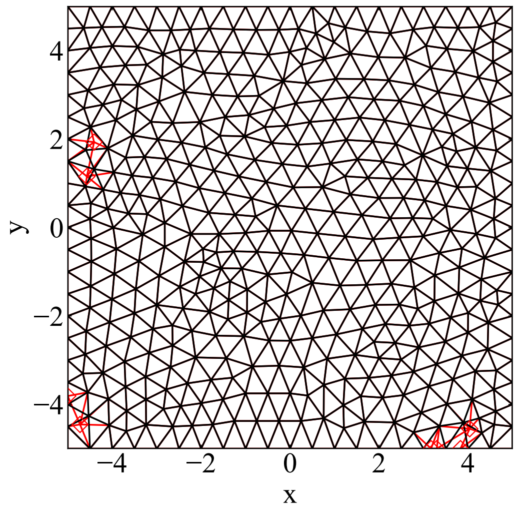
(b) Mesh modification

Fig. 9 DMD stability improvement in a 2D cavity problem

This problem is solved once more using the $k-\omega$ SST turbulence model. The residual history for the original solution is depicted in black in Figure 10a. DMD is performed on the latest 10 solution update vectors to identify solution modes. At iteration 400, the dominant DMD mode is used for mesh optimization to modify the vertices presented in Figure 10b. The final residual in this case undergoes the improvement depicted in Figure 10a. In this test case, the DMD vector is used to identify multiple vertices on the mesh for modification. For this purpose, all the entries of the vector with a magnitude larger than a threshold value point to the vertices in question. This approach increases the effectiveness of the stability improvement approach in some scenarios.



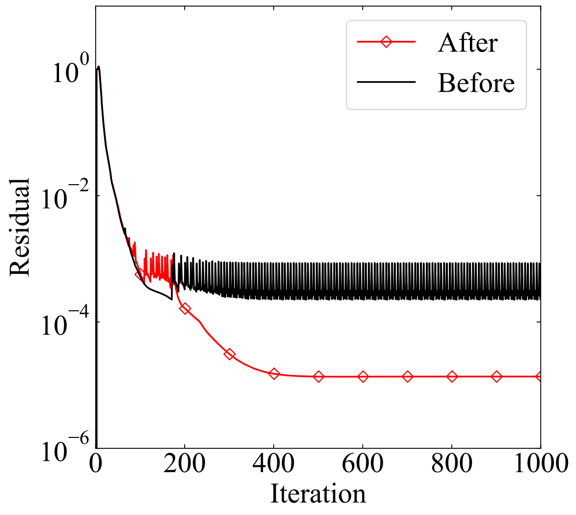
(a) The residual history



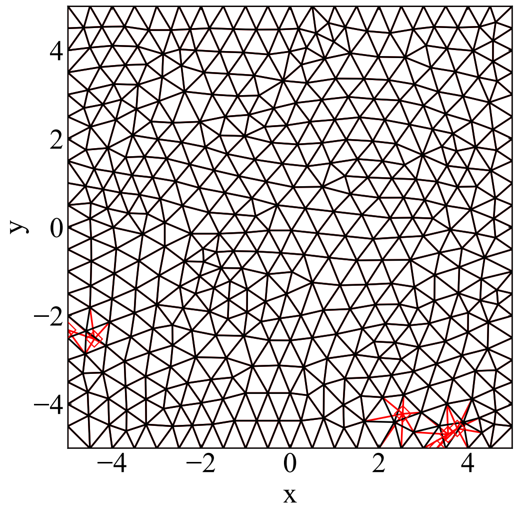
(b) Mesh modification

Fig. 10 DMD stability improvement in a 2D cavity problem

The same simulation is performed with the velocity of the top wall set to $0.1\frac{\mathrm{m}}{\mathrm{s}}$ with the $k-\omega$ SST turbulence model. In this case, the Reynolds number is 68,500. The residual history for the original solution is depicted in black in Figure 11a. DMD is performed on the latest 10 solution update vectors to identify solution modes. At iteration 100, the second dominant DMD mode is used for mesh optimization to modify the vertices presented in Figure 11b. The final residual in this case undergoes the improvements shown in Figure 11a.



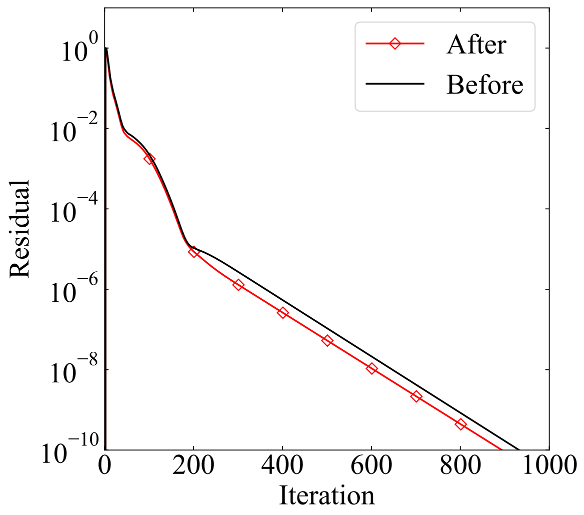
(a) The residual history



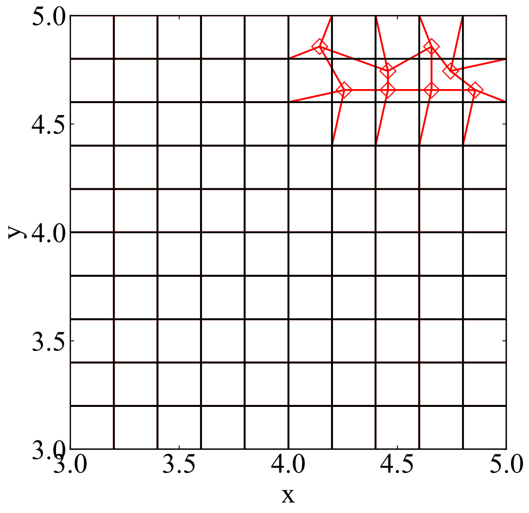
(b) Mesh modification

Fig. 11 DMD stability improvement in a 2D cavity problem

A structured mesh is generated to solve this problem with 2500 cells, 5100 faces, and 2601 nodes. The top wall velocity is set to $1.0\frac{\mathrm{m}}{\mathrm{s}}$ for a Reynolds number of 685,000. Once again, the $k-\omega$ SST turbulence model is used in the simulation. The residual history for the original solution is depicted in black in Figure 12a. DMD is performed on the latest 10 solution update vectors to identify solution modes. At iteration 300, the second dominant DMD mode is used for mesh optimization to modify the vertices presented in Figure 12b. The final residual in this case undergoes the improvements depicted in Figure 12a.



(a) The residual history



(b) Mesh modification

Fig. 12 DMD stability improvement in a 2D cavity problem

B. 3D Cavity

The 3D lid-driven cavity is chosen as the next test case. The domain is a simple 10m $\times$ 10m $\times$ 10m cube with the top wall moving at a constant velocity. The boundary conditions are set to no-slip moving wall for the cavity lid and no-slip stationary wall for the 5 remaining sides of the cube. The mesh consists of 39812 tetrahedral cells, 82105 faces, and 7951 nodes. Fluid density and viscosity are both constant, set to $1.225\frac{\text{kg}}{\text{m}^3}$ and $1.7894 \times 10^{-5}\frac{\text{kg}}{\text{m}\cdot\text{s}}$, respectively.

