Dynamic Mode Decomposition For Improved Numerical Stability of Finite Volume Simulations

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

We propose a novel approach to mesh optimization for improved stability of finite-volume simulations using dynamic mode decomposition on a subset of the solution vectors. A minimal number of the most recent solution vectors in the simulation are selected for dynamic mode decomposition. The eigenvalues of the Koopman matrix depict the magnitude growth rate and oscillation frequency of the largest solution modes. The computational cost of this method depends on the number of solution vectors in use, which is considerably less expensive compared to the eigenanalysis of the full Jacobian matrix. The dynamic eigenvectors are utilized to identify which control volumes and vertices have the greatest impact on each dynamic solution mode. The gradients of the Jacobian matrix diagonal are calculated with respect to the movement of the selected vertices. The positions of these points are adjusted to increase the diagonal dominance of the Jacobian matrix on the corresponding rows. The results verify the effectiveness and feasibility of the novel approach in numerical stability improvement through unstructured mesh optimization. This state-of-the-art method addresses the challenges faced by the latest studies in this field with full automation of the mesh optimization process and substantial computational savings.

I. Introduction

Numerical stability is a crucial aspect of many Computational Fluid Dynamics (CFD) applications and has been studied substantially over the years. Nevertheless, proper stability is a major limiting factor in the size and resolution of today's numerical simulations. Mesh optimization is a general area of research with a focus on CFD stability improvement. Among these studies, the most recent work focuses on improving the linear stability of the Jacobian matrix through local modifications to the mesh. Inspired by this class of techniques, we propose a novel method for mesh optimization based on the Dynamic Mode Decomposition (DMD) of a subset of solution vectors in the simulation.

Zandsalimy and Ollivier-Gooch [1] presented a novel approach for mesh optimization through the linear stability analysis of the Jacobian matrix. In this method, the eigenvalue problem is solved on the Jacobian of the linearized dynamical system. According to the Lyapunov stability theory, the eigenvalues with positive real parts are the unstable solution modes. Zandsalimy and Ollivier-Gooch [1] used this idea to identify problematic local areas in the mesh and improved the stability of the linear solver through mesh modification. Zandsalimy and Ollivier-Gooch [2] utilized unsupervised anomaly detection models on the residual vector to identify the diverging solution modes by analyzing the outlier values. Synthetic vectors were constructed from the residual vector on the selected cells that resemble the unstable eigenvectors in the solution. Using such synthetic vectors proved to be a plausible approach to mesh optimization without performing the computationally expensive eigenanalysis.

In the present work, we focus on the mesh optimization of cell-centered unstructured mesh finite volume simulations through dynamic mode decomposition of the solution vectors. DMD of the solution vectors gives a better approximation of the dynamics of the unstable solution modes before they cause the solution to diverge. The dynamic modes extracted are the generalization of the global stability modes. These modes help explain the physical mechanisms in the data sequence and simplify complex large-scale problems into a dynamical system with substantially fewer degrees of freedom. Using DMD helps remove the need for large-scale eigenanalysis in the optimization application which results in substantial computational savings. In addition, DMD provides a more accurate approximation of the dynamic solution modes compared to the Singular Value Decomposition (SVD) of the solution vectors. Unstable solution modes can be identified before they become visible in the residual or solution vectors. This was a challenge faced by the work of Zandsalimy and Ollivier-Gooch [2], which relied on unstable solution modes growing large enough to be identifiable through outliers in the residual vector.

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

The next issue, after timely unstable mode identification, is finding the correct cells and vertices that have the largest effect on each selected unstable mode. Numerical and physical solution modes can be distinguished by finding eigenvalue gradients with respect to mesh movement. The numerical modes usually have large gradients with respect to local changes in the mesh and have a small number of non-zero values which means they are pointing to a local region in the mesh. In the case of numerically unstable modes, DMD eigenvectors behave similarly. This enables us to identify the correct cell on the mesh for modification.

Herein, we propose a novel method to find effective directions for vertex modification for improved stability of the solution. In this method, the gradients of the diagonal of the Jacobian matrix on the corresponding rows are calculated with respect to mesh movement. The vertex locations are then modified for larger diagonal values on the rows of interest. According to the Gershgorin circle theorem, increasing the diagonal dominance of the Jacobian matrix is a step in the right direction for stability improvement of the linear system as the Gershgorin circles are being pushed to the left side of the eigenspectrum. After mesh modification, the user can decide whether to continue the simulation or restart for a stable solution. The results presented herein depict improved numerical stability behavior of initially unstable solutions and slow converging simulations.

II. Background

A. Flow Solver

The conservation of a dependent variable U can be written as a function of time and a vector of independent variables $\vec{x}$, which in this context are the mesh point locations.

$$\frac{\partial U}{\partial t}(t,\vec{x}) + \vec{\nabla}\cdot\vec{F}(U) = f(t,\vec{x}) \tag{1}$$

In this equation, $\vec{F}$ and f are the flux vector and the source term, respectively. The differential equations are written in divergence form and integrated over control volumes. Applying the finite volume method [3–6] to this equation we arrive at the following.

$$\frac{d\bar{U}_i}{dt} = -\frac{1}{|V_i|}\oint_{\partial V_i}\left(\vec{F}\cdot\vec{n}\right)dA + \frac{1}{|V_i|}\int_{V_i} f\ dV = R(\bar{U}) \tag{2}$$

Here, V_i refers to the cell i with a volume of $|V_i|$. $\vec{n}$ is the outward pointing unit normal vector at the cell boundary, R is the residual, and $\bar{U}_i$ is the average conservative property inside the cell,

$$\bar{U}_i = \frac{1}{|V_i|}\int_{V_i} U dV \tag{3}$$

The following approach is used to calculate the flux integral in Equation 2 with second-order accuracy:

1) Reconstruct a piece-wise linear solution approximation from the piece-wise constant control volume averages using the linear least-squares method [7].
2) Compute the flux at each quadrature point on the cell's boundaries. We utilize Roe's scheme [8] for inviscid flux calculation.
3) Integrate flux values on Gauss quadrature points.

The boundary conditions are applied weakly using flux values on the boundaries. The Crank-Nicolson time advance scheme is utilized for second-order time integration as presented in Equation 4 (the bar notation is dropped for convenience).

