# CFD Stability Improvement Using Dynamic Mode Decomposition of Solution Update Vectors

Mohammad Zandsalimy[a,*], Carl Ollivier-Gooch[a]

[a] *Department of Mechanical Engineering, The University of British Columbia, 2054-6250 Applied Science Lane, Vancouver, BC Canada V6T 1Z4*

## Abstract

We present a novel method for improving the numerical stability of finite volume simulations by optimizing the mesh through dynamic mode decomposition of solution update vectors. Our approach leverages dynamic mode decomposition to approximate the non-linear solution evolution as a linear mapping, enabling us to represent the dynamic system with fewer degrees of freedom. By conducting eigenanalysis on the reduced system, we gain insights into the growth rate and oscillation frequency of dominant solution modes. We then identify the control volumes and vertices that have a significant influence on each dynamic solution mode. We compute the gradients of the Jacobian diagonal elements with respect to the movement of the selected vertices. Based on these gradients, we adjust the positions of the vertices to improve the diagonal dominance of the corresponding Jacobian rows. We utilize this approach in conjunction with our in-house flow solver as well as with Ansys Fluent to test its effectiveness when applied to different types of CFD software architecture. We show that the presented methodology is fully non-invasive to the host flow solver and does not require any modifications or access to the source code. Our approach offers a substantial computational cost reduction compared to existing methods for numerical stability improvement. Various results demonstrate the strength and efficacy of this state-of-the-art approach for improving numerical stability

[*] Corresponding Author
*Email address:* mohammad.zandsalimy@ubc.ca (Mohammad Zandsalimy)

through mesh optimization.

Edition: 3.0 on June 6, 2024

*Keywords:* Mesh Optimization, Stability Improvement, Dynamic Mode Decomposition, Finite Volume Method, Solution Mode Analysis, Ansys Fluent.

## 1. Introduction

In recent years, attention has been given to mesh optimization as a means to achieve improved numerical stability in Computational Fluid Dynamics (CFD) applications. One particular area of focus has been addressing the linear stability of the dynamic system, as highlighted by Zandsalimy and Ollivier-Gooch [1]. Building upon this line of research, we propose a pioneering approach for mesh optimization that utilizes the computationally cheap Dynamic Mode Decomposition (DMD) technique. By employing DMD on solution update vectors, we devise a novel method to enhance finite-volume simulation stability, offering promising avenues for further advancements in this field.

The main purpose of a “high quality” mesh is to facilitate a numerical solution that yields accurate and reliable outputs. Previous investigations of truncation error have shown no strong connection with a priori mesh quality metrics based solely on geometric characteristics [2]. Despite this, mesh generation methods that optimize such metrics are used frequently in automated mesh generation, and are quite successful for simulations with nearly isotropic solutions that exhibit slow variations. It is important to define more sophisticated mesh quality measures and optimization techniques for problems in which accuracy depends mostly on solution characteristics.

Habashi et al. [3] took an early step towards mesh and solver independent CFD solutions. Their method utilized a directional error estimator coupled with a mesh adaptation technique to obtain a controllable solution error level. They also showed that the order of accuracy of the numerical scheme has a reduced impact on solution accuracy for a well adapted mesh. Jiao [4] introduced a method of local volume conservation for meshes under smoothing or other types of mesh adaptation. This is critical in mesh optimization due to challenges posed during mesh modification such as preservation of sharp features and volume conservation. Scherer et al. [5] developed a numerical method for energy-based mesh optimization in CFD. They treated mesh optimization as a nonlinear minimization problem with equality and inequality constraints. Clark et al. [6] presented a mesh optimization strategy to produce “high quality” triangular meshes preserving the geometry for a high-order numerical solution. They showed that low-order remeshing can reduce the convergence accuracy or even cause numerical instability.

Zandsalimy and Ollivier-Gooch [1] focused on finite volume mesh optimization for improved numerical stability and convergence behavior. This method relies on the computationally expensive eigenanalysis of large sparse linear systems. Additionally, the optimization process necessitated frequent human intervention for critical decision-making in the process. To address these issues, Zandsalimy and Ollivier-Gooch [7] proposed a novel method for mesh modification through unsupervised anomaly detection in the residual vector. By exclusively analyzing the residual vector, they successfully identified unstable solution modes, eliminating the need for computationally expensive eigenanalysis. Synthetic vectors resembling the unstable eigenvectors were constructed from the residual and employed for mesh optimization. This novel approach automated the process entirely, eliminating the requirement for human decision-making.

In a separate contribution, Zandsalimy and Ollivier-Gooch [8] applied an improved outlier detection model to the solution modes resulting from Principal Component Analysis (PCA) of solution update vectors for unstable mode identification. This innovative approach yielded a fully automated method for dynamically modifying meshes during the numerical solution of finite volume problems. Once again, the researchers observed notable enhancements in numerical stability and convergence behavior. Building on their previous work, Zandsalimy and Ollivier-Gooch [9] presented a mesh optimization approach based on the dynamic mode decomposition of solution update vectors. This approach involved leveraging the eigenanalysis of the small-scale projected linear mapping to identify the unstable solution modes during the numerical simulation. The DMD eigenvectors were utilized to determine appropriate vertices for mesh modification, as well as to calculate the gradients necessary for precise vertex displacement. Zandsalimy and Ollivier-Gooch [10] described a novel method for approximate eigenanalysis of the Jacobian matrix for solution mode identification. They successfully utilized this approach in the context of mesh optimization for numerical stability improvement.

Similar to previous works of [7, 8], our focus here is on identifying dominant solution modes in numerical simulations without relying on eigenanalysis of the Jacobian matrix. This alternative approach, as demonstrated in the current study, offers significant computational savings. The dynamic solution modes are obtained using the dynamic mode decomposition technique, applied to a subset of the most recent solution update vectors. Notably, the number of solution update vectors required is independent of the number of degrees of freedom, making DMD computationally much more efficient compared to the eigenanalysis of the Jacobian matrix. Compared to principal component analysis of solution update vectors, DMD provides a superior approximation of solution modes while incurring only a negligible increase in computational cost. Additionally, DMD enables early identification of problematic solution modes before they lead to solution divergence or numerical instability. This timely identification of unstable solution modes was a challenge in previous studies that laid the foundation for our current work.

Once the timely identification of the unstable solution modes is addressed, the next challenge lies in determining the specific cells and vertices that significantly influence the solution modes in question. To differentiate between numerical and physical solution modes, we employ the methodology proposed by Zandsalimy and Ollivier-Gooch [8]. This technique involves calculating the gradients of the Jacobian eigenvalues with respect to mesh movement in order to identify numerically unstable modes. Unlike physical modes, numerical modes exhibit large gradients in response to local changes in the mesh and are typically concentrated in specific control volumes with few non-zero elements. When encountering unstable numerical modes during a simulation, DMD eigenvectors also point to a local selection of cells which helps with the correct vertex identification. For each control volume of interest, we choose a single vertex to modify and perform shape optimization.

In contrast to the work of [1], a new approach is adopted herein to calculate proper vertex modification vectors for improved numerical stability of the solution. This technique involves computing the Jacobian diagonal gradients of the rows in question. Experimental results show that a dynamic system with improved Jacobian diagonal dominance offers better numerical stability. Furthermore, according to the Gershgorin circle theorem [11], better Jacobian diagonal dominance corresponds to an improved set of circles that are located further to the stable side of the eigenspectrum. Although this approach does not guarantee that all the eigenvalues of the Jacobian will lie on the stable side of the eigenspectrum after mesh optimization, it is still a step in the right direction. Further, our experimental results depict improved stability on a wide range of finite volume simulations utilizing this mesh optimization technique.

## 2. Background

### *2.1. Numerical Approach*

The finite volume method is utilized for the discretization of the Partial Differential Equations (PDE) herein. This work primarily focuses on non-linear PDEs. The general conservation equation can be written as follows.

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{F}(\boldsymbol{U}) = \boldsymbol{s}(\boldsymbol{U}) \tag{1}$$

Here, $\boldsymbol{U}$ are the conserved quantities, $\boldsymbol{F}(\cdot)$ is the flux tensor, and $\boldsymbol{s}(\cdot)$ is the source term. In this equation, the second term on the left hand side and the right hand side term are non-linear. To obtain a numerical solution to Equation 1, the solution domain is discretized into a set of non-overlapping control volumes $\Omega_i$

the union of which builds the entire solution domain. We can write the integral form of Equation 1 by integrating over each fixed control volume as follows.

$$\frac{d}{dt}\int_{\Omega_i} \bar{\boldsymbol{U}} + \oint_{\partial\Omega_i} \boldsymbol{F}(\bar{\boldsymbol{U}}) \cdot \hat{n} dS - \int_{\Omega_i} \boldsymbol{s}(\bar{\boldsymbol{U}}) d\Omega = 0 \tag{2}$$

Here, $\bar{\boldsymbol{U}}$ are the control volume averages of the conservative variables and $\hat{n}$ is the unit outward pointing normal vector from the faces of cell $\Omega_i$. The Gauss divergence theorem has been used to convert the flux divergence integral into a surface integral over the boundaries. Assuming the control volumes are fixed to space, we can pull the time derivative out of the integral by the Reynolds transport theorem. In the finite volume method, a numerical solution $\boldsymbol{U}_h$ is sought that approximates the control volume averages $\bar{\boldsymbol{U}}$. As a result, we can rewrite Equation 2 as follows.

$$|\Omega_i| \frac{d\boldsymbol{U}_h}{dt} = -\oint_{\partial\Omega_i} \boldsymbol{F}_h(\bar{\boldsymbol{U}}) \cdot \hat{n} dS + \int_{\Omega_i} \boldsymbol{s}(\bar{\boldsymbol{U}}) d\Omega = \boldsymbol{R}(\boldsymbol{U}_h) \tag{3}$$

Here, $|\Omega_i|$ is the volume of cell $\Omega_i$ and $\boldsymbol{R}(\cdot)$ is the steady state residual vector. Equation (3) can be discretized in time using an appropriate Ordinary Differential Equation (ODE) time integrator. In a number of experiments in this work, the Crank-Nicolson time advance scheme is utilized as presented in Equation (4).

$$\frac{\delta \boldsymbol{U}_h}{\delta t} = \frac{\boldsymbol{U}_h^{k+1} - \boldsymbol{U}_h^k}{\delta t} = \frac{1}{2}\left(\boldsymbol{R}(\boldsymbol{U}_h^{k+1}) + \boldsymbol{R}(\boldsymbol{U}_h^k)\right) \tag{4}$$

where $\delta t$ is the time-step size. The linearization of this equation results in the following approximation,

$$\left(\frac{1}{\delta t}\boldsymbol{I} - \frac{1}{2}\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}_h}\right) \delta \boldsymbol{U}_h = \boldsymbol{R}(\boldsymbol{U}_h^k) \tag{5}$$

Implicit Euler method (backward Euler) is another time advance scheme utilized in the current study. This approach to time integration is presented as follows,

$$\frac{\delta \boldsymbol{U}_h}{\delta t} = \frac{\boldsymbol{U}_h^{k+1} - \boldsymbol{U}_h^k}{\delta t} = \boldsymbol{R}(\boldsymbol{U}_h^{k+1}) \tag{6}$$

This methodology is locally conservative which guarantees global conservation. The surface flux integral in Equation 3 can be calculated with second-order accuracy through the following procedure.

1. Reconstruct the control volume averages using the linear least-squares method.
2. Compute the flux at each quadrature point on the boundaries of each cell. Roe's scheme [12] is used for inviscid flux calculation in the present study.
3. Find flux values using Gauss quadrature rules.

The Jacobian matrix $\dfrac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}$ can be calculated using the finite difference method or by chain rule differentiation [13] as follows.

$$\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} = \frac{\partial \text{FluxInt}}{\partial \text{Flux}} \frac{\partial \text{Flux}}{\partial \text{RecSol}} \frac{\partial \text{RecSol}}{\partial \text{RecCoef}} \frac{\partial \text{RecCoef}}{\partial \text{PVars}} \frac{\partial \text{PVars}}{\partial \text{CVars}} \tag{7}$$

Here, FluxInt is the flux integral, Flux are the numerical fluxes, RecSol are the reconstructed solutions at Gauss points, RecCoef are the reconstruction coefficients, PVars are the control volume averages of the primitive variables used in the reconstruction, and CVars are the control volume averages of the conserved variables [13]. Herein, only the matrix-vector products of the Jacobian matrix are required which enables us to use a matrix-free approach when necessary. It should be noted that the stability improvement methods presented in this study are matrix-free which does not necessitate the flow solver to be matrix-free as well. In fact, one of the flow solvers in our experiments provides explicit access to the Jacobian matrix while the other is a fully matrix-free approach.