The moving wall velocity is set to $0.001\frac{\text{m}}{\text{s}}$ for a Reynolds number of 685. The $k - \omega$ SST turbulence model is used for this simulation. DMD is performed on the latest 10 solution update vectors to find the solution modes. Mesh optimization is performed at iteration 100 of the solver on the dominant DMD mode. During mesh optimization, in this case, the locations of 6 vertices on the mesh are modified. Figure 13 shows the residual history before and after the application. Previous cases showed only modest improvements in convergence, but this case shows a qualitative improvement, eliminating a source of stalled convergence and enabling convergence to machine zero.

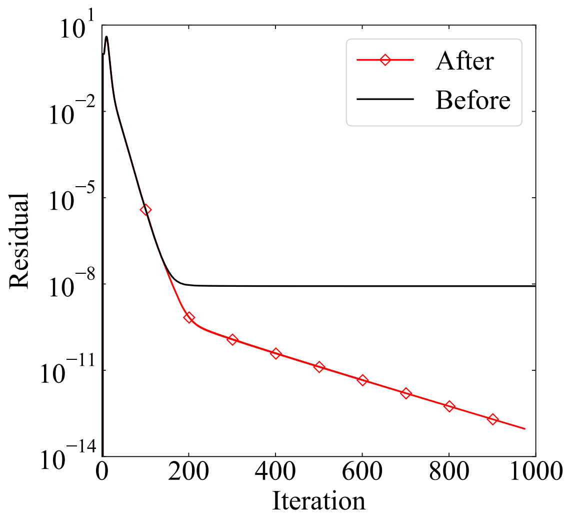


Fig. 13 DMD stability improvement in a 3D cavity problem

Next, the lid velocity is increased to $0.01\frac{\mathrm{m}}{\mathrm{s}}$ for a Reynolds number of 6,850. The $k - \omega$ SST turbulence model is used for this simulation. DMD is performed on the latest 10 solution update vectors to find the solution modes. The first mesh optimization iteration is performed at iteration 100 of the solver on the dominant DMD mode to modify the location of 19 vertices. The oscillations in the residual history undergo a frequency reduction and amplitude increment depicted in Figure 14 in orange. The next mesh optimization iteration is performed on the new solution at iteration 100 of the solver to modify the locations of 25 vertices on the mesh. The residual history after this optimization iteration is presented in Figure 14 in green. The final optimization iteration is performed at iteration 100 of the new solution to modify the locations of 15 vertices. As seen in Figure 14 in red, the final residual is substantially improved over the initial solution.

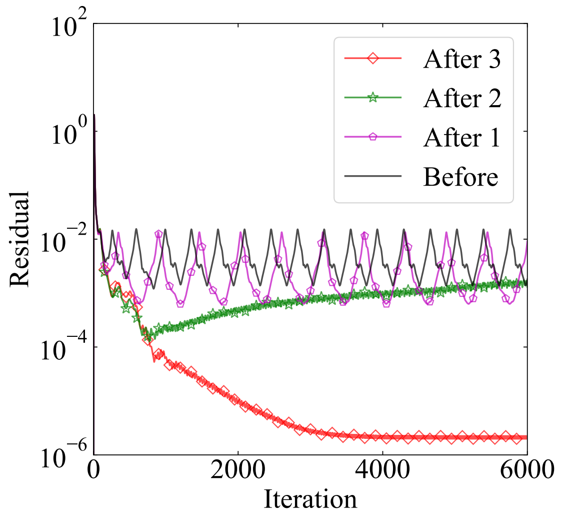
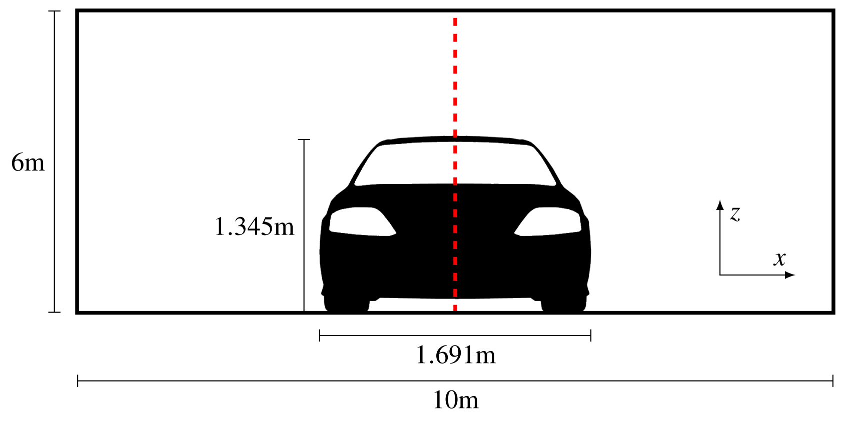


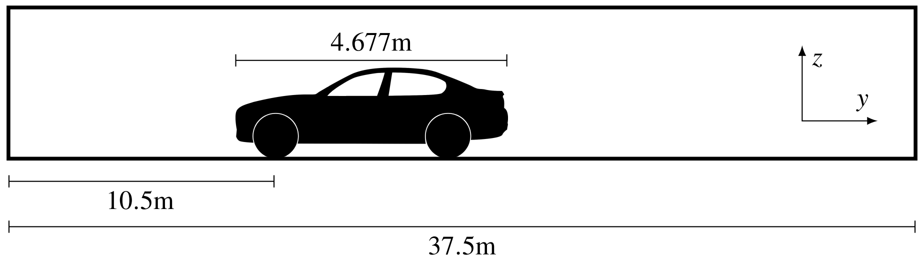
Fig. 14 DMD stability improvement in a 3D cavity problem

C. Large-Scale 3D Turbulent Automotive

An incompressible 3D turbulent flow around a sedan car is selected as the next test case in Ansys Fluent. Figure 15 shows the front and side views of the solution domain with the dashed red line used as a symmetry plane in the simulation. The car body, wheels, and the floor plane are set to no slip walls. The velocity inlet boundary condition is set to $27.8\frac{\mathrm{m}}{\mathrm{s}}$ in the $+y$ direction and the opposing boundary is set to pressure outlet. The remaining planes are all set to symmetry boundary condition. The simulation Reynolds number is 8,700,000 and the $k-\omega$ SST turbulence model is used. Figure 16 shows the mixed mesh on the symmetry plane that is used for this simulation. The mesh includes 3,618,080 cells, 8,252,872 faces, and 1,266,920 nodes. The highest aspect ratio for the cells in this mesh is 172. This large-scale problem includes 21,708,480 degrees of freedom and clearly depicts the efficacy and scalability of my presented methodology to substantial CFD simulations and anisotropic meshes. Fluid density and viscosity are both constant, set to $1.225\,\frac{\text{kg}}{\text{m}^3}$ and $1.83\times10^{-5}\,\frac{\text{kg}}{\text{m}\cdot\text{s}}$, respectively. DMD is performed on only the latest 10 solution update vectors to identify the dominant solution modes which is remarkable considering the problem size of 21.7 million degrees of freedom. One optimization iteration is performed in this case which results in the residual history improvement presented in Figure 17a. Figure 17b depicts the history of drag force on the body and the wheels of this vehicle. As seen, the result is slightly different in the new solution with smaller oscillations. Figure 18 shows the contours of skin friction coefficient before and after mesh optimization in this test case. As depicted, there is a similar pattern between the two solutions with little noticeable differences. The minimum skin friction coefficient after mesh optimization is slightly lower than before.