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

In this equation, $\vec{U} = \{U_1, U_2, \ldots, U_k\}$ is the vector of control volume averages and $\vec{R}$ is the vector of control volume residuals. The linearized form of Equation 4 is

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

B. Baseline Mesh Optimization Approach

The presented methodology herein is an improvement of the stabilization algorithm presented by Zandsalimy and Ollivier-Gooch [1]. This algorithm was based on the Lyapunov stability theory [9] for dynamical systems without explicit integration. The Lyapunov theorem of stability implies that the linear time-invariant system $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$ is locally stable if all the eigenvalues of $\boldsymbol{A}$ have non-positive real parts. The real part of an eigenvalue depicts the magnitude growth rate and the imaginary part shows the frequency of evolution of the state variable in the direction of the corresponding eigenvector [10]. Figure 1 shows the dynamic response of the linear system characterized by the eigenspectrum. As seen here, the eigenmodes with larger real parts are less stable while the eigenmodes with larger imaginary parts have a higher evolution frequency. It can also be seen that the eigenmodes with positive real parts have infinite growth in magnitude with time while the ones with negative real parts are converging.

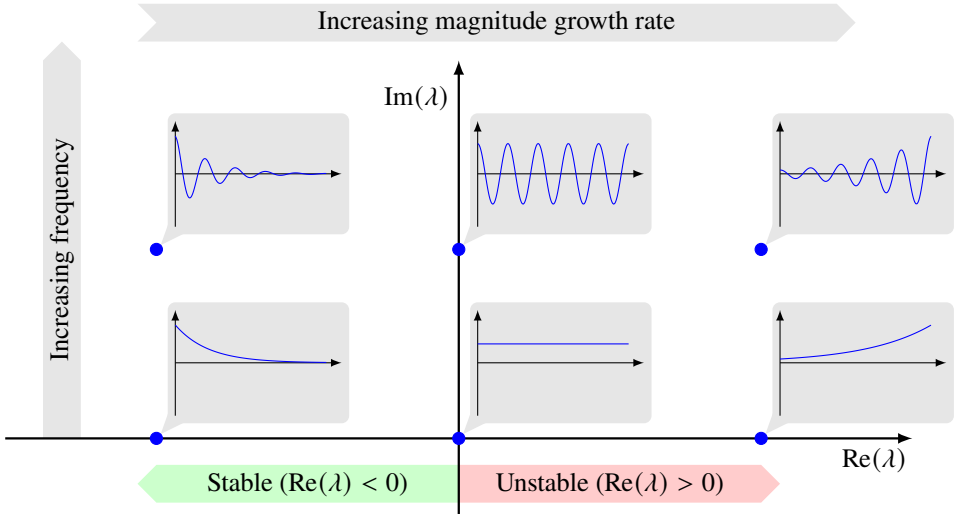


Fig. 1 The dynamic response of the linear system characterized by the eigenspectrum.

The Jacobian and therefore its eigenspectrum is dependent on several factors including the physics, discretization method, mesh topology, and vertex locations. This implies that vertex movement can be employed to move unstable eigenmodes as a means of stability improvement in CFD [1]. In this approach, the first step of a successful mesh optimization for stability is to perform a partial eigenanalysis of the Jacobian matrix. After identifying the unstable values in the right open half of the spectrum the corresponding eigenvectors are calculated and used to select specific vertices for modification. Using this method, the modification of a single vertex is typically sufficient to stabilize each unstable eigenmode. Then, the gradients of the unstable modes with respect to the movement of the selected vertices can be calculated and used to modify vertex location to stabilize the unstable eigenmodes using the steepest descent method. The reader is referred to Zandsalimy and Ollivier-Gooch [1] for the details of their mesh optimization approach.

A schematic of the optimization algorithm in conjunction with the non-linear solver is depicted in Figure 2. As seen in the *Optimization* block, the first step is deciding to perform the stabilization in a certain iteration of the solver. This step can be automated through the novel approach presented in the current study. As the solution progresses, the eigenvalues of the semi-discrete Jacobian change, and new unstable modes might appear in the solution. To remediate these cases, Zandsalimy and Ollivier-Gooch [1] applied their approach at one or more intermediate stages of convergence as needed. This method is capable of stabilizing initially unstable finite volume solutions on unstructured meshes as well as solutions that exhibit unstable behavior after several iterations of the solver.

This procedure has two main practical defects. First, eigenanalysis to find the rightmost eigenvalues is very expensive for problems of even moderate size. Second, this high cost makes it infeasible to apply the method at every iteration of the nonlinear solver, which means that some approach for automating the process is required. In previous work [2, 11], we have proposed one possible solution for these issues. This paper proposes a different solution, based on dynamic mode decomposition. This new approach for identifying unstable modes is much more efficient than eigenanalysis, and so can be applied much more frequently.

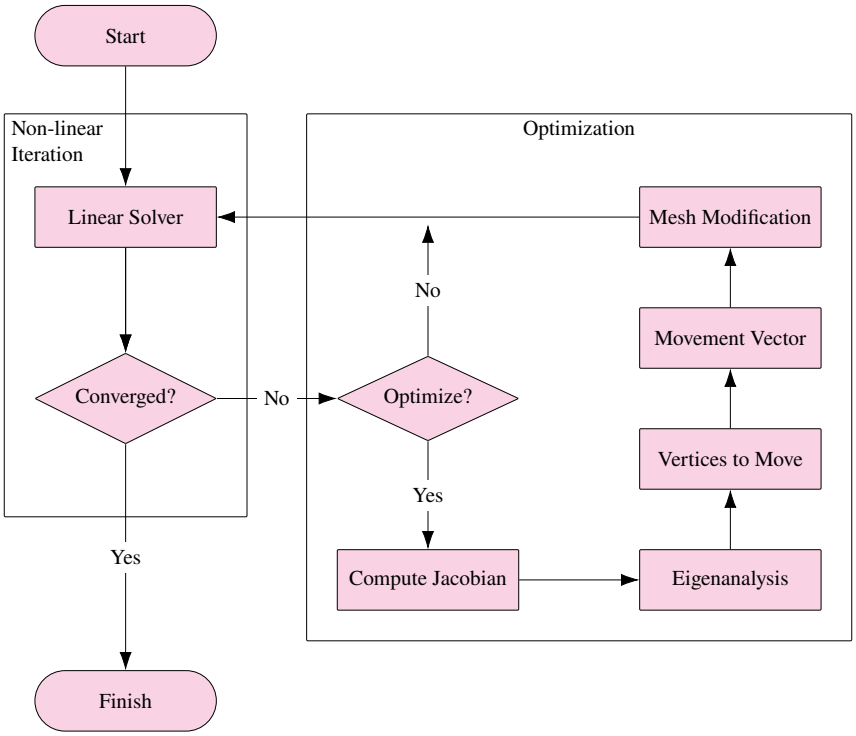


Fig. 2 Overview of the stabilization approach presented by Zandsalimy and Ollivier-Gooch [1] coupled with the non-linear solver

C. Dynamic Mode Decomposition

Eigenanalysis of the Jacobian matrix is a form of decomposition of the flow field in complex solution domains into modal structures. The Arnoldi algorithm and its variations are often utilized in which the Jacobian is reduced by successive orthogonalization and projections onto an equivalent matrix of smaller size whose eigenvalues approximate some of the eigenvalues of the original system. This method requires access to the full Jacobian matrix or its product with a given vector. This restriction plus the computational requirements has limited the application of eigenanalysis in stability improvement and mesh optimization applications.