To solve the non-linear system of equations, we have utilized a Newton-based solver that uses a line search method. In this approach, at every non-linear iteration, the approximate solution to the linear system of equations is calculated and the result is applied to update the solution vector. The Generalized Minimal Residual (GMRES) method [14] is used for the solution to the linear system. This method is available in many linear algebra packages such as Portable, Extensible Toolkit for Scientific Computation (PETSc) [15] which is our library of choice. The convergence behavior of GMRES strongly depends on the eigenspectrum of the coefficients matrix. A more compact eigenspectrum would result in fewer iterations of the linear solver for convergence. We utilize preconditioning to improve the condition number of the coefficients matrix to enhance GMRES performance.

Zandsalimy and Ollivier-Gooch [1] were able to identify unstable non-linear solutions by analyzing the eigenspectrum of the Jacobian matrix (linear coefficients matrix). According to Lyapunov stability theory [16], eigenvalues with positive real parts in the coefficients matrix will result in solution instability. The real part of an eigenmode depicts the solution mode growth rate and the imaginary part signifies the oscillation frequency. The unstable numerical modes are highly local to certain areas in the mesh used for the solution to the PDEs. Using the right eigenvectors associated with the unstable modes, Zandsalimy and Ollivier-Gooch [1] identified the problematic control volumes and vertices on the mesh whose shape and location have a considerable influence on the eigenmode in question. They were successful in calculating proper modification vectors for the selected vertices that would result in moving the unstable eigenvalues to the stable region.

Although highly effective in stabilizing unstable non-linear PDE solutions and improving the convergence rate, this method relied on the computationally expensive eigenanalysis of the large sparse Jacobian matrix. Herein, the convergence rate improvement refers to a reduction in the number of solution iterations to reach the convergence criteria. Another issue with the proposed method was the requirement for human decision to turn on the mesh optimization process at intermediate solution iterations. To rectify these issues, we proposed the use of outlier detection models on the residual history or the principal components of solution update vectors [7, 8] for unstable solution mode identification. These new methods completely automated the mesh optimization approach and removed the need for eigenanalysis. Consequently, the computational requirements of the proposed method were reduced dramatically. In these improved methods, synthetic vectors are formed out of the residual vector or the principal components of the solution that resemble and behave like the unstable eigenvectors. Such vectors were shown to have a large dot product with the eigenvectors in question which made it possible to replace the eigenvectors completely. The current study addresses the complications faced by Zandsalimy and Ollivier-Gooch [1] in a novel approach. The unstable solution mode identification is reached by means of dynamic mode decomposition of a collection of the most recent solution update vectors.

### *2.2. Dynamic Mode Decomposition*

Dynamic mode decomposition is a data reduction method to quantify the dynamic solution behavior from a selection of solution snapshots [17]. 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,

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}\boldsymbol{x}_k \tag{8}$$

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

$$\mathcal{K}_1^n(\boldsymbol{A}, \boldsymbol{x}_1) = \{\boldsymbol{A}^0\boldsymbol{x}_1, \boldsymbol{A}^1\boldsymbol{x}_1, \boldsymbol{A}^2\boldsymbol{x}_1, \ldots, \boldsymbol{A}^{n-1}\boldsymbol{x}_1\} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\} \tag{9}$$

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

$$\mathcal{K}_2^n(\boldsymbol{A}, \boldsymbol{x}_2) = \{\boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_{n+1}\} = \boldsymbol{A}\mathcal{K}_1^n = \mathcal{K}_1^n\boldsymbol{B} + \boldsymbol{r} \tag{10}$$

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,

$$\boldsymbol{B} = \arg\min_{\hat{\boldsymbol{B}}} \left\| \mathcal{K}_2 - \mathcal{K}_1\hat{\boldsymbol{B}} \right\|_F \tag{11}$$

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,

$$\mathcal{K}_1 = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^H \tag{12}$$

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

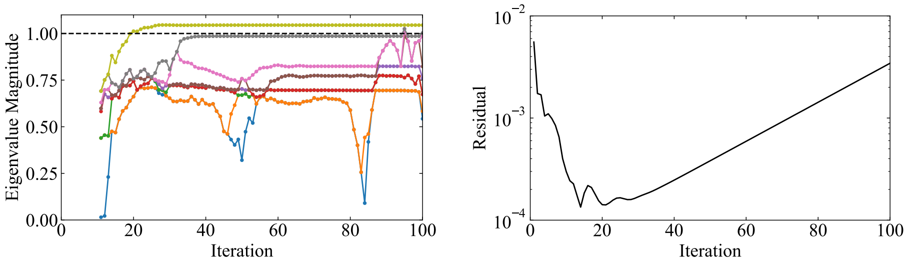
Hermitian transposed of a matrix. To generate the modified small-scale representation of the linear operator $\tilde{\boldsymbol{B}} = \boldsymbol{U}^H \boldsymbol{A} \boldsymbol{U}$, we can substitute Equation 12 in Equation 10 and rearrange to get,

$$\tilde{\boldsymbol{B}} = \boldsymbol{U}^H \mathcal{K}_2 \boldsymbol{V} \boldsymbol{\Sigma}^{-1} \tag{13}$$

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

$$\boldsymbol{w}_{\boldsymbol{A}} = \mathcal{K}_2 \boldsymbol{V} \boldsymbol{\Sigma}^{-1} \boldsymbol{w}_{\tilde{\boldsymbol{B}}} \tag{14}$$

The eigenanalysis of the small-scale matrix $\boldsymbol{B}$ is performed using a Krylov-Schur approach. Our library of choice for these computations is the Scalable Library for Eigenvalue Problem Computations (SLEPc) [18]. Note that in DMD, the large-scale mapping $\boldsymbol{A}$ is never computed at any stage of the computations. DMD provides highly accurate time dynamics of solution modes when applied to a small selection of solution update vectors. This high level of accuracy makes DMD a desirable approach for solution mode identification in numerical simulations. To test this, the 3-dimensional Advection problem is solved on a 3×1×1 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 1a in which each color depicts the history of a different eigenvalue with solution iteration. DMD eigenvalues in the current study refer to the eigenvalues of the small-scale representation of the system, $\tilde{\boldsymbol{B}}$. 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 1b depicts a log-linear growth rate after iteration 30 of this unstable problem. The Frobenius norm of the residual vector on the right hand side of Equation (3) is used to generate the residual history plots in the current study. 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.

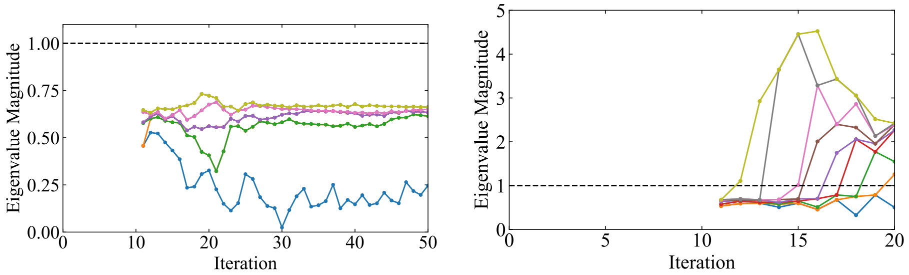


(a) DMD Eigenvalue Magnitude

(b) Residual History

Figure 1: 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 2 in which the changes to each eigenvalue are presented with a different color. Figure 2a depicts the results for a stable Burgers problem where the magnitudes of $\tilde{\boldsymbol{B}}$ eigenvalues stay smaller than 1.0. Figure 2b 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].



(a) Stable Solution

(b) Unstable Solution

Figure 2: Eigenvalue magnitude of $\tilde{\boldsymbol{B}}$ for two example Burgers problems [9]

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 $\mathcal{O}(m^3)$ 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 $\mathcal{O}(mn^2)$ which is a substantial improvement over the previous approach.

## 3. Methodology

The working details of the novel mesh optimization approach for improved numerical stability of CFD solutions are presented in this section. To address the issues faced by Zandsalimy and Ollivier-Gooch [1] (discussed in subsection 2.1), DMD is performed at every non-linear solution iteration so that unstable modes can be identified automatically. DMD vectors are highly similar to the unstable eigenvectors in the solution (dot product close to 1.0 and dominant values resolved). As a result, we can use DMD modes for problematic cell and vertex selection without performing a computationally expensive eigenanalysis. Further, a novel approach is presented to calculate vertex modification vectors that is more computationally feasible than the previous methods.

### *3.1. Solution Mode Identification*

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 1 and 2. Further, the mesh optimization module can be started as soon as eigenmodes of interest are detected for a fully automated mesh optimization procedure.

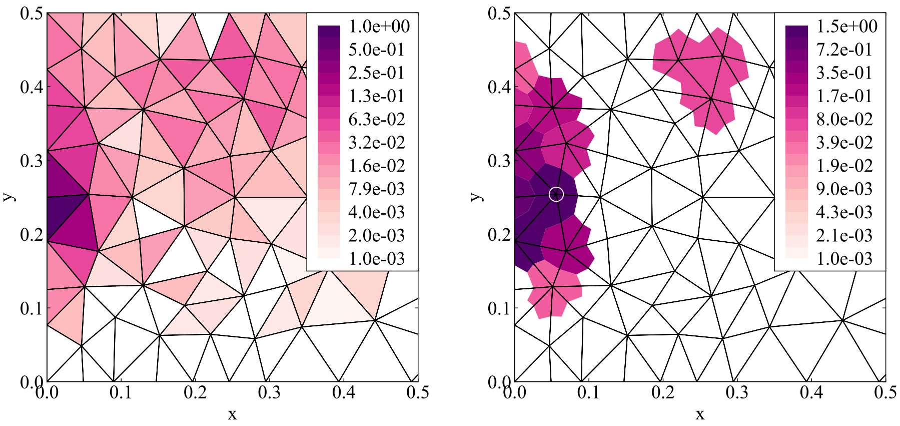
### *3.2. Cell and Vertex Selection*

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 3. Figure 3a 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 3b. 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 3b we ignore the small components in the DMD vector (magnitude smaller than 5% of the maximum entry).

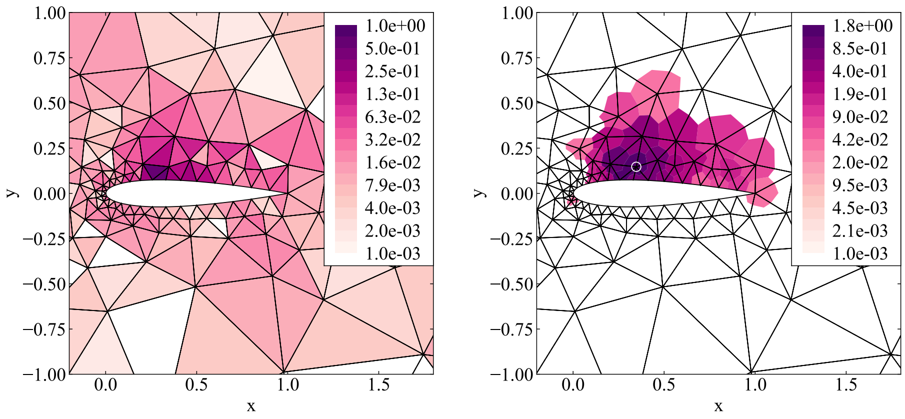


(a) Absolute value of DMD eigenvector

(b) Selection weight measure

Figure 3: Vertex selection in a Burgers problem (note the logarithmic colormap scale) [9]

In the next test, the absolute value of the DMD eigenvector corresponding to an unstable mode in an Euler problem is shown in Figure 4a. 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 4b. 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

Figure 4: Vertex selection in an Euler problem (note the logarithmic colormap scale) [9]

### *3.3. 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 [19], 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 15 presents a simple Jacobian matrix $\boldsymbol{E}$, the Gershgorin circles of which are presented in Figure 5. 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.

$$
\boldsymbol{E} = \begin{bmatrix} -11 & -2 & 0 & 1 \\ 0.5 & -7 & 0.5 & 0.5 \\ 0 & 1 & -2 & 0 \\ -1 & -1 & -1 & 5 \end{bmatrix} \tag{15}
$$

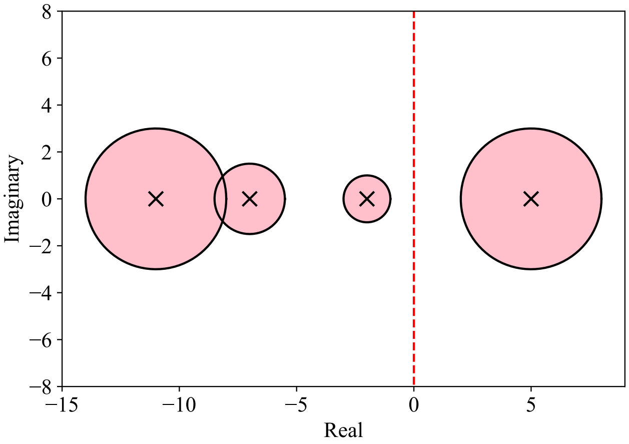


Figure 5: Example Gershgorin circles

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, $\frac{dJ_{ij}}{d\boldsymbol{\zeta}}$, we use the finite difference method as,

$$\frac{dJ_{ij}}{d\boldsymbol{\zeta}} \approx \frac{\partial J_{ij}}{\partial \boldsymbol{\zeta}} = \frac{J_{ij}(\boldsymbol{\zeta} + \delta\boldsymbol{\zeta}) - J_{ij}(\boldsymbol{\zeta})}{\delta\boldsymbol{\zeta}} \tag{16}$$

### *3.4. 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. To prevent mesh tangling we set a movement limit for each selected vertex which can be as simple as a fixed fraction of the length of the shortest incident edge on the vertex in question. In the case of vertices located on the boundary, the modification vector cannot be allowed to alter the boundary geometry. Consequently, in such cases, the vertex movement needs to be restricted to preserve the geometrical and topological features of the domain. Given these considerations, the gradient vector can be utilized to strategically adjust the location of the selected vertices to improve diagonal dominance in the Jacobian matrix.