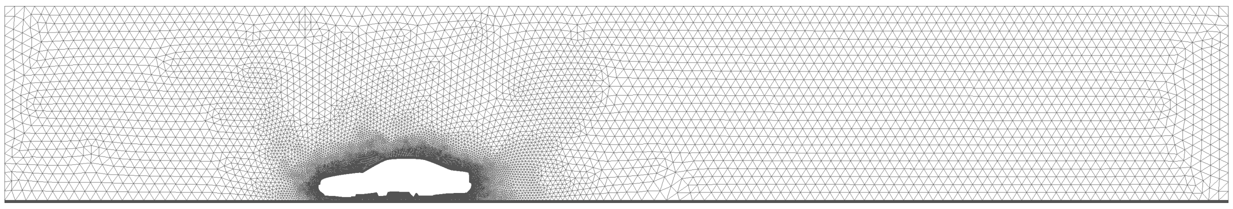


(a) Front view

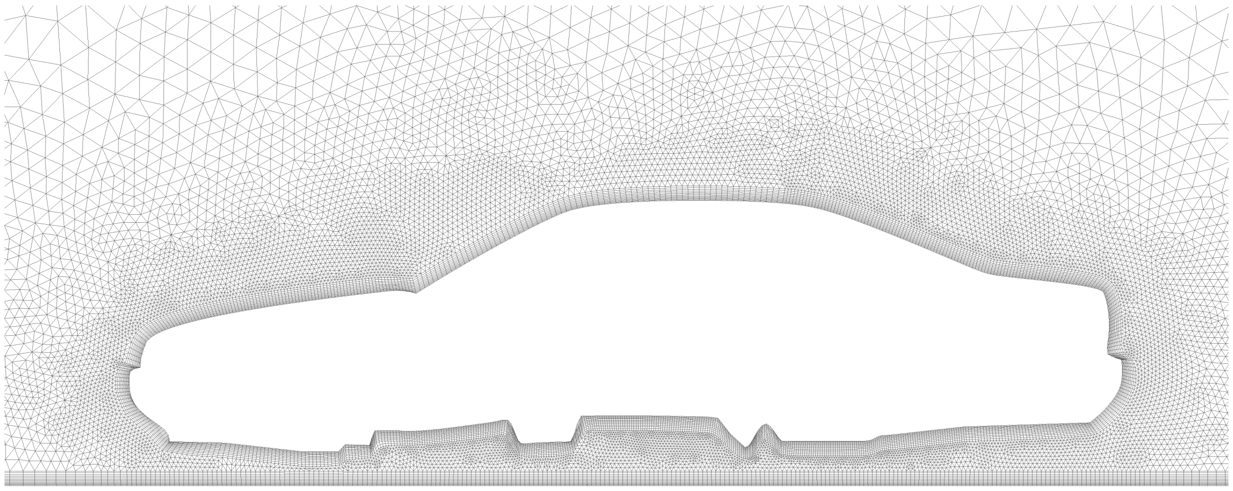


(b) Side view

Fig. 15 Solution domain for a large-scale turbulent automotive problem

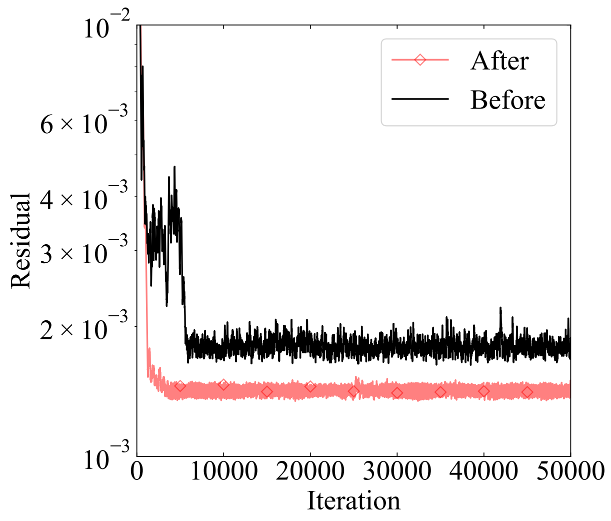


(a) Full view

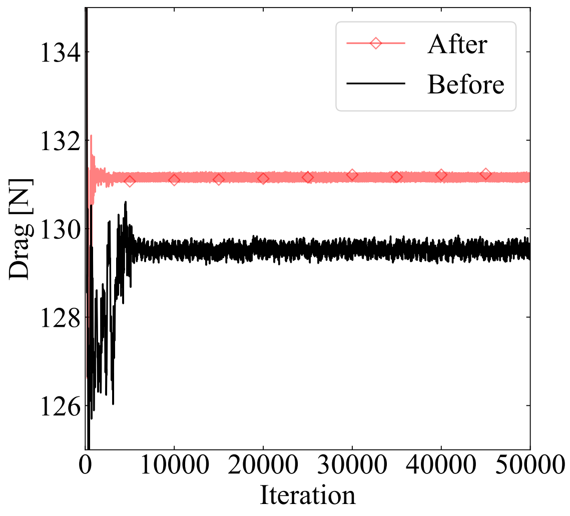


(b) Zoom view

Fig. 16 Symmetry plane mesh for a large-scale turbulent automotive problem

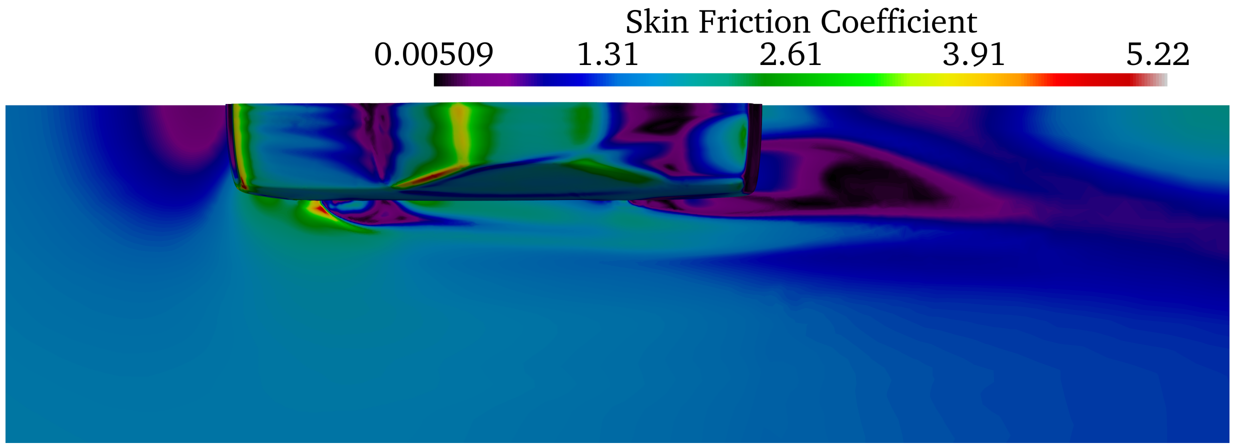


(a) Residual history

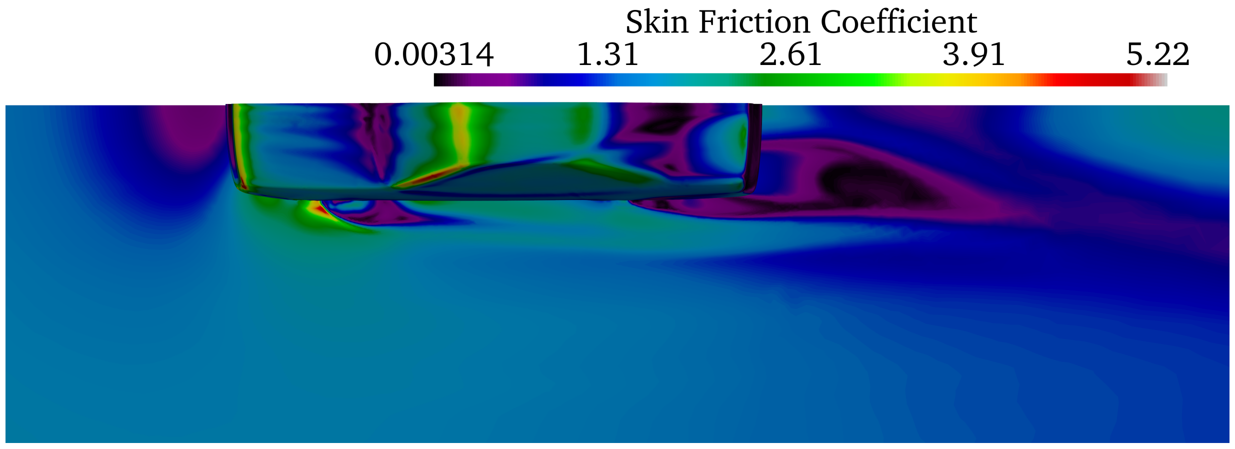


(b) Drag history

Fig. 17 Ansys Fluent mesh optimization in a large-scale turbulent automotive problem



(a) Before optimization



(b) After optimization

Fig. 18 Skin friction coefficient during mesh optimization in a large-scale turbulent automotive problem

D. NASA Common Research Model

The NASA Common Research Model (CRM) is an open-source and publicly available low-wing transonic commercial transport aircraft geometry that is developed for CFD validation and to encourage collaboration among global research teams. This model was developed with high priority for CFD validation studies and minimal flow separation. The geometry can be configured with various degrees of complexity including different combinations of wing, body, tail, pylon, and nacelle arrangements. Further, the CFD studies performed on this model are made publicly available which include the geometry, mesh, CFD simulation results, as well as the wind tunnel test data. An isometric view of the CRM wing-body geometry is shown in Figure 19. The transonic supercritical wing ensures well-behaved aerodynamic characteristics, with or without nacelle and pylon attachments. The horizontal tail is designed for