In the present study, we utilize a solution decomposition method known as dynamic mode decomposition. This method extracts the time dynamics of the system from the solution vectors without having direct access to the underlying model [12]. Like Principal Component Analysis (PCA), DMD provides information about the coherent spatial structures in the data, while also giving information about their evolution behavior over time. This makes DMD a powerful tool for identifying unstable solution modes in the numerical simulation long before they have a chance to grow out of control and result in numerical divergence. The dynamic modes from DMD provide the temporal growth rates and oscillation frequencies of the simulation without ever forming the Jacobian matrix.

This data-driven dimensionality reduction technique computes a set of dynamic modes given a multivariate time series dataset. Multivariate time series forecasting is another important facet of the approach. The process of DMD starts with solution snapshots in the fluid flow. In the simplest version, which we describe here, these snapshots are equally spaced in time. We form two matrices $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ out of the n most recent solution update vectors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ as follows.

$$\boldsymbol{X}_1 = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_{n-1}] \in \mathbb{R}^{m \times n-1} \tag{6}$$

$$\boldsymbol{X}_2 = [\boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{m \times n-1} \tag{7}$$

DMD aims to find a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times m}$ that relates $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ as follows.

$$\boldsymbol{X}_2 = \boldsymbol{A}\boldsymbol{X}_1 \tag{8}$$

where $\boldsymbol{A}$ can be expressed as $\boldsymbol{X}_2\boldsymbol{X}_1^{\dagger}$ in which $\boldsymbol{X}_1^{\dagger}$ is the Moore-–Penrose inverse of $\boldsymbol{X}_1$. Here, $\boldsymbol{A}$ does not have to be explicitly computed as only a low-ranked matrix to approximate the most important coefficients (called the Koopman

matrix) is sufficient. The Koopman matrix $\boldsymbol{K} \in \mathbb{R}^{n-1 \times n-1}$ is a mapping of the matrices $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ as follows.

$$\boldsymbol{X}_2 = \boldsymbol{X}_1 \boldsymbol{K} \tag{9}$$

Singular Value Decomposition (SVD) of $\boldsymbol{X}_1$ is utilized to find the low-ranked representation of $\boldsymbol{A}$.

$$\boldsymbol{X}_1 = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^H \tag{10}$$

In this equation, $\boldsymbol{U}$ and $\boldsymbol{V}$ contain the left and right singular vectors and $\boldsymbol{\Sigma}$ is a diagonal matrix consisting of the singular values. In the next step, the Koopman matrix $\boldsymbol{K}$ is found as follows.

$$\boldsymbol{K} = \boldsymbol{U}^H \boldsymbol{X}_2 \boldsymbol{V} \boldsymbol{\Sigma}^{-1} \tag{11}$$

Computing the dynamic modes is as simple as finding the eigenvalues of the small matrix $\boldsymbol{K}$.

$$\boldsymbol{K} = \boldsymbol{Q} \boldsymbol{\Phi} \boldsymbol{Q}^H \tag{12}$$

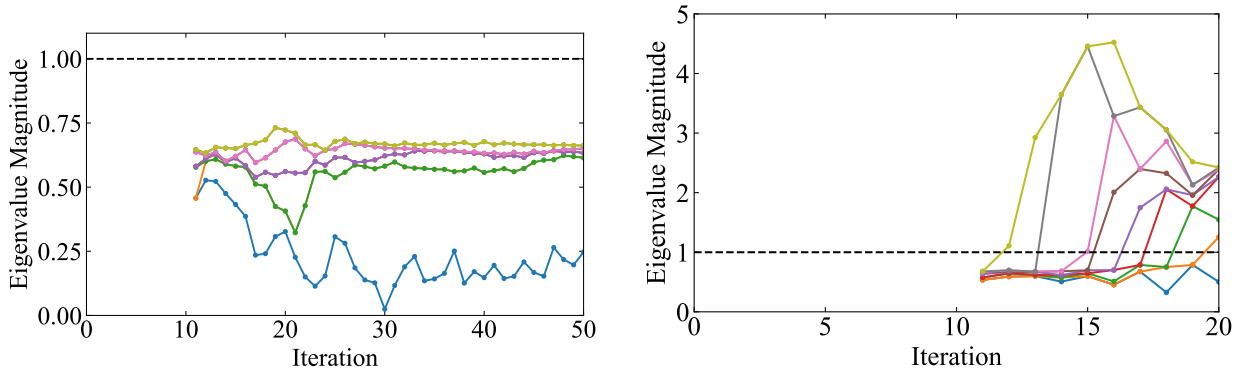
in which, $\boldsymbol{Q}$ and $\boldsymbol{\Phi}$ contain the eigenvectors and eigenvalues, respectively. The eigenvalues of $\boldsymbol{A}$ and $\boldsymbol{K}$ are the same. The corresponding DMD modes, or eigenvectors of the high-dimensional system, are computed from the projected eigenvectors $\boldsymbol{Q}$ as [13],

$$\boldsymbol{\Psi} = \boldsymbol{X}_2 \boldsymbol{V} \boldsymbol{\Sigma}^{-1} \boldsymbol{Q} \tag{13}$$

where the eigenvectors of the high-dimensional system $\boldsymbol{A}$ are the columns of $\boldsymbol{\Psi}$. As a result, it is possible to find the eigenvectors of the high-dimensional space at low computational cost (without explicit formation of the high-dimensional system). These are required in the next steps of the presented mesh optimization approach.

An important aspect of using DMD of solution vectors instead of Jacobian eigenanalysis is the possibility of large computational savings in the process. The most resource-intensive aspect of the approach presented by Zandsalimy and Ollivier-Gooch [1] is the solution to the large sparse eigenvalue problem. The computational cost of this module can be up to $O(m^3)$ for a problem with m degrees of freedom which makes the approach infeasible for large-scale industrial simulations. The present study, on the other hand, aims to use only a small number n of the solution vectors ($n \ll m$) in the process. In this case, DMD has a computational complexity of $O(mn^2)$ which results in substantial computational savings compared to the eigenanalysis of the Jacobian matrix.