### *3.5. Algorithm*

An overview of the presented mesh optimization approach is shown in Figure 6. The details of each computational module in this algorithm can be found in its respective section presented in Sections 3.1 to 3.4. To summarize, 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

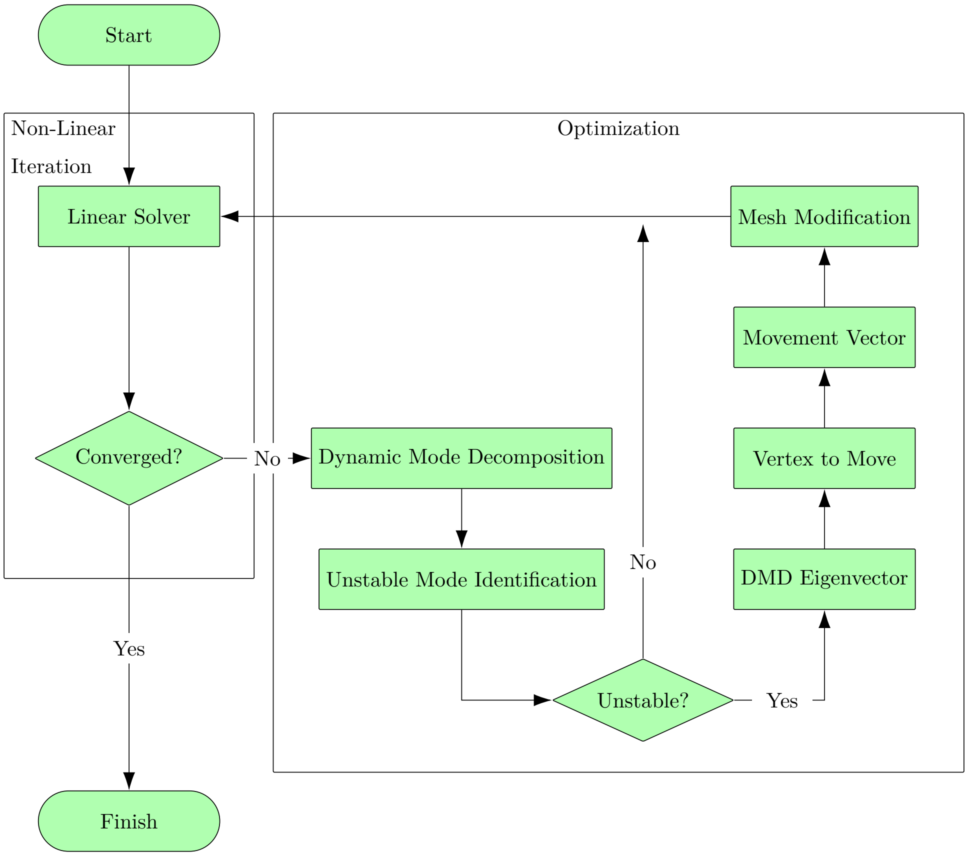


Figure 6: Overview of the mesh optimization approach [9]

around $2.17 \times 10^7$ 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.

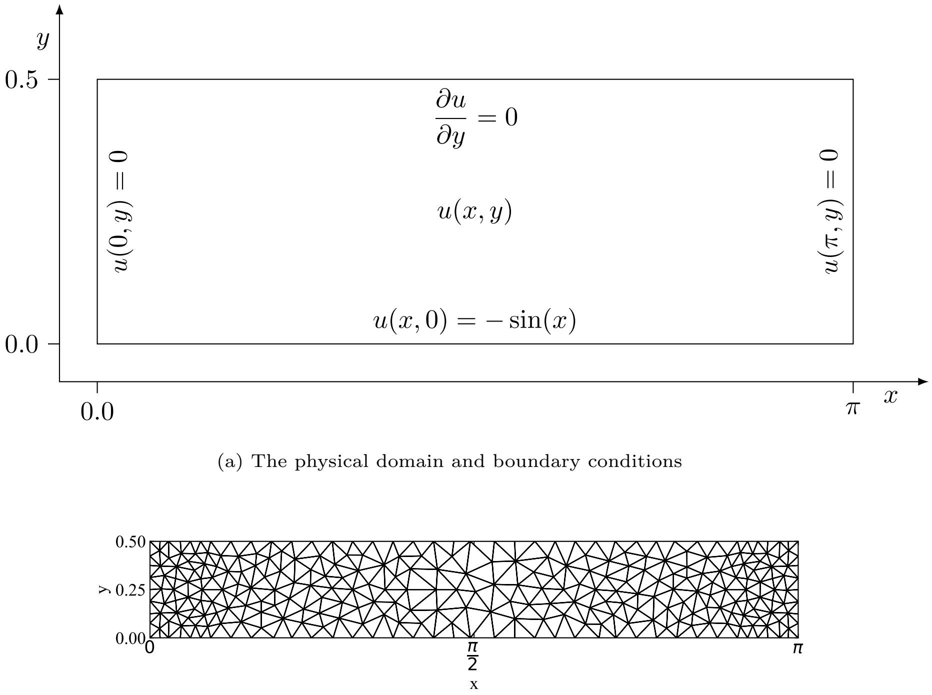
## 4. Results

We have demonstrated the efficiency of the algorithm by implementing mesh optimization on diverse test cases, highlighting the improvements in residual history. Furthermore, we have applied this method seamlessly to Ansys Fluent simulations 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. As a result, Sections 4.1 to 4.3 utilize our in-house flow solver in conjunction with the presented mesh optimization algorithm while Section 4.4 utilizes Ansys fluent simulations for mesh optimization.

### *4.1. Burgers Problem*

The non-linear inviscid Burgers problem, $\frac{\partial u}{\partial y} + u\frac{\partial u}{\partial x} = 0$, is selected as a first test case. This problem is solved on a $\pi \times 0.5$ rectangular channel with the boundary conditions presented in Figure 7a. An example of the mesh used for the solution of this problem using finite-volume methods is presented in Figure 7b. This mesh is considered high-quality according to the traditional mesh quality guidelines. Only 10 most recent solution vectors are used in the DMD process for all the tests in this section. The solution to this problem is performed using the Crank-Nicolson time-stepping method. The original solution on a mesh with 500 control volumes is unstable. Unstable DMD modes are identified (magnitude larger than 1.0) at iteration 12 of the solver. The application of our novel mesh modification approach at this iteration and restarting the simulation results in the full stabilization of the solution. The residual history of this problem before and after a single optimization iteration is presented in Figure 8a. The original and optimized meshes are presented in Figure 8b which shows that the location of a single vertex is modified for a stable problem. As depicted, the methodology presented herein results in solution convergence with around half as many iterations compared to the work of Zandsalimy and Ollivier-

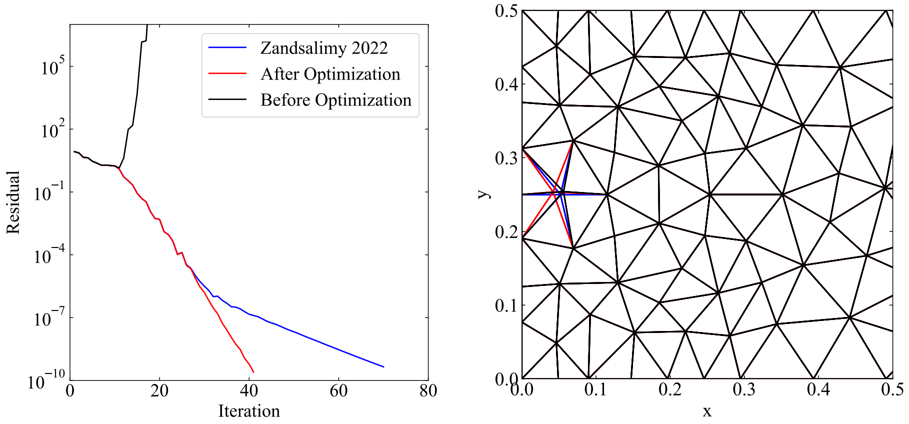
Gooch [1]. Unless otherwise indicated, by iteration we refer to the number of non-linear solution iterations of a given simulation in this study.



(a) The physical domain and boundary conditions

(b) An example of the mesh

Figure 7: The domain, boundary conditions, and mesh for the solution to the Burgers problem [1]

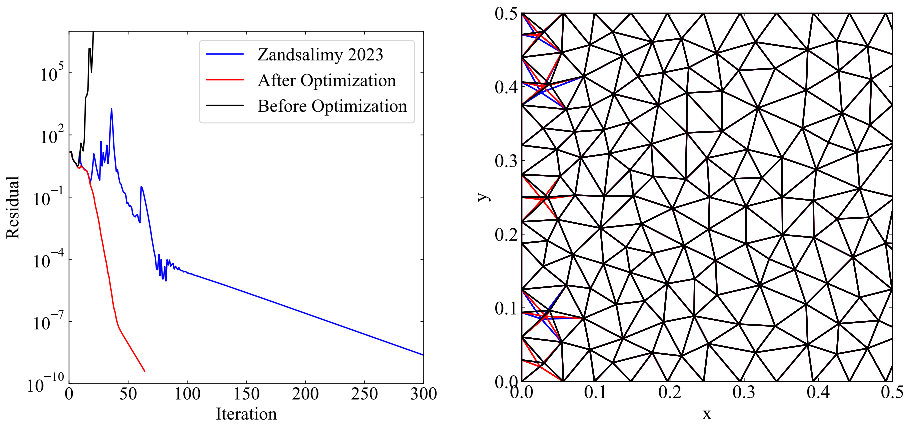


(a) The residual history

(b) Mesh modification

Figure 8: Mesh optimization in a Burgers problem on a mesh with 500 cells

The next solution to the Burgers problem is performed using the Crank-Nicolson time-stepping method on a mesh with 1100 control volumes. This problem contains multiple unstable modes which required multiple iterations of the optimization approach for full stability. Unstable DMD eigenvalues are automatically detected at iterations 13, 21, 22, 28, and 35. These five optimization iterations result in the modification of five vertices on the mesh for full stability. The residual history before and after optimization is presented in Figure 9a. The original and optimized meshes are presented in Figure 9b which shows the locations of five different vertices that are modified for a stable problem. As depicted, the methodology presented herein results in solution convergence with around $\frac{1}{7}$ as many iterations compared to the work of Zandsalimy and Ollivier-Gooch [7].

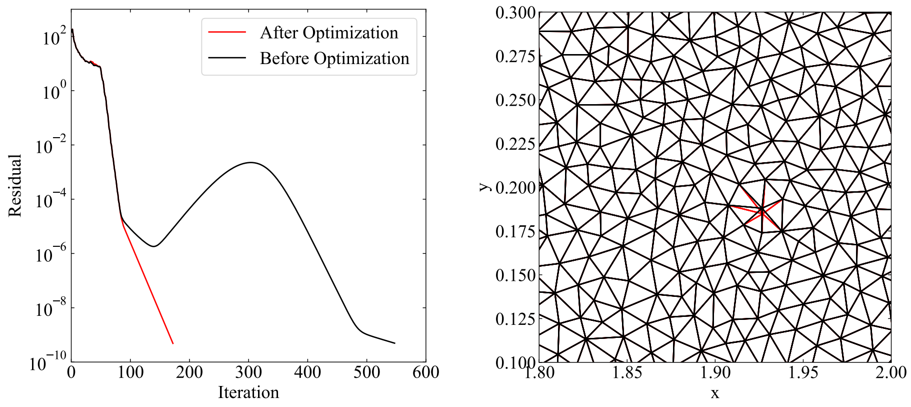


(a) The residual history

(b) Mesh modification

Figure 9: Mesh optimization in a Burgers problem on a mesh with 1100 cells

The next Burgers problem is solved using the Crank-Nicolson time-stepping method on a mesh with 17500 cells. This problem is stable but some slow converging modes are present in the simulation. Numerical modes with magnitudes larger than 0.96 are detected automatically at iterations 105, 154, and 160. Performing the optimization at these iterations results in the residual history presented in red in Figure 10a which depicts convergence in around $\frac{1}{3}$ of the solution iterations. The locations of 4 vertices are modified in this case, one of which is indicated in Figure 10b. Once again, we note that only the last 10 solution update vectors are used in the mesh optimization of this problem containing 17500 degrees of freedom.