standard stability and control requirements. The fuselage is representative of a wide-body commercial transport aircraft, featuring the wing-body fairing. The single-cowl nacelle is engineered with a high bypass ratio, and its exit area is sized to attain a natural, unforced mass flow ratio similar to cruise conditions in commercial aircraft engines. The reference quantities of the full-scale CRM model are presented in Table 1. The details of this geometry can be found in the work of Vassberg et al. [12].

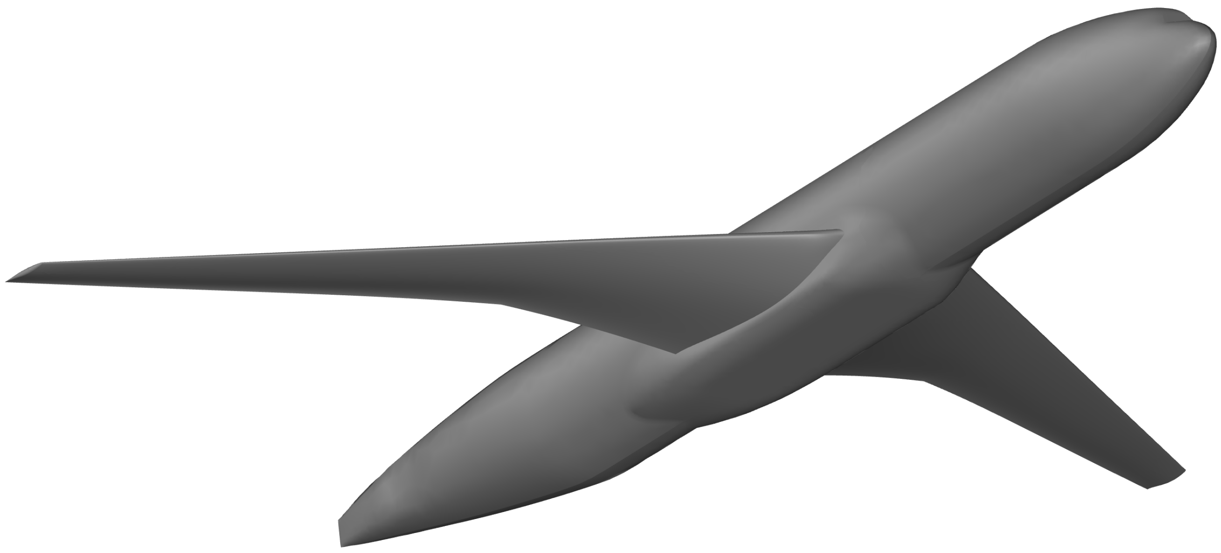


Fig. 19 CRM wing-body configuration isometric view

Table 1 CRM full-scale reference quantities

Quantity	Value
Full Span Reference Area	383.6896 m^2
Reference Chord	7.0053 m
Wing Span	58.7629 m
Moment Reference Center in x	33.6779 m
Moment Reference Center in y	0.0 m
Moment Reference Center in z	4.5199 m
Aspect Ratio	9.0
Taper Ratio	0.275
Quarter-Chord Sweep	35°

The High Lift Common Research Model (CRM-HL) is a geometry set developed based on the original CRM model to enable high lift configurations representative of those found on modern commercial airliners. The primary objectives of CRM-HL are to present flow physics of interest for the intended use, to provide reference performance metrics for a conventional high-lift system, and to be easy to replicate by a wide user base. This model introduces complex high-lift devices, brackets, fairings, and sealing between elements to the CRM model. An isometric view of the CRM-HL wing-body-pylon-nacelle geometry along with the high-lift devices is shown in Figure 20. The details of this geometry and configurations are given in the work of Lacy and Clark [13].