DMD is performed on the latest 10 solution vectors in two example problems and the magnitude of the eigenvalues of $\boldsymbol{K}$ is shown in Figure 3. In these figures, each color depicts the evolution of a different eigenvalue with solution iteration. Figure 3a shows the results for a Burgers problem solved using the Crank-Nicolson time-stepping method which is a stable simulation and converges to 10^{-10} in 40 iterations. As seen here, the magnitudes of DMD eigenvalues stay smaller than 1.0 during the simulation. This means that all the dominant solution modes are converging to zero. Figure 3b refers to a different Burgers problem solved using the Crank-Nicolson time-integration method which is unstable and diverges at iteration 20 of the solver. Here, we can see some of the solution modes show a magnitude larger than 1.0 and continue to grow during the solution iteration. In fact, these solution modes cause the numerical simulation to blow up entirely.



(a) Stable Solution **(b) Unstable Solution**

Fig. 3 Eigenvalue magnitude of the Koopman matrix for two example Burgers problems

III. Methodology

The present section will discuss the novel algorithm of mesh optimization for the numerical stability improvement of finite-volume simulations. Each numerical module in the method will be discussed in detail.

A. Solution Mode Identification

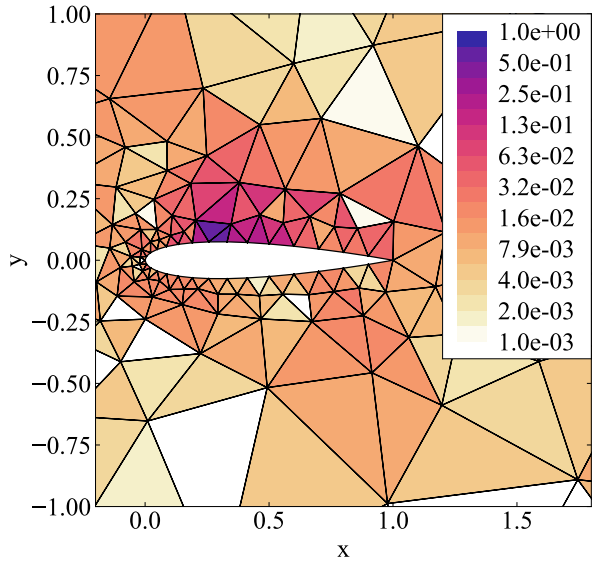
We start by selecting a number of the most recent successive solution vectors in the simulation for the application of DMD. The number of solution vectors is a user-specific decision and one may choose as many as it is computationally feasible on the available resources in their disposal. However, according to our experiments even for solutions with a large number of degrees of freedom only a handful of the most recent solution vectors is enough to identify the dominant solution modes which usually include the unstable ones. In some examples, we stabilize solutions with 30000 degrees of freedom with the use of only 10 solution vectors.

DMD is performed on the latest selection of solution vectors at every iteration of the solver. The eigenvalues of the Koopman matrix can then be calculated and analyzed for unstable solution mode identification. The eigenvalues with a magnitude larger than 1.0 can be considered as unstable modes. As discussed previously, DMD is computationally much cheaper than eigenanalysis which means it is feasible to apply at every iteration of the solution without significant resource requirements. Further, it is possible to be more conservative in the eigenvalue selection process (magnitudes smaller than 1.0) for optimization. As we will show in section IV, this approach can improve the convergence rate of initially stable simulations. As a result, the presented methodology can both be used to stabilize unstable finite-volume simulations at low computational cost and to improve the convergence rate of initially stable simulations dramatically.

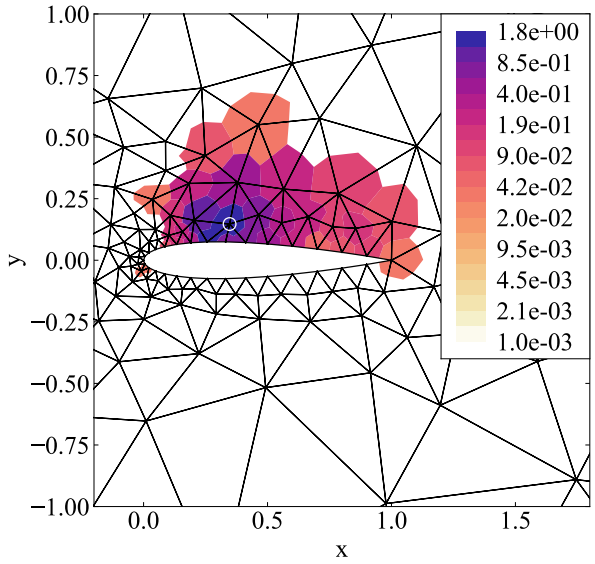
B. Control Volume and Vertex Selection

The next step of the optimization is the selection of the correct cells and vertices in the mesh for modification that would result in an optimized simulation. It should be noted that modification to these vertices and cells should have the highest possible effect on the solution modes in question. Zandsalimy and Ollivier-Gooch [1] presented a computationally efficient method for an effective selection. In this method, instead of finding the eigenvalue gradients directly (which is computationally expensive) the eigenvectors of the Jacobian matrix are used as a proxy for selection. We know that the unstable eigenvectors point directly to the problematic cells in the solution which have the largest effect on the unstable solution modes. The non-zero entries in the normalized eigenvector are laid out on the mesh and for each vertex, the absolute values of the eigenvector in the adjacent cells are added as a selection weight. Finally, the vertex with the largest selection weight is selected for optimization. This has proved as a plausible approach to vertex selection instead of calculating the gradient of the eigenvalues directly.

We adopt the same approach with the only difference being in the type of vector used. Herein, the DMD eigenvectors of the matrix $\boldsymbol{A}$ ($\boldsymbol{\Psi}$) are used in the process of vertex selection. These eigenvectors also point to the local areas in the mesh with stability issues. As a result, it is possible to select the correct cell for mesh modification that would have the largest possible effect on the corresponding solution mode. To demonstrate this approach, the absolute values of the DMD eigenvector corresponding to an unstable mode in an Euler problem are plotted on the mesh as shown in Figure 4a. The next step is to use the summation of the vector value in each cell on its vertices as a measure of selection weight. Figure 4b shows the distribution of the weight measure and the vertex that is selected as a candidate for mesh optimization indicated with a white circle. As depicted, the selected vector is highly local to a certain area in the mesh which echos the findings of [1] about the unstable right eigenvectors. Further, note that in the selection weight calculation of Figure 4b only the cells that have a non-negligible DMD eigenvector (greater than 5% of the largest entry) are considered.