(a) The residual history

(b) Mesh modification

Figure 10: Mesh optimization in a Burgers problem using Implicit Euler time integration on a mesh with 17500 cells

### *4.2. Advection*

The 3D Advection problem, $\frac{\partial \psi}{\partial t} + \boldsymbol{u} \cdot \boldsymbol{\nabla} \psi = 0$, is used as the next test case in the current study. Here, $\psi$ is a scalar, $\boldsymbol{u} = 1.0\vec{e_i} + 0.0\vec{e_j} + 0.0\vec{e_k}$ is a 3D velocity field, and $\boldsymbol{\nabla} = \frac{\partial}{\partial x}\vec{e_i} + \frac{\partial}{\partial y}\vec{e_j} + \frac{\partial}{\partial z}\vec{e_k}$. This problem is solved in a $3 \times 1 \times 1$ channel using the Crank-Nicolson time-stepping method. For all the tests in this section, only 10 solution update vectors are used in the DMD computations. The first test case is solved on a mesh with 500 tetrahedral cells which is unstable as shown in Figure 11 in black. In this case, unstable DMD modes are identified at iteration 11 of the solver. The mesh optimization algorithm is applied at this iteration modifying a single vertex which stabilizes the solution. The residual history before and after optimization is presented in Figure 11.

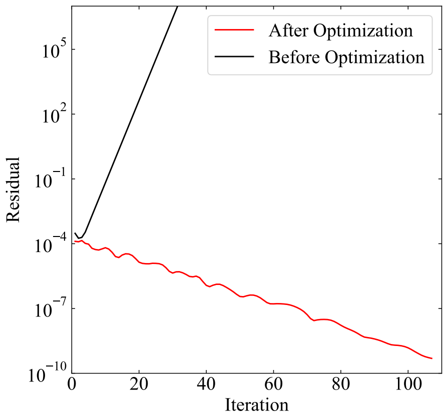


Figure 11: Residual history in mesh optimization of an Advection problem with 500 cells

475 The next simulation is performed on a mesh with around 30000 cells which is originally unstable. First unstable solution modes are detected at iteration 11 of the solver. The mesh optimization approach results in modifying the location of one vertex on the mesh and fully stabilizes the simulation. The residual histories for this test case are shown in Figure 12.

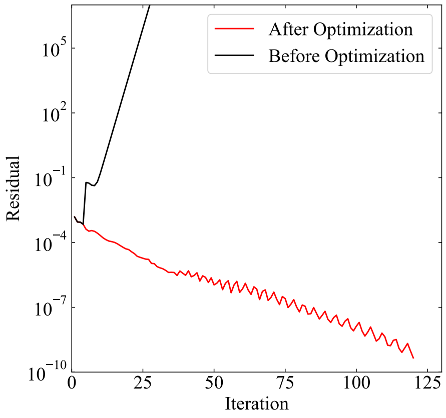
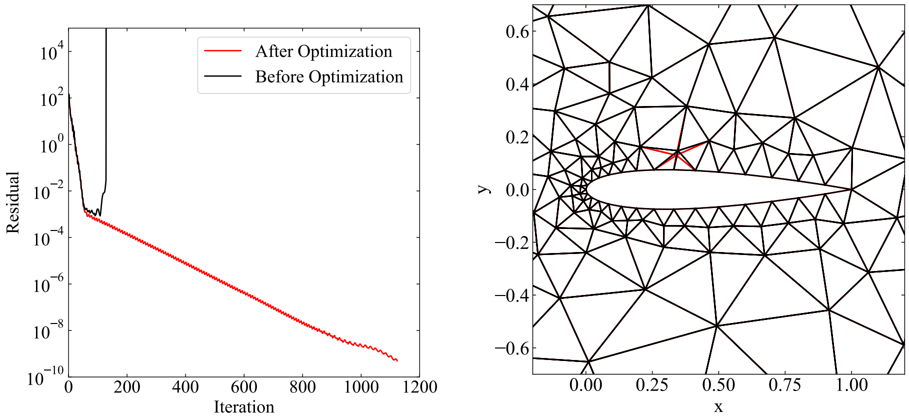


Figure 12: Residual history in mesh optimization of an Advection problem with 29100 cells

## 480 *4.3. Euler Problem*

The Euler problem is selected as the next test case in the present study which is solved around the NACA 0015 airfoil inside a circular domain with a radius of 500 chords. The flow variables are nondimensionalized with the free-stream conditions for a general solution in non-dimensional form. In the initial solution for this problem, free-stream density $\rho_\infty$ is set to 1.0, velocity in $(x, y)$ direction to $M_\infty(\cos(\alpha), \sin(\alpha))$, and free-stream pressure $P_\infty$ to $\frac{1}{\gamma}$. In these relations, $M_\infty$ is the free-stream Mach number, $\alpha$ is the angle of attack, $P_\infty$ is the resulting pressure during isentropic expansion to $M_\infty$, and $\gamma$ is the heat capacity ratio of the fluid. Further in this problem, stagnation values are set from the isentropic relations. Only 10 most recent solution vectors are used in the DMD process for all the tests in this section. The first mesh to be tested contains 600 control volumes. The simulation using the Crank-Nicolson time-stepping method with CFL $= 1.0$, $M_\infty = 0.5$ and $\alpha = 0$ is unstable. An unstable DMD mode with a magnitude larger than 1.0 is detected at iteration 103 of the solver. Performing the optimization at this iteration results in the residual history presented in red in Figure 13a. The optimized mesh is presented in Figure 13b depicting the modification of a single vertex.

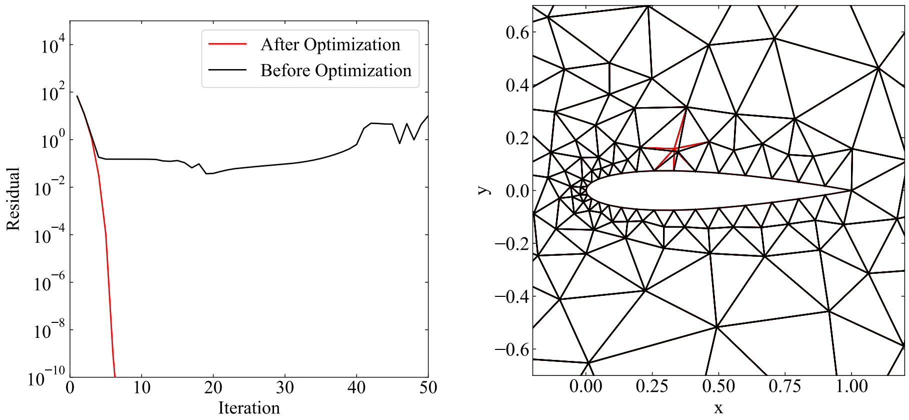


(a) The residual history

(b) Mesh modification

Figure 13: Mesh optimization in an Euler problem using Crank-Nicolson time integration on a mesh with 600 cells

This problem is solved on the same mesh using the Implicit Euler time-stepping method. CFL evolution strategies [20] are utilized to increase the convergence rate of the problem. In this case, unstable DMD modes are detected at iteration 13 of the solver. Performing the optimization at this point results in the residual history presented in red in Figure 14a. The modified mesh is presented in Figure 14b. As seen, only a single vertex is modified to reach a fully stabilized solution.

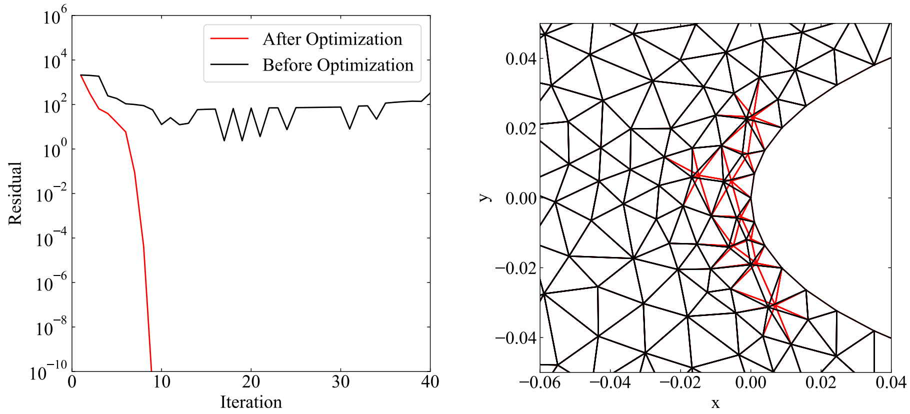


(a) The residual history

(b) Mesh modification

Figure 14: Mesh optimization in an Euler problem using Implicit Euler time integration on a mesh with 600 cells

The next Euler problem is solved using the Implicit Euler time-stepping method on a mesh with 7600 cells. A few unstable DMD modes are detected at iteration 11 of the solver. Performing the optimization at this iteration results in the residual history presented in Figure 15a. The optimized mesh is presented in Figure 15b depicting the modification of 9 vertices near the leading edge of the airfoil. Once again, we note that only 10 solution update vectors are used for mesh optimization in this problem containing 30400 degrees of freedom.



(a) The residual history

(b) Mesh modification

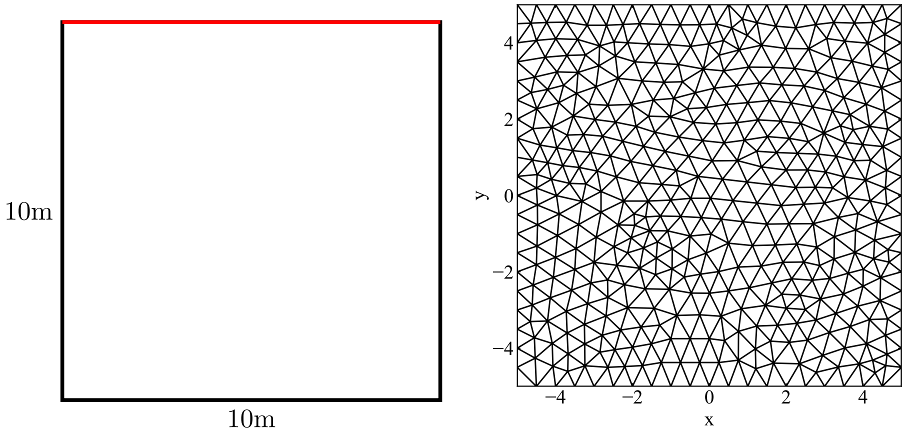
Figure 15: Mesh optimization in an Euler problem using Implicit Euler time integration on a mesh with 7600 cells

### *4.4. Ansys Fluent*

The presented mesh optimization approach offers a significant advantage due to its non-invasive nature when integrated with external solvers. Our primary objective was to develop an algorithm with seamless integration without necessitating any modifications to the underlying flow solver architecture. This crucial goal has been successfully accomplished in our current study.

In our experimentation, the mesh optimization approach was applied to Ansys Fluent version 2023 R2, showcasing its effortless integration. Notably, this process did not require access to the source code. In simpler terms, anyone with only a valid user license for Ansys Fluent software can replicate the results demonstrated in this study. This demonstrates the accessibility and usability of the presented approach for a wide range of industrial applications without delving into complex code modifications. Note that the current section includes Ansys Fluent specific nomenclature (e.g. boundary condition types and turbulence models) that are explained in the software manual.

The lid-driven cavity is chosen as the first test case in Ansys Fluent. Figure 16a shows the 10m × 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 16b 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.

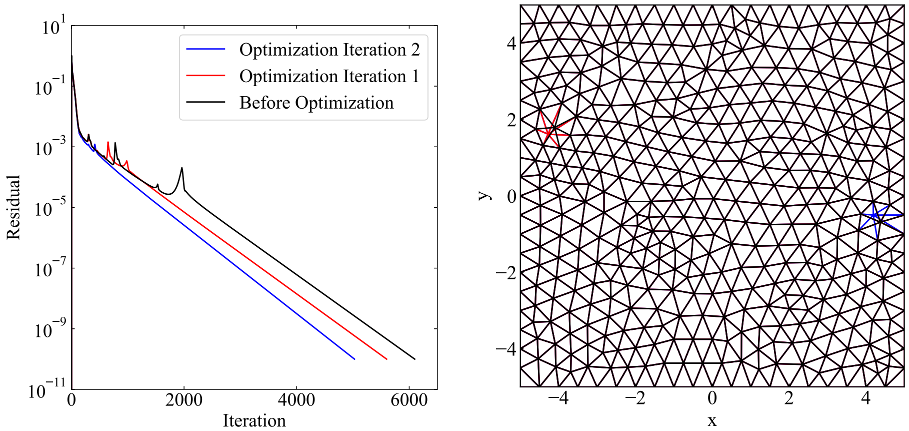


(a) The physical domain

(b) An example of the mesh

Figure 16: 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 17a 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 red vertex in Figure 17b with the improved residual history depicted in Figure 17a. 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 blue vertex presented in Figure 17b with the improved residual history depicted in Figure 17a.