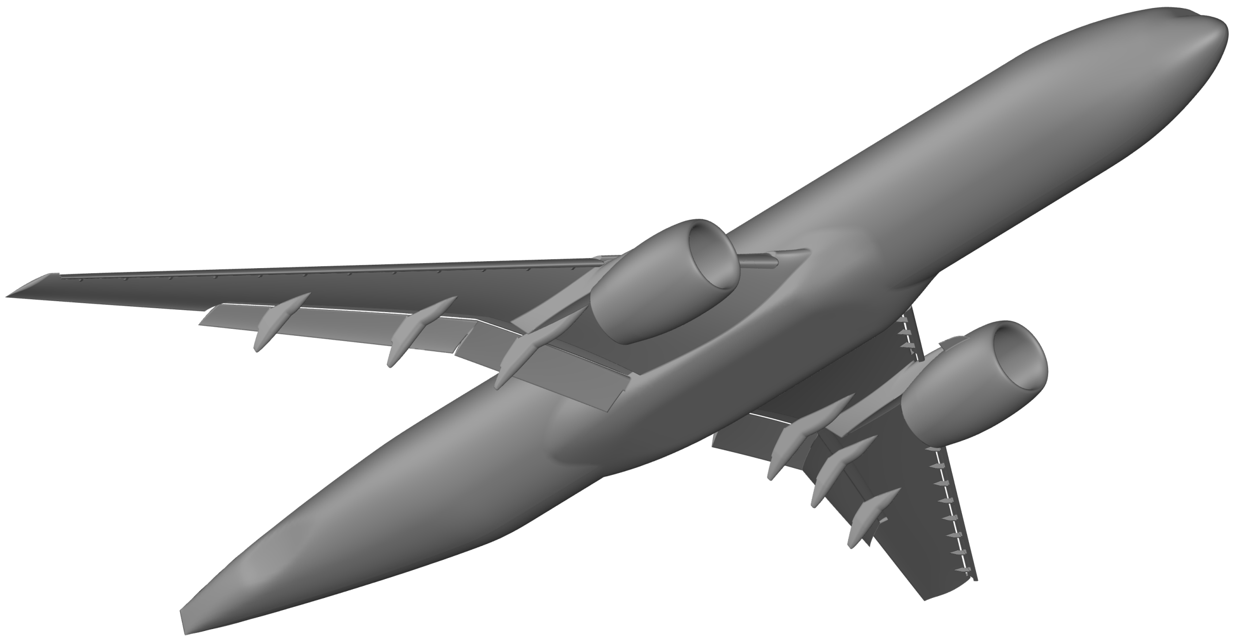


Fig. 20 CRM-HL wing-body-pylon-nacelle configuration along with high-lift devices isometric view

To conduct tests on numerical stability improvement in the context of my work, I use meshes and test cases provided by the fifth AIAA CFD High Lift Prediction Workshop. Three general test cases are presented including Case 1 for verification, Case 2 for configuration build-up, and Case 3 for the study of Reynolds number effects. This workshop utilizes different configurations of the CRM-HL model [13]. Case 1 is a wing-body configuration geometry with no high-lift devices attached. This geometry includes a cuboid computational domain with dimensions that extent $[-1651, 1651]$ meters in x, $[0, 1651]$ meters in y, and $[-1651, 1651]$ meters in z. Symmetry is specified at the $y = 0$ plane and far-field boundary conditions on Riemann invariants are assigned at all other far-field boundaries of the domain. The flow is considered compressible at Mach 0.2 and 11° angle of attack. The reference static temperature is set to 289.444 Kelvin and the fluid is assumed to be an ideal gas with $\gamma = 1.4$. The walls are set to no-slip solid boundaries with zero heat flux.

As a first test case, the Ansys Fluent solver is set up with the Case 1 wing-body geometry. Grid 1v from the work of Diskin et al. [14], with 2,582,295 cells, 5,975,424 faces, and 956,174 nodes is selected for this test case. The simulation Reynolds number is 32.7 million and the $k - \omega$ SST turbulence model is used for a fully turbulent RANS simulation. The highest aspect ratio for the cells in this mesh is 4.4734×10^4 and the minimum orthogonal quality metric is 2.1024×10^{-6}. This problem includes 18.1 million degrees of freedom.

DMD is performed on only the latest 10 solution update vectors to identify the dominant solution modes in this large-scale problem. A single optimization iteration is performed in this case and the locations of three vertices are modified located near the tip of the wing on the top surface indicated in Figure 21 with white. This optimization iteration results in half an order of magnitude reduction in the residual history presented in Figure 22a. Figures 22b and 22c depict the history of lift and drag coefficients on this wing-body configuration. As seen, the results are slightly different in the new solution with around 0.1% difference for the lift coefficient and around 0.3% difference for the drag coefficient. The skin friction coefficient contours on the top surface of the wing before and after mesh optimization are presented in Figure 23. As depicted, the skin friction coefficient patterns are similar in both solutions with little noticeable difference.

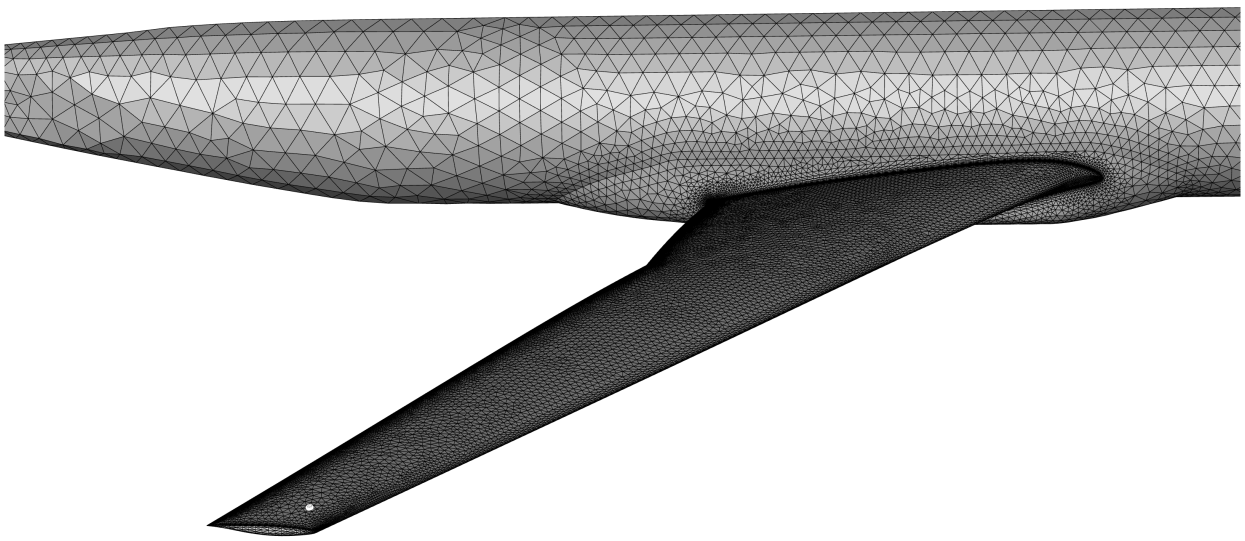
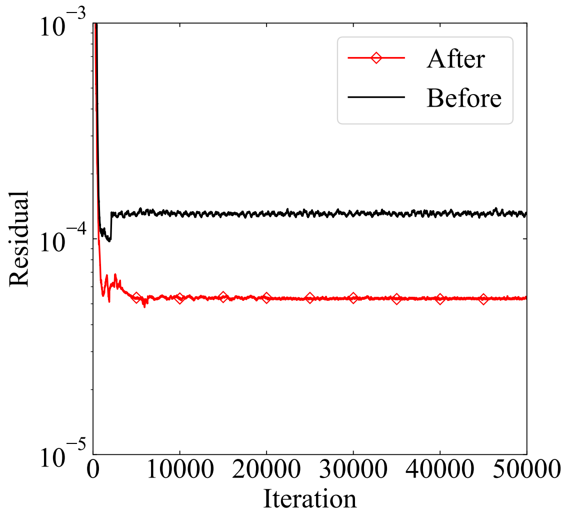
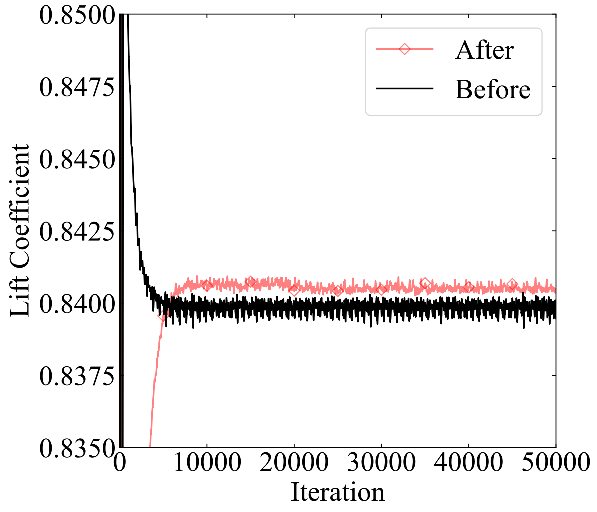