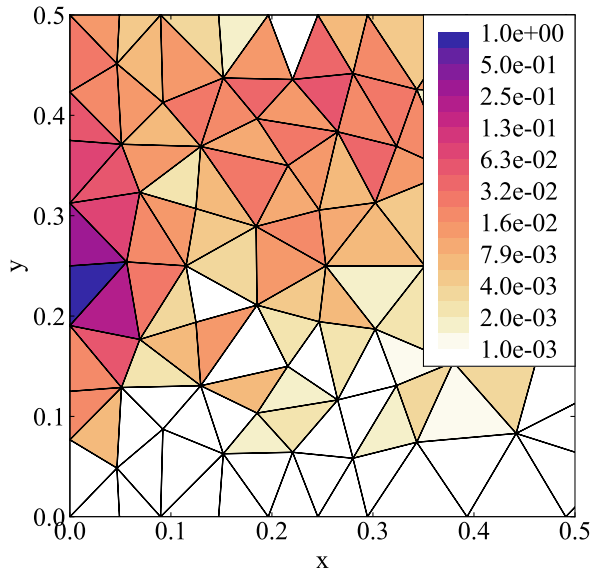
(a) The absolute value of the unstable DMD eigenvector components



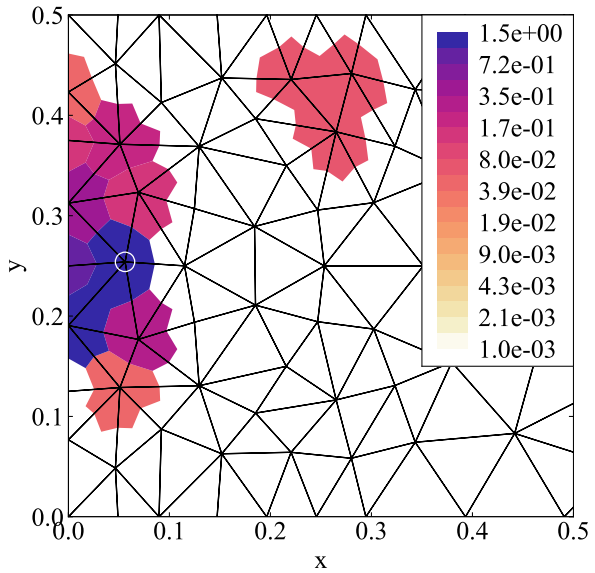
(b) The summation of weights for each vertex

Fig. 4 The vertex selection procedure in an Euler problem (note the colormap logarithmic scale)

The next experiment is performed on a Burgers problem in a rectangular channel. The absolute values of the unstable DMD eigenvector are depicted in Figure 5a. As seen, the selected vector is highly local to a certain area in the mesh which indicates the problematic cells. Figure 5b shows the selection weight measure on the vertices at the same iteration of the problem. In this case, a single vertex is selected for modification through the presented methodology which is indicated with a white circle. Once again, note that in the selection weight calculation of Figure 5b only the cells that have a non-negligible DMD eigenvector (greater than 5% of the largest entry) are considered.



(a) The absolute value of the unstable DMD eigenvector components



(b) The summation of weights for each vertex

Fig. 5 The vertex selection procedure in a Burgers problem (note the colormap logarithmic scale)

C. Movement Vector Calculation

Previous research has focused on computing the gradients of eigenvalues in relation to mesh movement to determine suitable modification vectors for the selected vertices. In contrast, we propose a novel approach to this issue by computing the gradient of the Jacobian matrix directly. The Gershgorin circle theorem [14], associates a disc to each row of the matrix. The diagonal entries of the Jacobian matrix indicate the center of each disc, and the sum of absolute values of the off-diagonal entries shows the radius. The union of these circles will contain all the eigenvalues of the Jacobian. We aim to increase the diagonal dominance of the Jacobian rows indicated as problematic in the previous steps of the optimization. This approach does not guarantee that the corresponding Gershgorin circle will be completely pushed to the stable side of the eigenspectrum. Nonetheless, it is a significant step in the right direction and the possible computational savings makes it a plausible approach for vertex movement calculation. An obvious virtue of this approach is the complete elimination of the eigenanalysis while still not having to take refuge in the less numerically dependable approaches of machine learning.

Another aspect of this approach lies in its versatility beyond mesh modification for stability enhancement. Instead of solely focusing on adjusting vertex positions, we can achieve similar effectiveness by considering other factors in the numerical simulation that influence the Jacobian matrix. By computing the gradients of the diagonal elements of the Jacobian matrix with respect to variations in the chosen factor, such as the reconstruction stencil size and topology, time-integration scheme, and time step size, we can attain comparable results. Moreover, we only need to determine the gradient of the Jacobian diagonal for the designated cells, thereby circumventing the need to construct the Jacobian matrix entirely. This is particularly advantageous when dealing with computationally intensive large-scale problems, as avoiding the formation of the Jacobian matrix can alleviate resource constraints. To approximate the derivative of the Jacobian matrix with respect to mesh movement, denoted as $\frac{d\boldsymbol{J}}{d\boldsymbol{\zeta}}$, we can readily employ the finite difference method.

$$\frac{d\boldsymbol{J}}{d\boldsymbol{\zeta}} \approx \frac{\partial \boldsymbol{J}}{\partial \boldsymbol{\zeta}} = \frac{\boldsymbol{J}(\boldsymbol{\zeta} + \delta\boldsymbol{\zeta}) - \boldsymbol{J}(\boldsymbol{\zeta})}{\delta\boldsymbol{\zeta}} \tag{14}$$

The only optimization step remaining at this point is vertex modification. Considering the gradient vector and the permissible range of vertex movement, we strategically adjust the position of the vertex in a manner that maximizes the diagonal dominance. These vertex movement limitations are essential to prevent mesh tangling and the emergence of degenerate cells.