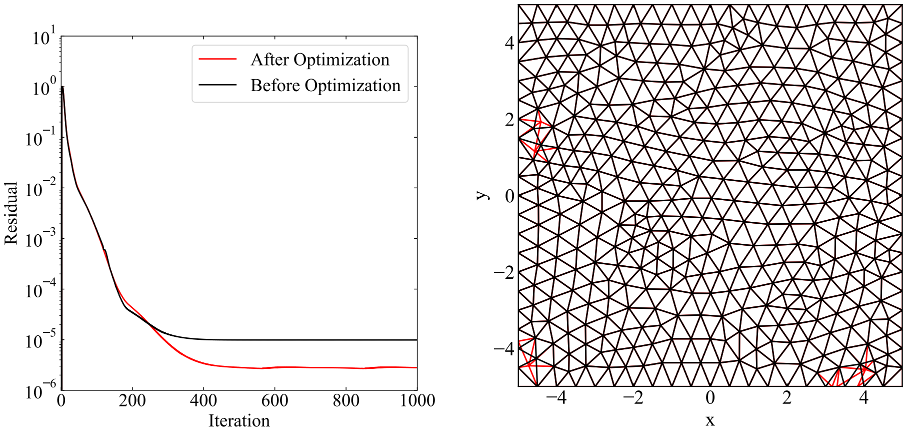


(a) The residual history

(b) Mesh modification

Figure 17: Ansys Fluent mesh optimization in a 2D cavity problem

550 The same problem is used for the next test case with a few modifications. The $k-\omega$ SST turbulence model is used for a proper turbulence modeling simulation. 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 18b. The final residual in this case undergoes the improvement depicted in Figure 18a.

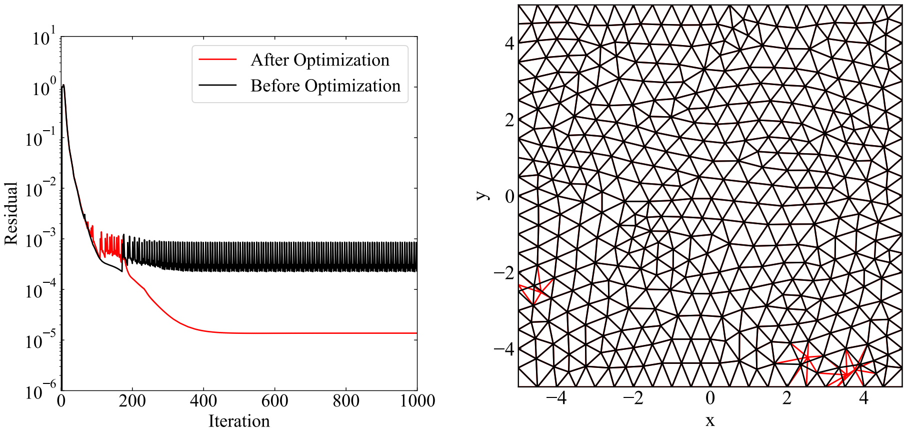


(a) The residual history

(b) Mesh modification

Figure 18: Ansys Fluent mesh optimization in a 2D cavity problem

The same simulation is performed with the velocity of the top wall set to $0.1\frac{\text{m}}{\text{s}}$ with the $k-\omega$ SST turbulence model. In this case, the Reynolds number is 68,500. 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 19b. The final residual in this case undergoes the improvements shown in Figure 19a.



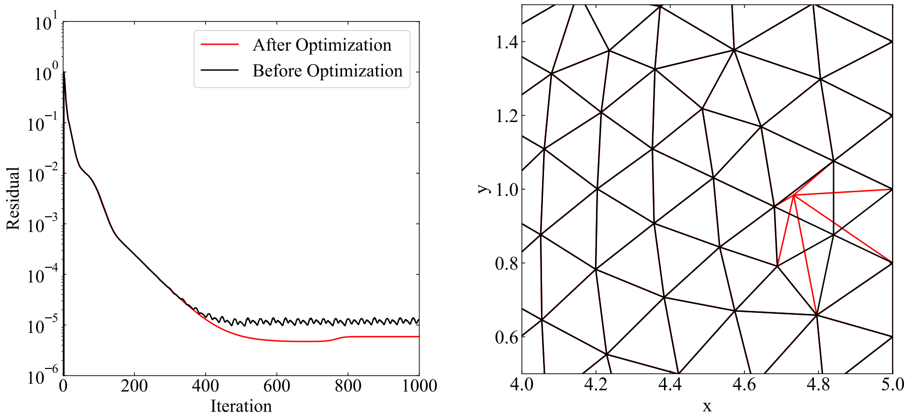
(a) The residual history

(b) Mesh modification

Figure 19: Ansys Fluent mesh optimization in a 2D cavity problem

A finer triangular mesh is generated to solve this problem with 5194 cells, 7891 faces, and 2698 nodes. The top wall velocity is set to $10.0\frac{\mathrm{m}}{\mathrm{s}}$ for a Reynolds number of 6,850,000. Once again, the $k-\omega$ SST turbulence model is used in the simulation. DMD is performed on the latest 10 solution update vectors to identify solution modes. At iteration 128, the dominant DMD mode is used for mesh optimization to modify the single vertex presented in Figure 20b. The final residual in this case undergoes the improvements depicted in Figure 20a.

As seen here, the optimized mesh contains a single cell with an internal angle of around 180 degrees which is considered low-quality by traditional metrics. Zandsalimy and Ollivier-Gooch [1] studied this behavior and presented examples of better-performing numerical simulations on meshes that are considered low-quality by traditional measures in comparison to high-quality meshes. The numerical simulation performance on a given mesh depends strongly on the physics, discretization scheme, boundary conditions, physical instabilities in the solution, and so on. Consequently, it is implausible to make a concrete decision on the numerical stability and convergence, solely based on traditional mesh quality measures. Further, the counter-example of Figure 20 also shows the deficiency of such traditional quality metrics in mesh generation for numerical stability and convergence.

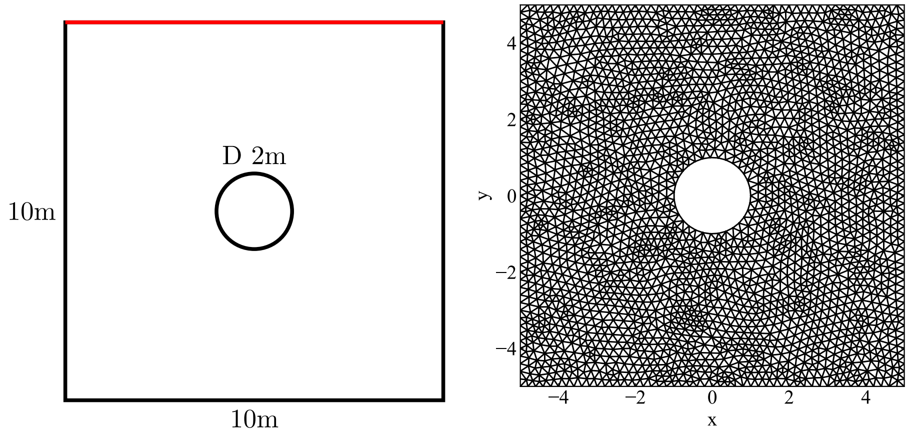


(a) The residual history

(b) Mesh modification

Figure 20: Ansys Fluent mesh optimization in a 2D cavity problem

A modified lid-driven cavity is chosen as the second test case in Ansys Fluent. Figure 21a shows the 10m × 10m solution domain of this simulation with a 2m diameter hole punched in the middle. Here, the black lines are set to stationary no-slip walls and the red line is set to moving no-slip wall. Figure 21b 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.

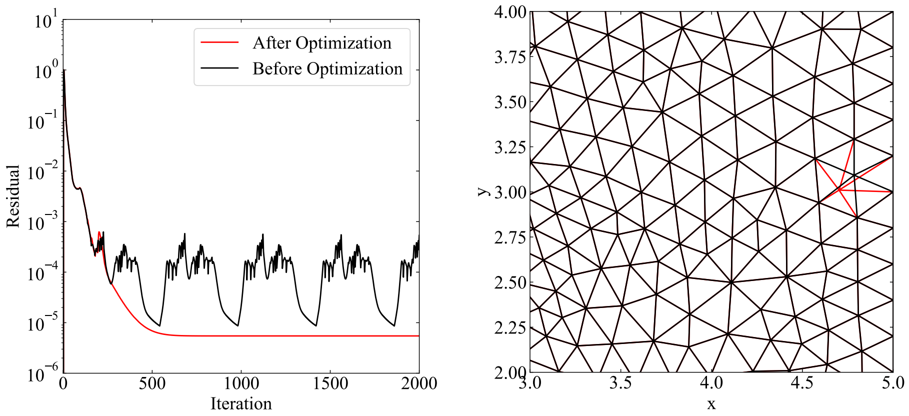


(a) The physical domain

(b) An example of the mesh

Figure 21: Solution domain and mesh for a modified lid-driven cavity problem

For this test case, the top wall moves at a velocity of $1\frac{\mathrm{m}}{\mathrm{s}}$ to the right for a Reynolds number of 685,000. The $k - \omega$ SST turbulence model is used for this simulation. The mesh includes 5209 triangular cells, 7929 faces, and 2720 nodes generated with the Ansys Meshing software. DMD is performed on the latest 10 solution update vectors to identify solution modes. One optimization iteration is performed on the dominant solution mode at iteration 350 of the solver. This optimization iteration results in the modification of the single vertex presented in Figure 22b. The improved residual history after mesh optimization is shown in Figure 22a.



(a) The residual history

(b) Mesh modification

Figure 22: Ansys Fluent mesh optimization in a modified 2D cavity problem

The 3D lid-driven cavity is chosen as the third test case in Ansys Fluent. The domain is a simple 10m × 10m × 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 23 shows the residual history before and after the application.

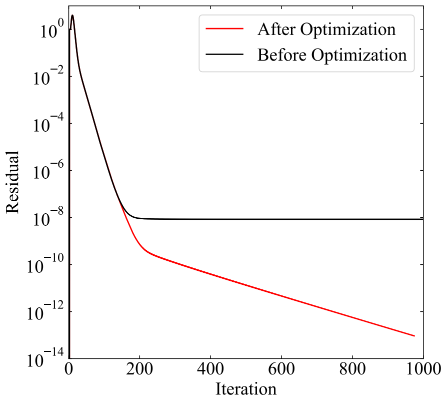


Figure 23: Ansys Fluent mesh optimization in a 3D cavity problem

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 24 in red. 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 24 in blue. The final optimization iteration is performed at iteration 100 of the new solution to modify the locations of 15 vertices. As seen in Figure 24 in green, the final residual is substantially improved over the initial solution.