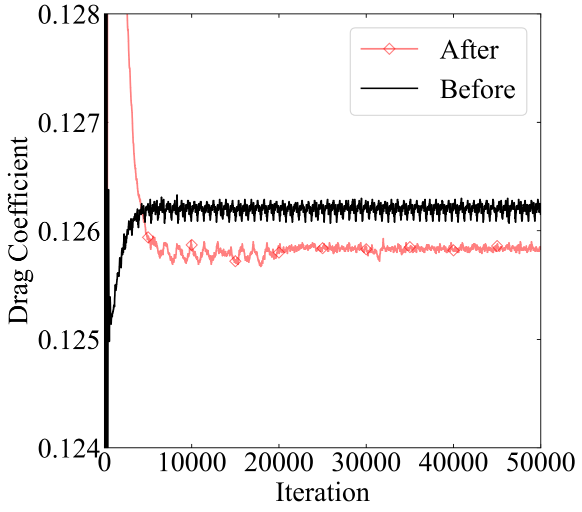
Fig. 21 The modified vertices during mesh optimization in CRM-HL wing-body configuration



(a) Residual history

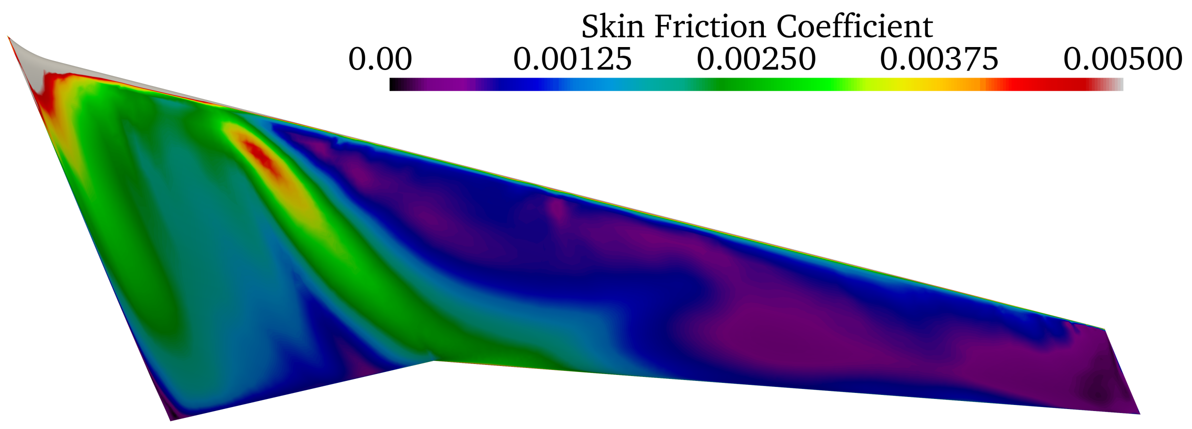


(b) Lift coefficient history

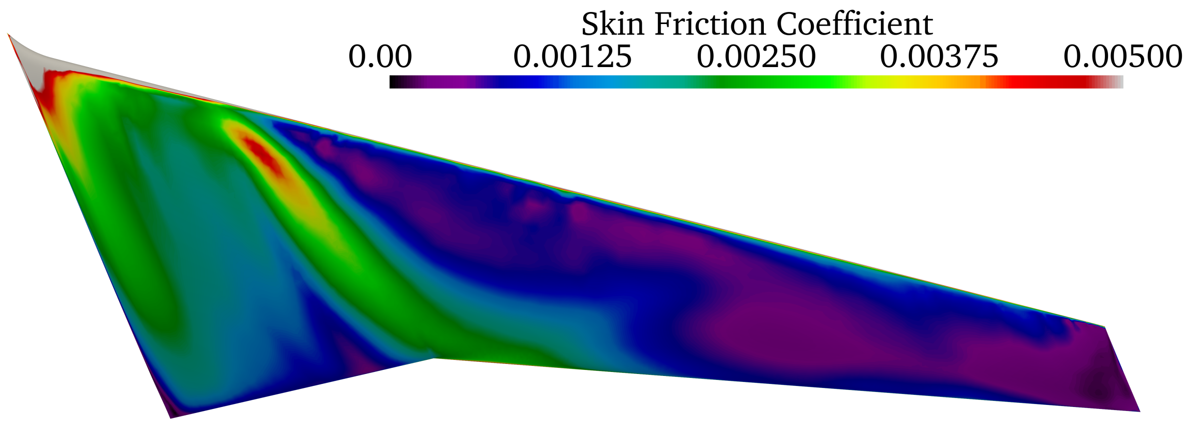


(c) Drag coefficient history

Fig. 22 Ansys Fluent mesh optimization in CRM-HL wing-body configuration



(a) Before optimization



(b) After optimization

Fig. 23 Skin friction coefficient during mesh optimization in CRM-HL wing-body configuration

The same test case is set up with a Reynolds number of 5,600,000 in the Ansys Fluent solver. The $k-\omega$ SST turbulence model is used for a fully turbulent RANS simulation in this case as well. DMD is performed on only the latest 10 solution update vectors to identify the dominant solution modes in this large-scale problem. A single optimization iteration is performed in this case and the locations of six vertices are modified on the top surface of the wing indicated in Figure 24 with white. This optimization iteration results in half an order of magnitude reduction in the residual history presented in Figure 25a. Figures 25b and 25c depict the history of lift and drag coefficients on this wing-body configuration. As seen, the results are slightly different in the new solution with around 0.1% difference for the lift coefficient and around 0.5% difference for the drag coefficient. The skin friction coefficient contours on the top surface of the wing before and after mesh optimization are presented in Figure 26. As depicted, the skin friction coefficient patterns are similar in both solutions with little noticeable difference.