D. Algorithm

The modules discussed previously are assembled into a novel stability improvement approach. The architecture is depicted in Figure 6 in conjunction with the non-linear solver. This approach eliminates the need for human intervention in the process, as well as the calculation of eigenvalues of the Jacobian of the PDE discretization. At each non-linear iteration, DMD is performed on a collection of solution update vectors to find the dominant solution mode. The magnitude of each DMD eigenvalue gives an indication of unstable versus stable modes. Going forward in the optimization, the DMD eigenvectors of the matrix $\boldsymbol{A}$ are constructed. These vectors are then utilized to find proper vertices for effective mesh modification. The vertex movement vector is calculated to increase the diagonal dominance of the Jacobian of the space discretization. This approach brings us one step closer to a more feasible stability improvement software that can be readily incorporated into current CFD solvers.

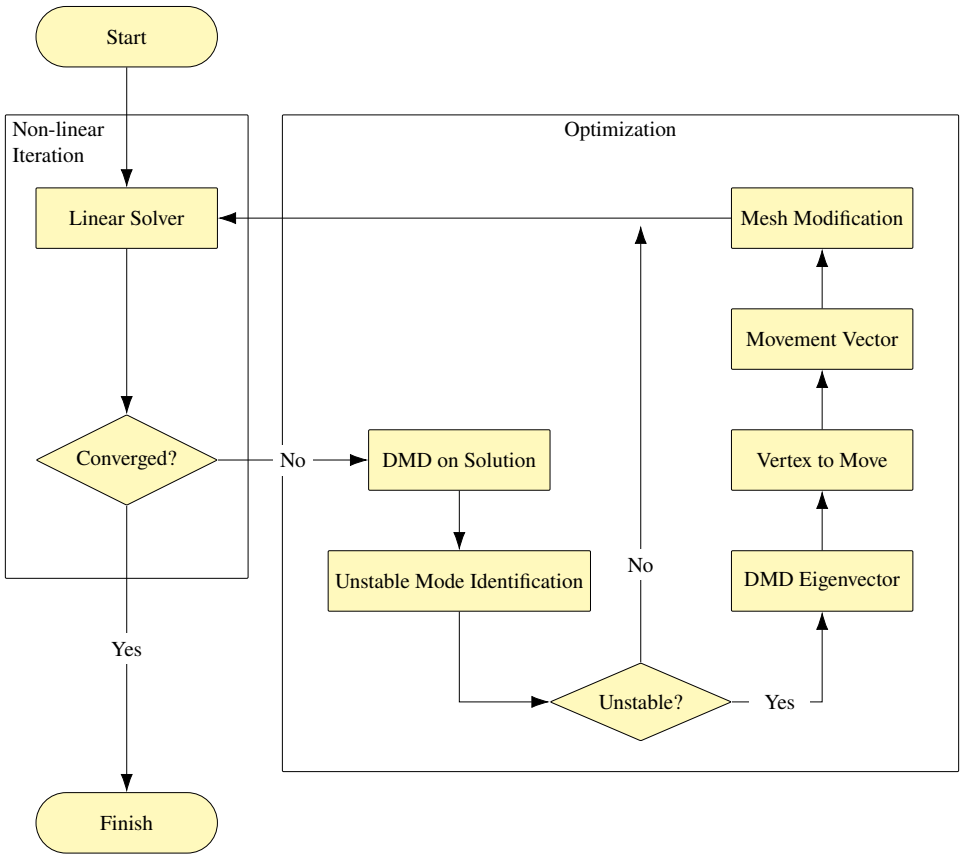


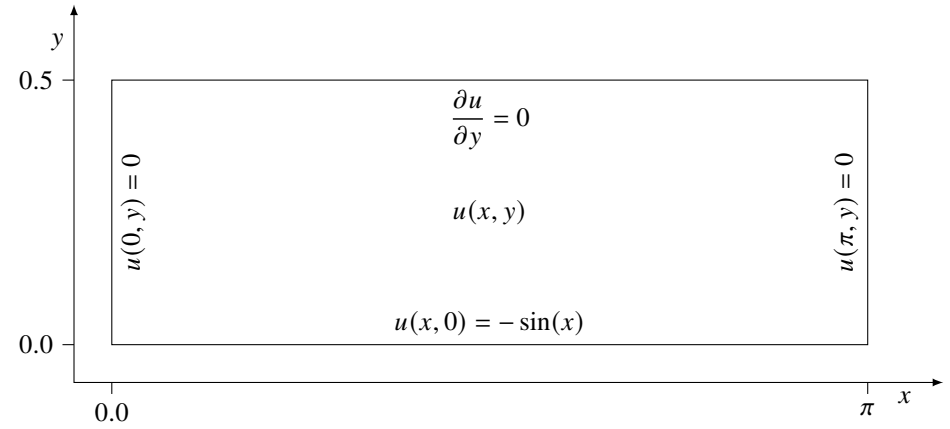
Fig. 6 Overview of the novel stabilization approach

IV. Results

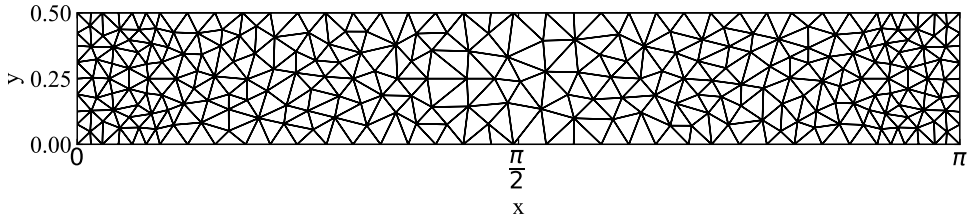
This section demonstrates the preliminary results obtained from the proposed mesh optimization application. Different finite volume simulations are selected and solved on various unstructured meshes to depict the capabilities of the proposed method.

A. 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 unstructured 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 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].