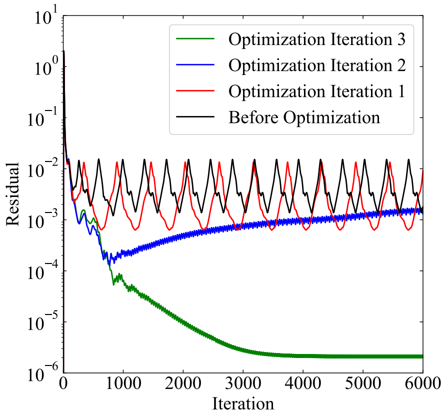
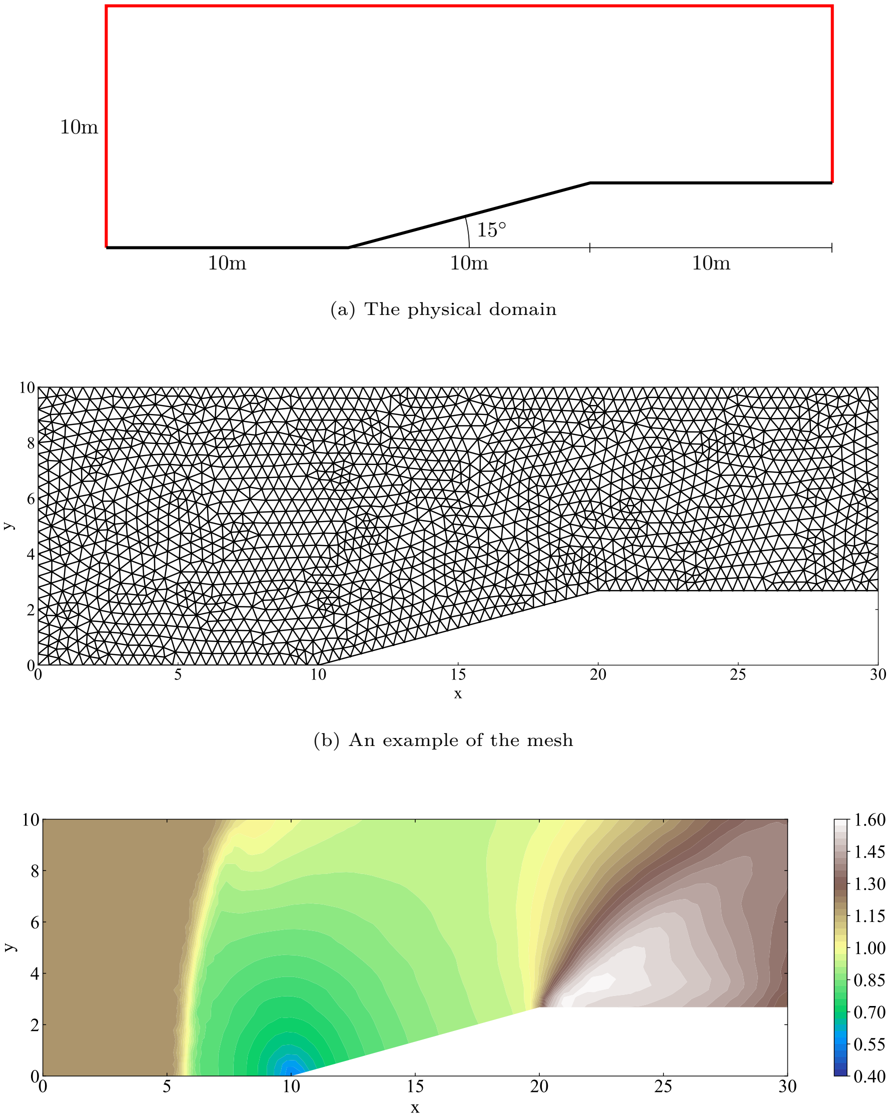


Figure 24: Ansys Fluent mesh optimization in a 3D cavity problem

A compressible flow problem is selected as the next test case in Ansys Fluent. Figure 25a shows the solution domain of the simulation with the black lines set to zero shear walls and the red lines set to pressure far-field boundary condition at Mach 1.2. Figure 25b depicts a typical triangular mesh that was used for this simulation. The ideal gas law is applied to the fluid for a compressible flow simulation with constant viscosity of $1.7894 \times 10^{-5} \frac{\mathrm{kg}}{\mathrm{m \cdot s}}$, specific heat capacity $C_p$ of $1006.43 \frac{\mathrm{J}}{\mathrm{kg \cdot K}}$, thermal conductivity of $0.0242 \frac{\mathrm{W}}{\mathrm{m \cdot K}}$, and molecular weight of $28.966 \frac{\mathrm{kg}}{\mathrm{kmol}}$.

The Ansys Fluent version 2023 R2 used in the current study does not support the Adjoint solution for inviscid flows which is required to find the Jacobian matrix. Consequently, we are not able to test the presented stability improvement approach on the current compressible flow case using an inviscid solution. Although this simulation may deviate from physical accuracy, it serves as a substantiation of the proposed stability improvement methodology. Notably, it demonstrates the methodology's efficacy in enhancing the solution with minimal computational overhead. Figure 25c shows the contours of Mach number in the converged solution for this problem.



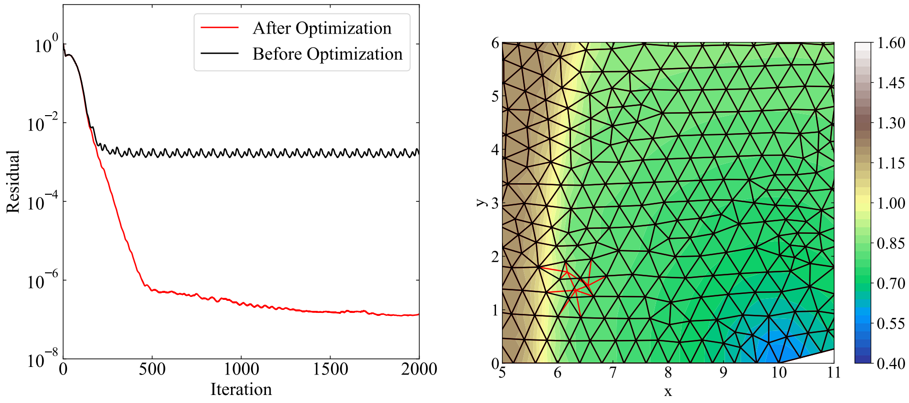
(a) The physical domain

(b) An example of the mesh

(c) An example of the mesh

Figure 25: Solution domain and mesh for a compressible flow problem

In this case, the triangular mesh includes 3348 cells, 5119 faces, and 1772 nodes with the $k - \omega$ SST turbulence model selected for the solution. DMD is performed on the latest 10 solution update vectors to identify solution modes. One optimization iteration is performed on the dominant solution mode at iteration 300 of the solver to modify the location of 3 vertices on the mesh shown in Figure 26b. These vertices are located just downstream of the shock wave. The improved residual history after mesh optimization is depicted in Figure 26a.

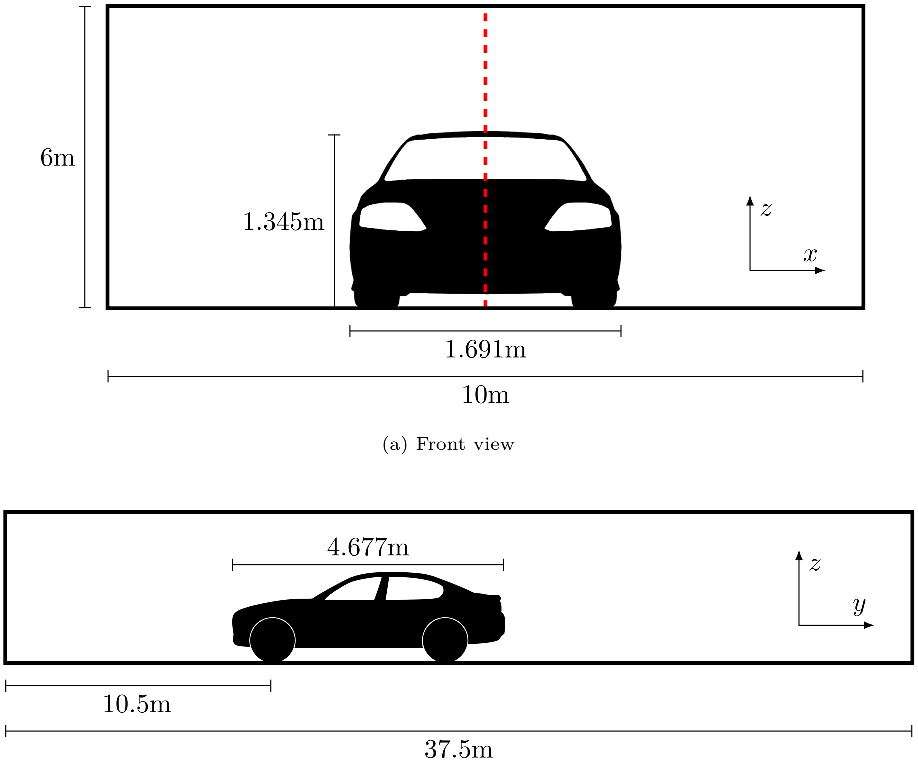


(a) The residual history

(b) Mesh modification

Figure 26: Ansys Fluent mesh optimization in a compressible flow problem

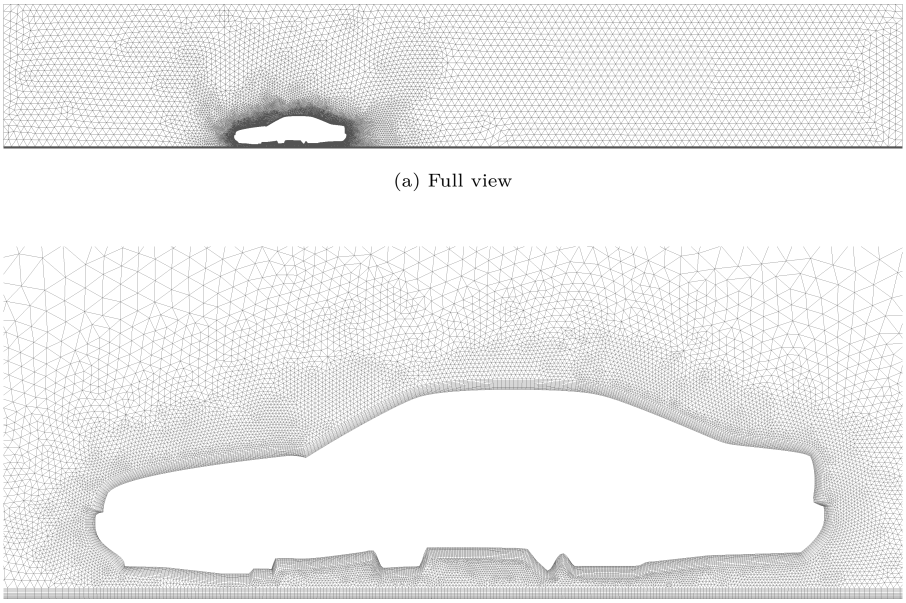
An incompressible 3D turbulent flow around a sedan car is selected as the next test case in Ansys Fluent. Figure 27 shows the front and side views of the solution domain with the dashed red line used as a symmetry plane in the simulation. The car body, wheels, and the floor plane are set to no slip walls. The velocity inlet boundary condition is set to $27.8\frac{\text{m}}{\text{s}}$ 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 28 shows the mesh on the symmetry plane. This unstructured 3D mesh includes 1,806,224 prism cells, 1,811,856 tetrahedral 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 our 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 $1.83 \times 10^{-5}\frac{\text{kg}}{\text{m}\cdot\text{s}}$, 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 29a. Figure 29b 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 30 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

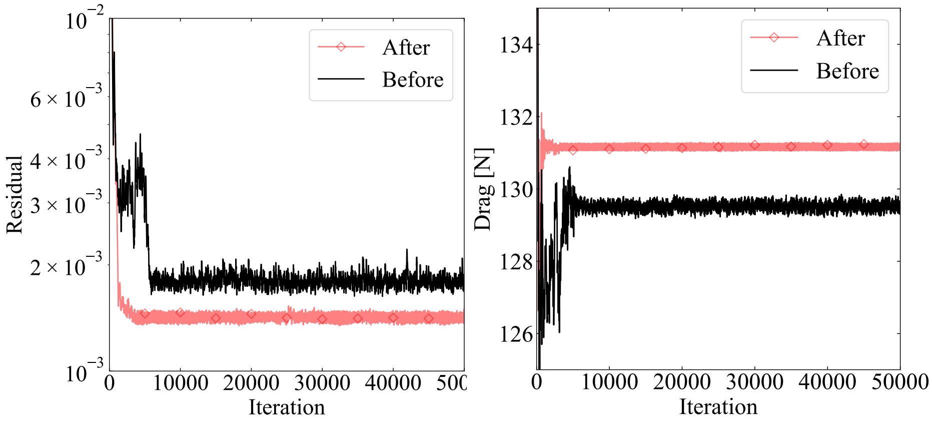
Figure 27: Solution domain for a large-scale 3D turbulent automotive problem



(a) Full view

(b) Zoom view

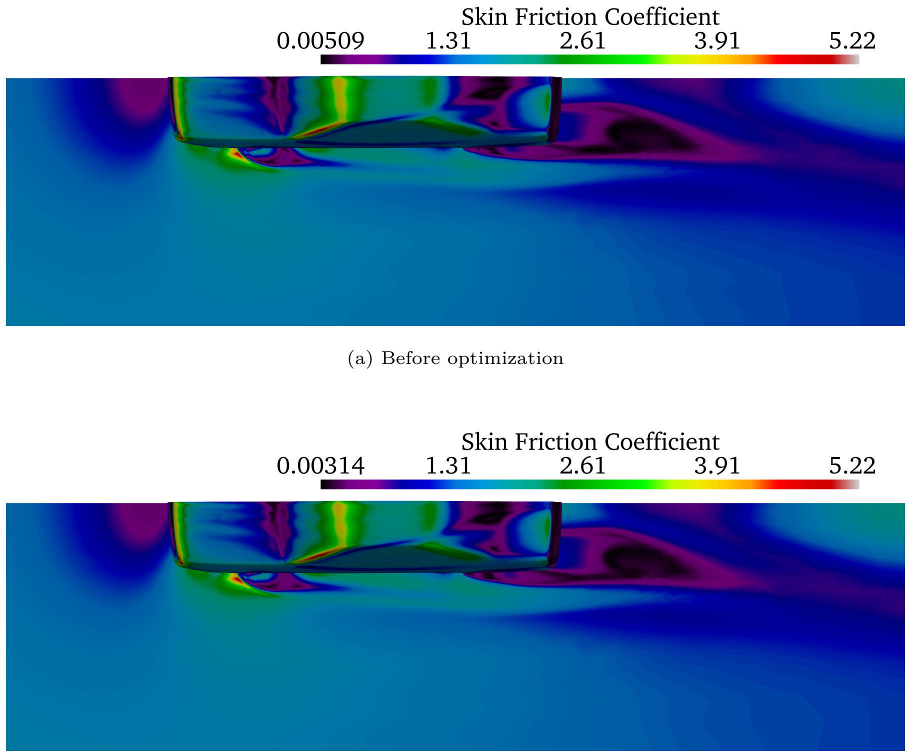
Figure 28: Symmetry plane mesh for a large-scale 3D turbulent automotive problem



(a) Residual history

(b) Drag history

Figure 29: Ansys Fluent mesh optimization in a large-scale turbulent automotive problem



(a) Before optimization

(b) After optimization

Figure 30: Skin friction coefficient during mesh optimization in a large-scale turbulent automotive problem

The NASA Common Research Model (CRM) is selected as the next test case in this study. 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. 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 31. The details of this geometry and configurations are given in the work of Lacy and Clark [21].