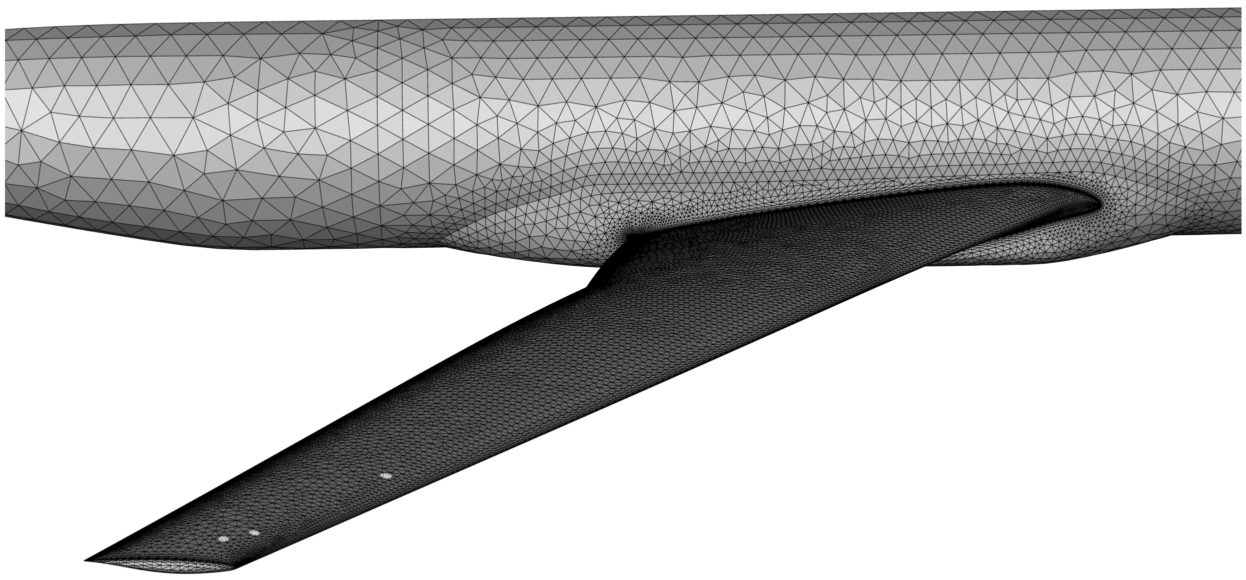
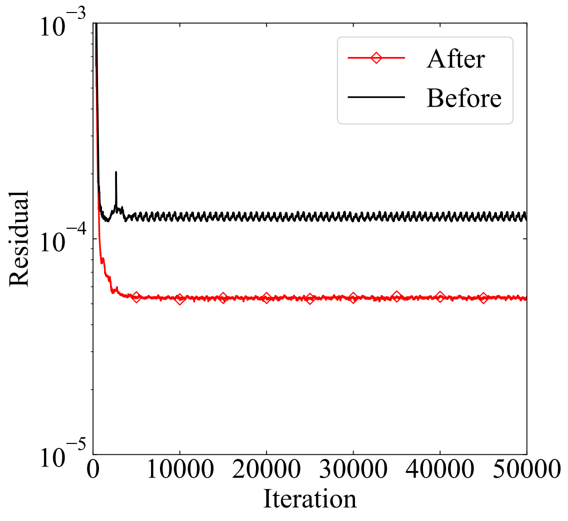
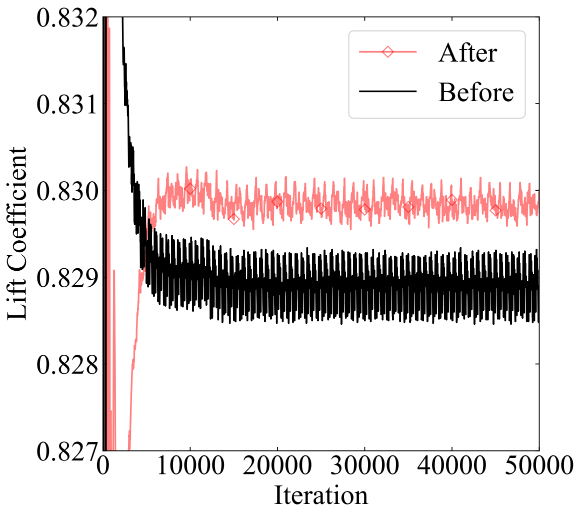


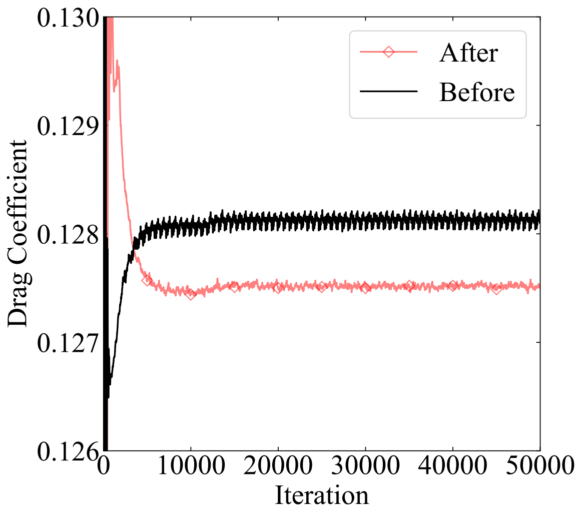
Fig. 24   The modified vertices during mesh optimization in CRM-HL wing-body configuration