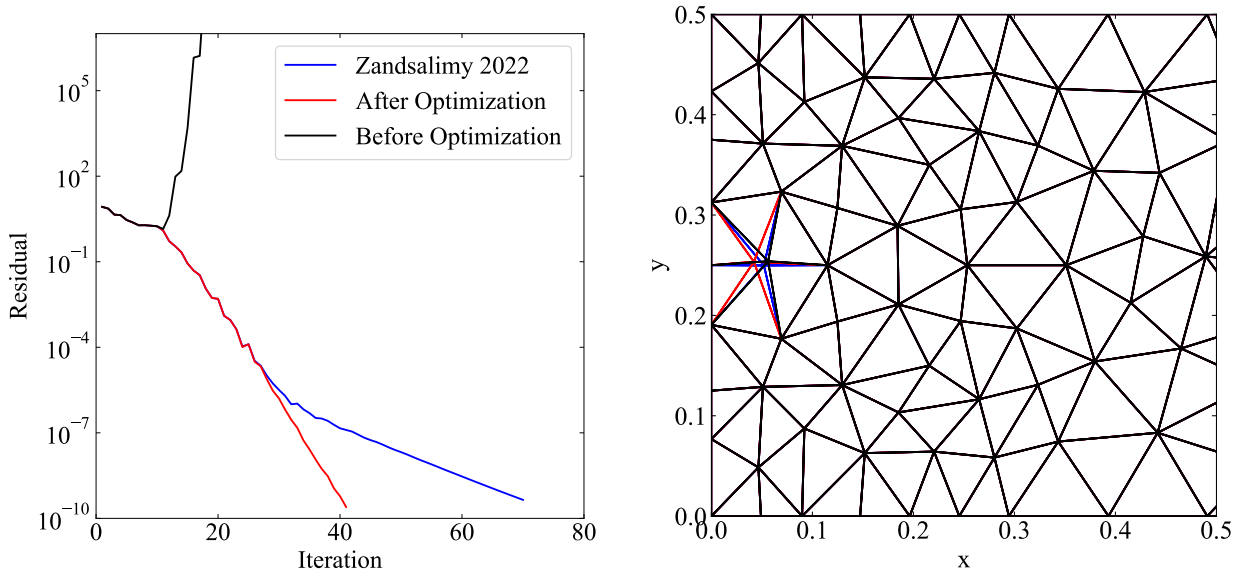


(a) The physical domain and boundary conditions



(b) An example of the unstructured mesh

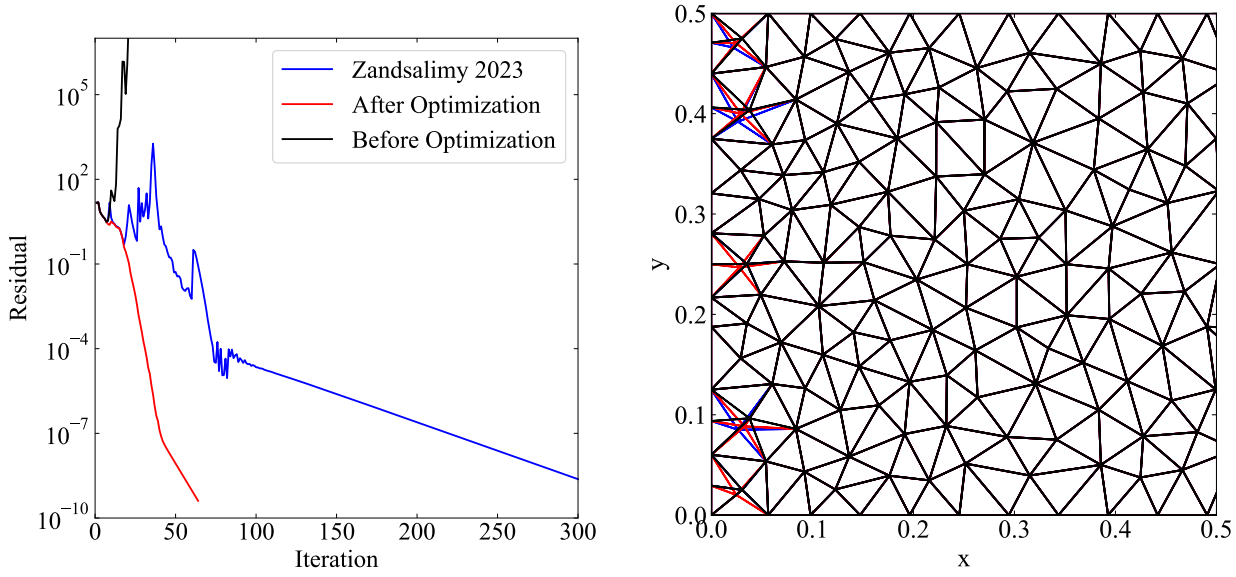
Fig. 7 The domain, boundary conditions, and unstructured mesh for the solution to the Burgers problem [1]



(a) The residual history **(b) Mesh modification**

Fig. 8 Mesh optimization in a Burgers problem on an unstructured mesh with 500 cells

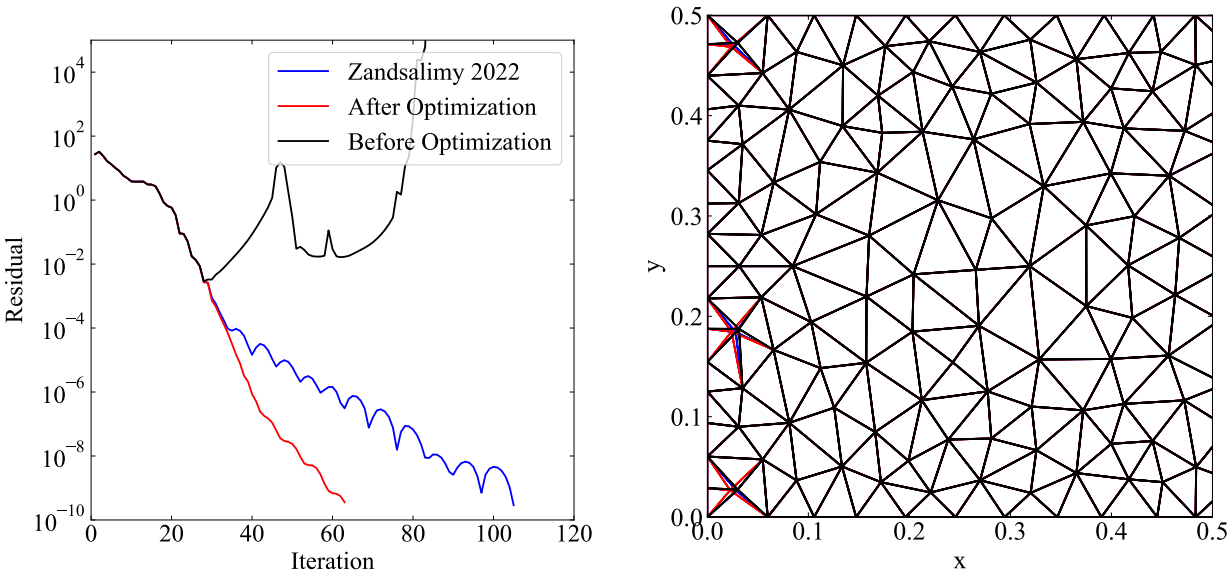
The next solution to the Burgers problem is performed using the Crank-Nicolson time-stepping method on a mesh with 1100 control volumes. This problem contains multiple unstable modes which required multiple iterations of the optimization approach for full stability. Unstable DMD eigenvalues are automatically 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 [2].