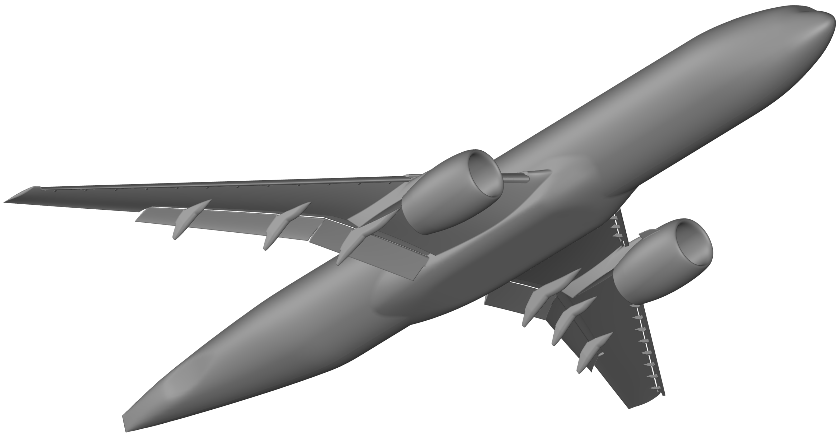


Figure 31: CRM-HL wing-body-pylon-nacelle configuration along with high-lift devices isometric view

We 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. 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 $11^{\circ}$ 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. [22], with 1,051,621 tetrahedral cells, 21,297 pyramid cells, 1,509,377 prism 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 $4.4734 \times$

$10^4$ and the minimum orthogonal quality metric is $2.1024 \times 10^{-6}$. 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 3D 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 32 with white. This optimization iteration results in half an order of magnitude reduction in the residual history presented in Figure 33a. Figures 33b and 33c 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 34. As depicted, the skin friction coefficient patterns are similar in both solutions with small noticeable differences.

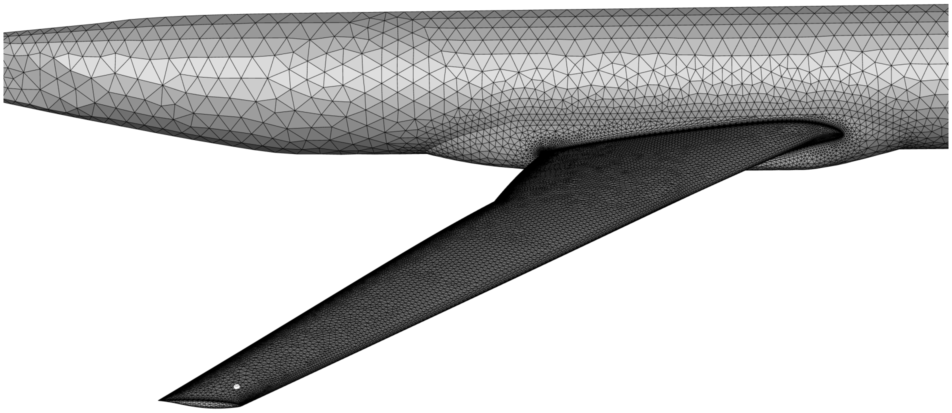
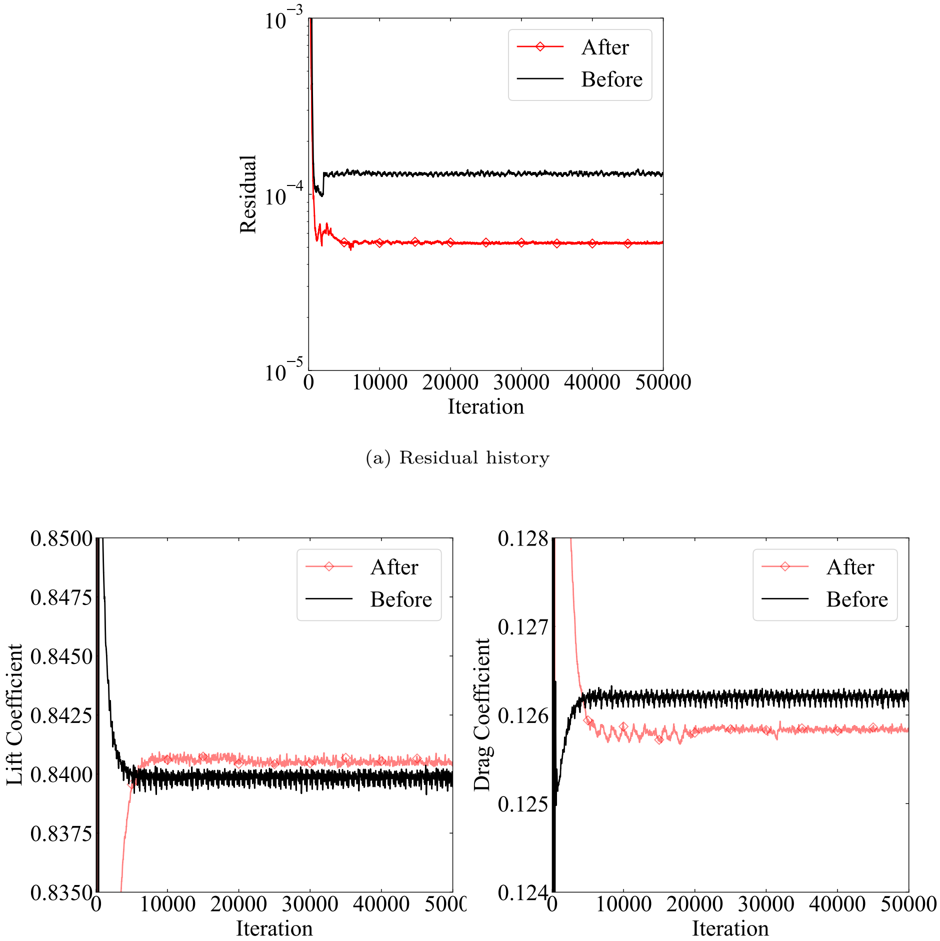


Figure 32: The modified vertices during mesh optimization in CRM-HL wing-body configuration

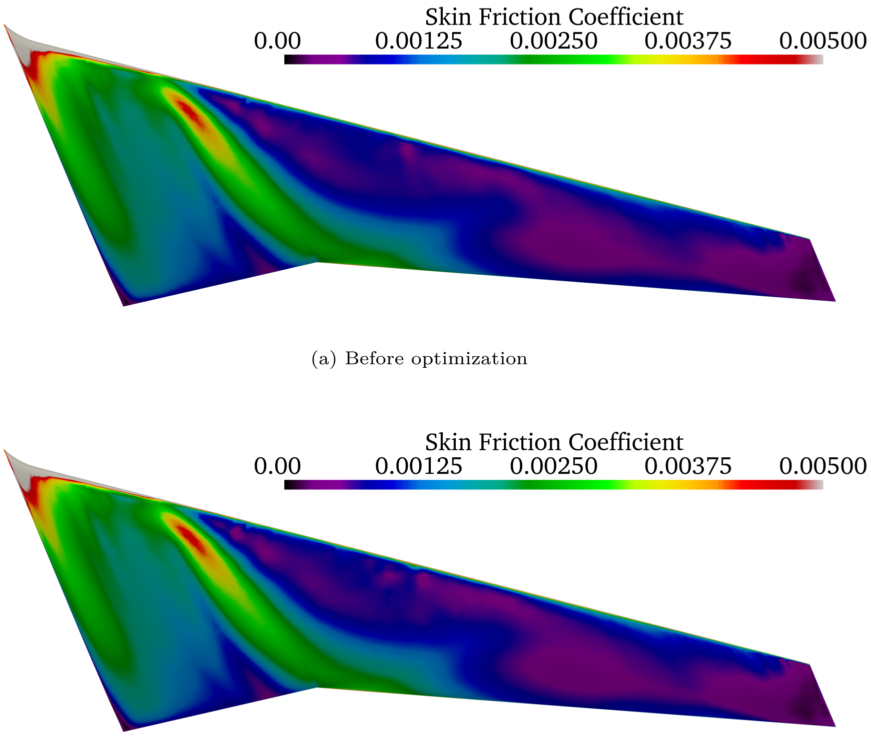


(a) Residual history

(b) Lift coefficient history

(c) Drag coefficient history

Figure 33: Ansys Fluent mesh optimization in CRM-HL wing-body configuration



(a) Before optimization

(b) After optimization

Figure 34: 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 35 with white. This optimization iteration results in half an order of magnitude reduction in the residual history presented in Figure 36a. Figures 36b and 36c 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 37. As depicted, the skin friction coefficient patterns are similar in both solutions with small noticeable differences.

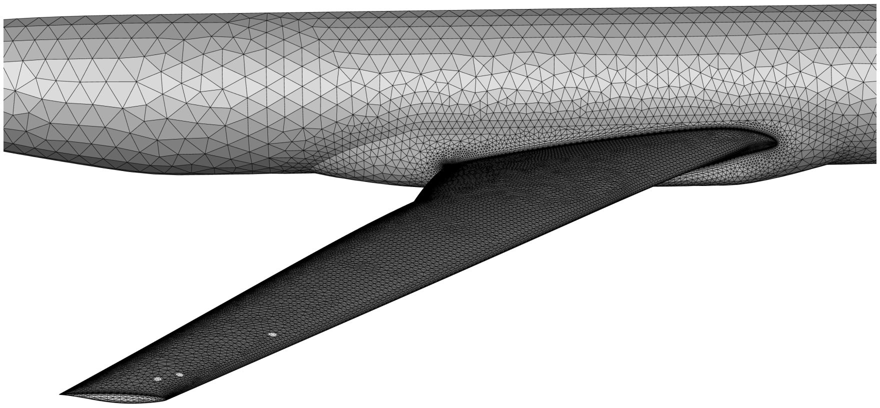
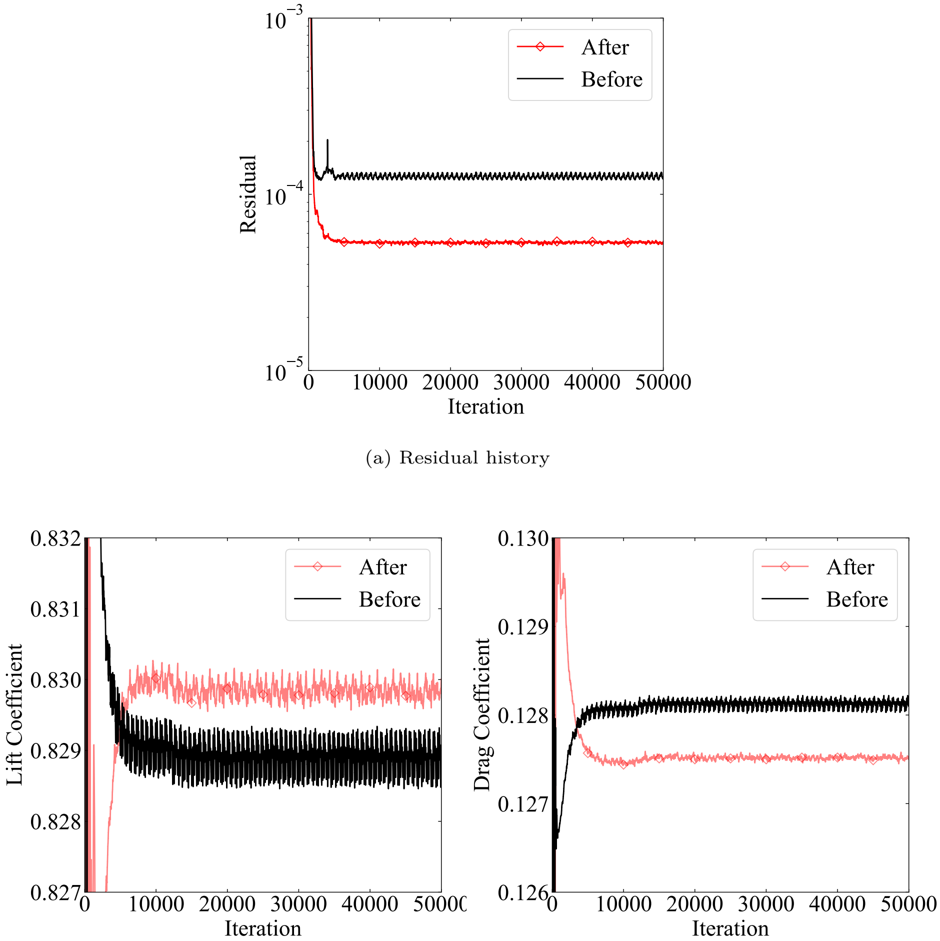