(a) Residual history

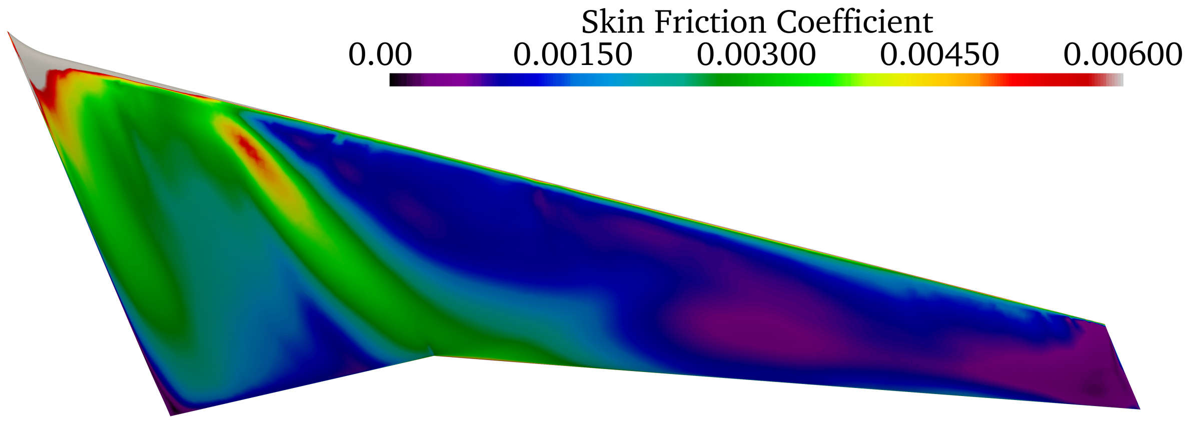


(b) Lift coefficient history

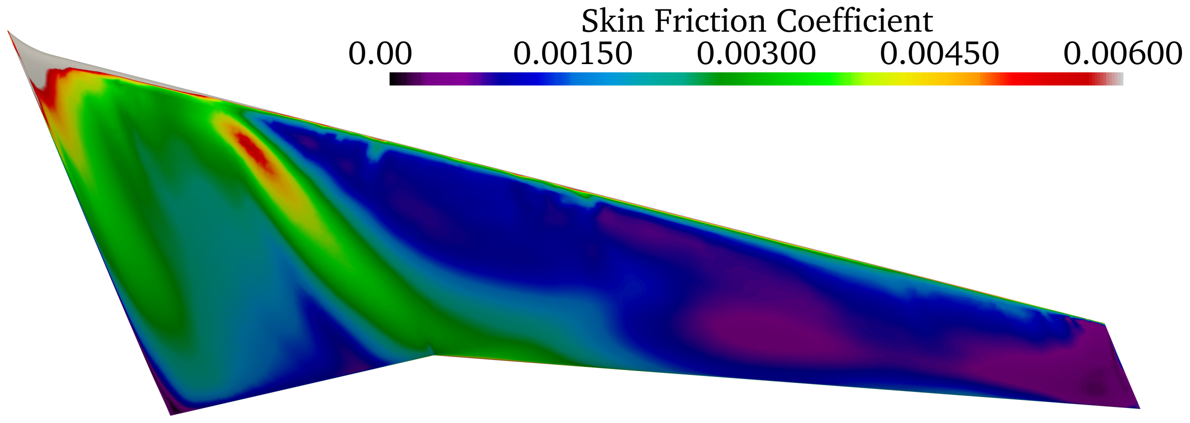


(c) Drag coefficient history

Fig. 25 Ansys Fluent mesh optimization in CRM-HL wing-body configuration



(a) Before optimization



(b) After optimization

Fig. 26 Skin friction coefficient during mesh optimization in CRM-HL wing-body configuration

VI. Conclusion

This study employs our innovative mesh optimization approach to enhance the numerical stability and convergence rate of finite-volume simulations, specifically in conjunction with the Ansys Fluent flow solver. Our approach leverages the computationally efficient dynamic mode decomposition of solution update vectors to identify problematic solution modes during the simulation. The seamless integration with Fluent requires only access to solution vectors and partial access to the Jacobian matrix. To facilitate this, custom user-defined functions were developed, providing the necessary information at each iteration of the solver without requiring access to the underlying software architecture. The results demonstrate substantial improvements in the numerical stability and convergence behavior of the test cases within Ansys Fluent. Notably, this work pioneers the application of the presented algorithm to fully turbulent simulations, showcasing its robustness and effectiveness. Furthermore, successful mesh optimization was achieved in a large-scale simulation featuring over 21.7 million degrees of freedom, accomplishing this at a negligible fraction of the computational cost incurred by each iteration of the flow solver.

Acknowledgement

This work is co-sponsored by Ansys, Inc. and the Natural Sciences and Engineering Research Council of Canada under the Cooperative Research and Development Grant F19-00204. Parts of the findings presented in this work were realized during an internship at Ansys, Inc. by the first author. We express our sincere gratitude for the mentorship provided by Chris Hill and other members of the Ansys Fluent team, whose constructive input greatly enriched the quality and depth of this study. Parts of the findings in this work were enabled through support provided by the Digital Research Alliance of Canada.

References

[1] Zandsalimy, M., and Ollivier-Gooch, C., "A Novel Approach to Mesh Optimization to Stabilize Unstructured Finite Volume Simulations," *Journal of Computational Physics*, Vol. 453, 2022, p. 110959. https://doi.org/10.1016/j.jcp.2022.110959.

[2] Zandsalimy, M., and Ollivier-Gooch, C., *Unsupervised Residual Vector Analysis for Mesh Optimization*, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2023, p. 0833. https://doi.org/10.2514/6.2023-0833.

[3] Zandsalimy, M., and Ollivier-Gooch, C., "Residual Vector and Solution Mode Analysis Using Semi-Supervised Machine Learning for Mesh Modification and CFD Stability Improvement," *Journal of Computational Physics*, Vol. 510, 2024, p. 113063. https://doi.org/10.1016/j.jcp.2024.113063.

[4] Zandsalimy, M., and Ollivier-Gooch, C., *Dynamic Mode Decomposition for Improved Numerical Stability of Finite Volume Simulations*, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2024, p. 1948. https://doi.org/10.2514/6.2024-1948.

[5] Ollivier-Gooch, C., "A Toolkit for Numerical Simulation of PDEs: I. Fundamentals of Generic Finite-Volume Simulation," *Computer Methods in Applied Mechanics and Engineering*, Vol. 192, No. 9, 2003, pp. 1147–1175. https://doi.org/10.1016/S0045-7825(02)00602-3.

[6] Zandsalimy, M., and Ollivier-Gooch, C., *Approximate Jacobian Eigenanalysis for Unstructured Mesh Optimization of Finite Volume Simulations*, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2024, p. 1949. https://doi.org/10.2514/6.2024-1949.

[7] Zandsalimy, M., and Ollivier-Gooch, C., "CFD Stability Improvement Using Dynamic Mode Decomposition of Solution Update Vectors," *Under Review, Journal of Computational Physics*, ????

[8] Schmid, P. J., "Dynamic Mode Decomposition of Numerical and Experimental Data," *Journal of Fluid Mechanics*, Vol. 656, 2010, pp. 5–28. https://doi.org/10.1017/S0022112010001217.

[9] Stanković, L., "On the Sparsity Bound for The Existence of a Unique Solution in Compressive Sensing by The Gershgorin Theorem," *Signal Processing*, Vol. 190, 2022, p. 108316. https://doi.org/10.1016/j.sigpro.2021.108316.

[10] Berger, M. J., and Colella, P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *Journal of Computational Physics*, Vol. 82, No. 1, 1989, pp. 64–84.

[11] MacNeice, P., Olson, K. M., Mobarry, C., De Fainchtein, R., and Packer, C., "PARAMESH: A Parallel Adaptive Mesh Refinement Community Toolkit," *Computer Physics Communications*, Vol. 126, No. 3, 2000, pp. 330–354.

[12] Vassberg, J., Dehaan, M., Rivers, M., and Wahls, R., *Development of a Common Research Model for Applied CFD Validation Studies*, Guidance, Navigation, and Control and Co-located Conferences, American Institute of Aeronautics and Astronautics, 2008. https://doi.org/10.2514/6.2008-6919.

[13] Lacy, D. S., and Clark, A. M., *Definition of Initial Landing and Takeoff Reference Configurations for the High Lift Common Research Model (CRM-HL)*, AIAA AVIATION Forum, American Institute of Aeronautics and Astronautics, 2020. https://doi.org/10.2514/6.2020-2771.

[14] Diskin, B., Liu, Y., and Galbraith, M. C., *High-Fidelity CFD Verification Workshop 2024: Spalart-Allmaras QCR2000-R Turbulence Model*, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2023. https://doi.org/10.2514/6.2023-1244.