Figure 35: The modified vertices during mesh optimization in CRM-HL wing-body configuration

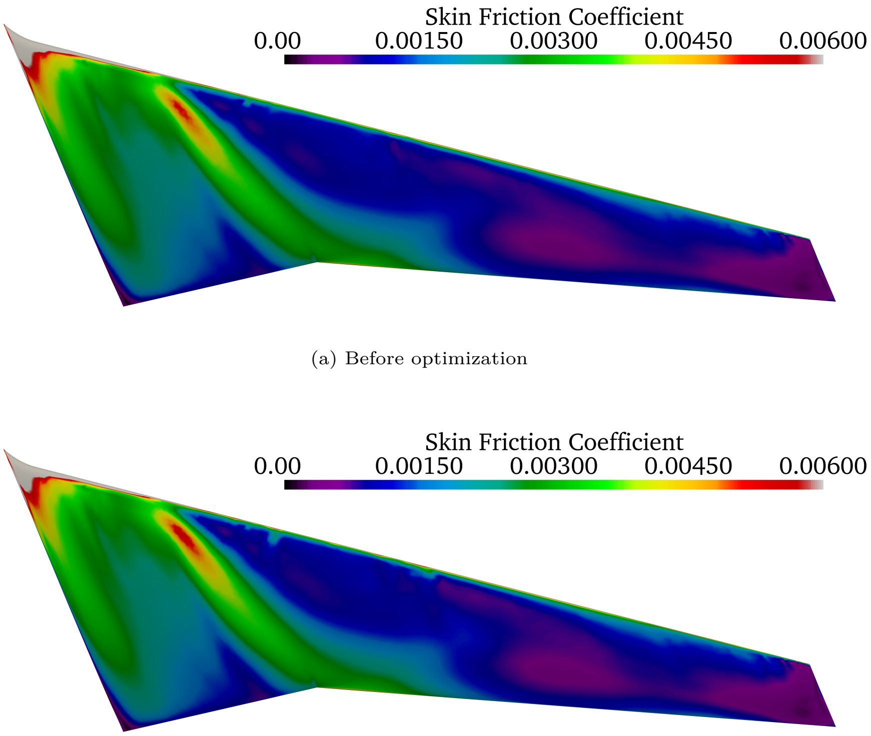


(a) Residual history

(b) Lift coefficient history

(c) Drag coefficient history

Figure 36: Ansys Fluent mesh optimization in CRM-HL wing-body configuration



(a) Before optimization

(b) After optimization

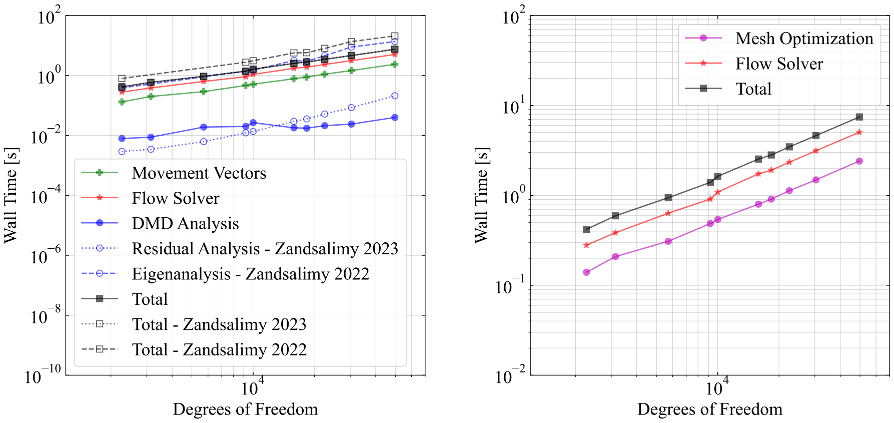
Figure 37: Skin friction coefficient during mesh optimization in CRM-HL wing-body configuration

## *4.5. Computational Cost*

The computational cost of the presented algorithm in conjunction with our in-house flow solver is presented in the current section. The computational time is computed in the solution and mesh optimization of an Euler problem on different mesh sizes. These results are quite similar to the test cases performed in Ansys Fluent with the only difference being the flow simulation times. One of the shortcomings of the previous studies on mesh optimization using numerical solution mode enhancement was the high computational complexity which hinders their approach from being applied to large turbulent simulations. The current study eliminates the need for eigenanalysis of the Jacobian matrix by employing Dynamic Mode Decomposition of a small selection of solution update vectors. This results in substantial computational savings compared to the work of [1]. This improvement is depicted with a solid blue line compared to the dashed blue line in Figure 38a.

As seen, the overall mesh optimization process is much faster in comparison to [1], mainly due to the complete elimination of the eigenanalysis module. Furthermore, optimization run-time is similar to [7] noting that the DMD Analysis module in the current work exhibits a smaller growth rate with increasing degrees of freedom compared to the Residual Analysis module. Furthermore, the computations concerning Movement Vectors and Flow Solver modules are similar across the board.

Further, the run time of our DMD mesh optimization (the box labeled Optimization in Figure 6) is the summation of the DMD Analysis (solid blue line) and Movement Vectors (solid green line) in Figure 38a. This summation is presented in Figure 38b to compare with the computational run-time of the flow solver. For different degrees of freedom, this figure depicts the wall time of a single DMD mesh optimization in magenta and the run-time of a single solution iteration in red. As shown, the cost of every iteration of our mesh optimization approach is smaller than each non-linear solution iteration. As presented in the current study, usually a single mesh optimization iteration is sufficient to fully stabilize an unstable simulation or to substantially reduce the number of iterations to reach numerical convergence. This confirms the substantial computational savings achieved through our presented methodology for stability improvement in computational fluid dynamics.



(a) Individual modules

(b) Mesh optimization versus solution

Figure 38: The optimization time versus degrees of freedom in an Euler problem

## 5. Conclusion

Novel strategies are unveiled in the current study for mesh optimization of finite-volume methods to enhance the residual convergence behavior. A key feature involves utilizing Dynamic Mode Decomposition of a small collection of the most recent solution update vectors to identify problematic solution modes. The automatic unstable mode identification is an important enhancement over the previous mesh optimization algorithms. By employing DMD eigenvectors associated with these modes appropriate vertices are selected and modification vectors are calculated, leading to favorable outcomes. Further, it is shown that the presented mesh optimization methodology is non-invasive in nature and can be used in conjunction with external flow solvers without any access to the underlying software architecture. To validate this, the presented algorithm is used in conjunction with Ansys Fluent simulations with the results depicting substantial enhancements in the residual history across various problems. The current work pioneers the application of the presented algorithm to fully turbulent simulations to show its robustness and effectiveness. Further, we have shown that minor alterations to a small collection of vertices on the mesh can make considerable changes to the aerodynamic characteristic predictions. Consequently, our study raises an important concern about traditional mesh quality metrics and the convergence criteria of numerical simulations. We believe that the CFD community needs to define more rigorous definitions of numerical convergence and mesh quality for further advancements in the field.

## 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. 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, Ollivier-Gooch C. A novel approach to mesh optimization to stabilize unstructured finite volume simulations. *Journal of Computational Physics* 2022;453:110959. doi:`10.1016/j.jcp.2022.110959`.

2. Knupp P. Label-invariant mesh quality metrics. In: *Proceedings of the 18th International Meshing Roundtable*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009:139–55.

3. Habashi WG, Dompierre J, Bourgault Y, Fortin M, Vallet MG. Certifiable computational fluid dynamics through mesh optimization. *AIAA Journal* 1998;36(5):703–11. doi:`10.2514/2.458`.

4. Jiao X. Volume and feature preservation in surface mesh optimization. In: *Proceedings of the 15th International Meshing Roundtable*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2006:359–73.

5. Scherer M, Denzer R, Steinmann P. On a solution strategy for energy-based mesh optimization in finite hyperelastostatics. *Computer Methods in Applied Mechanics and Engineering* 2008;197(6):609–22.

6. Clark B, Ray N, Jiao X. Surface mesh optimization, adaption, and untangling with high-order accuracy. In: *Proceedings of the 21st International Meshing Roundtable*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2013:385–402.

7. Zandsalimy M, Ollivier-Gooch C. Unsupervised residual vector analysis for mesh optimization. In: *AIAA SciTech 2023 Forum*. 2023:0833. doi:10.2514/6.2023-0833.

8. Zandsalimy M, 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* 2024;510:113063. doi:10.1016/j.jcp.2024.113063.

9. Zandsalimy M, Ollivier-Gooch C. Dynamic mode decomposition for improved numerical stability of finite volume simulations. In: *AIAA SciTech 2024 Forum*. 2024:1948. doi:10.2514/6.2024-1948.

10. Zandsalimy M, Ollivier-Gooch C. Approximate jacobian eigenanalysis for unstructured mesh optimization of finite volume simulations. In: *AIAA SciTech 2024 Forum*. 2024:1949. doi:10.2514/6.2024-1949.

11. Bejarano DAO, Ibargüen-Mondragón E, Gómez-Hernandez EA. A stability test for non linear systems of ordinary differential equations based on the Gershgorin circles. *Contemporary Engineering Sciences* 2018;11:4541–8.

12. Roe P. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics* 1981;43(2):357 –72. doi:10.1016/0021-9991(81)90128-5.

13. Michalak C, Ollivier-Gooch C. Globalized matrix-explicit Newton-GMRES for the high-order accurate solution of the Euler equations. *Computers & Fluids* 2010;39(7):1156 –67. doi:10.1016/j.compfluid.2010.02.008.

14. Saad Y. Iterative Methods for Sparse Linear Systems. Society for Industrial and Applied Mathematics; 2003. doi:10.1137/1.9780898718003.

15. Balay S, Abhyankar S, Adams MF, Benson S, Brown J, Brune P, Buschelman K, Constantinescu E, Dalcin L, Dener A, Eijkhout V, Faibussowitsch J, Gropp WD, Hapla V, Isaac T, Jolivet P, Karpeev D, Kaushik D, Knepley MG, Kong F, Kruger S, May DA, McInnes LC, Mills RT, Mitchell L, Munson T, Roman JE, Rupp K, Sanan P, Sarich J, Smith BF, Zampini S, Zhang H, Zhang H, Zhang J. PETSc/TAO users manual. Tech. Rep. ANL-21/39 - Revision 3.20; Argonne National Laboratory; 2023. doi:10.2172/1968587.

16. Bhatia N, Szegö G. Stability Theory of Dynamical Systems. Classics in Mathematics; Springer Berlin Heidelberg; 2002. ISBN 9783540427483.

17. Schmid PJ. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics* 2010;656:5–28. doi:10.1017/S0022112010001217.

18. Hernandez V, Roman JE, Vidal V. SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems. *ACM Trans Math Softw* 2005;31(3):351–362. doi:10.1145/1089014.1089019.

19. Stanković L. On the sparsity bound for the existence of a unique solution in compressive sensing by the Gershgorin theorem. *Signal Processing* 2022;190:108316. doi:10.1016/j.sigpro.2021.108316.

20. Bücker HM, Pollul B, Rasch A. On CFL evolution strategies for implicit upwind methods in linearized Euler equations. *International Journal for Numerical Methods in Fluids* 2009;59(1):1–18. doi:10.1002/fld.1798.

21. Lacy DS, Clark AM. 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:doi:10.2514/6.2020-2771.

22. Diskin B, Liu Y, Galbraith MC. High-Fidelity CFD Verification Workshop 2024: Spalart-Allmaras QCR2000-R Turbulence Model. 2023 AIAA SciTech Forum; American Institute of Aeronautics and Astronautics; 2023:doi:10.2514/6.2023-1244.