(a) The residual history **(b) Mesh modification**

Fig. 9 Mesh optimization in a Burgers problem on an unstructured mesh with 1100 cells

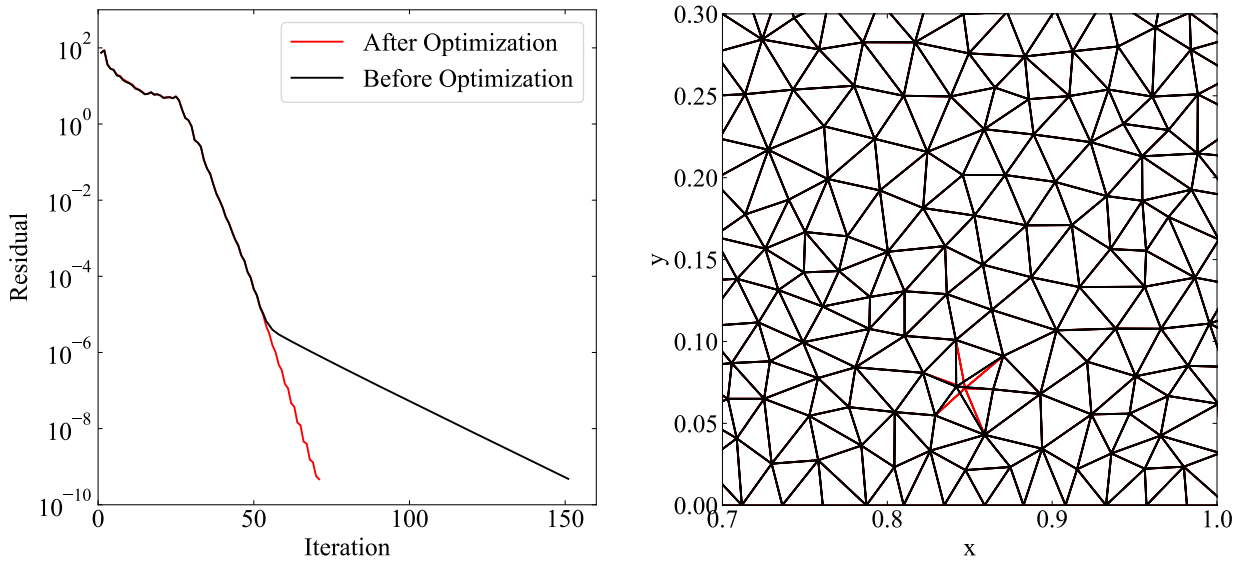
Another solution to the Burgers problem is performed using the Crank-Nicolson time-stepping method on a mesh with 1400 control volumes. This problem also contains multiple unstable modes requiring multiple optimization iterations. Unstable DMD eigenvalues are automatically detected at iterations 29, 34, and 40. In this case, the last selected mode is stable with a magnitude larger than 0.99. This mode is also identified for modification as it will cause a slow converging solution if unresolved. These three optimization iterations result in the modification of three vertices on the mesh for full stability. The residual history before and after optimization is presented in Figure 10a. The original and optimized meshes are presented in Figure 10b which shows the locations of three different vertices that are 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].



(a) The residual history **(b) Mesh modification**

Fig. 10 Mesh optimization in a Burgers problem on an unstructured mesh with 1400 cells

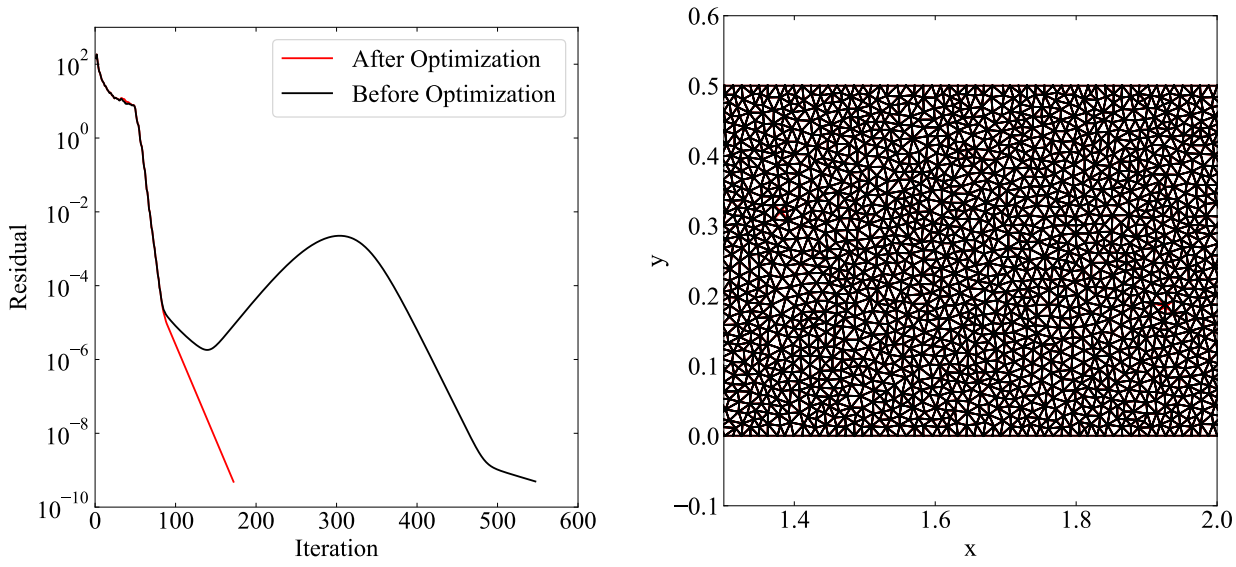
The next solution to the Burgers problem is performed using the Crank-Nicolson time-stepping method on a mesh with 4500 control volumes. This problem is originally stable and no unstable DMD modes are detected during solution iteration. However, there is a slow converging mode in the solution which has a magnitude larger than 0.9. Selecting this solution mode for the optimization process at iteration 45 of the solver results in a faster-converging simulation. The residual history is presented in Figure 11a that depicts convergence in half as many iterations compared to before the optimization. The original and optimized meshes are presented in Figure 11b which shows the modification of a single vertex.



(a) The residual history **(b) Mesh modification**

Fig. 11 Mesh optimization in a Burgers problem on an unstructured mesh with 4500 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 12a which depicts convergence in around $\frac{1}{3}$ of the solution iterations. The locations of 4 vertices are modified in this case, 2 of which are indicated in Figure 12b. 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.

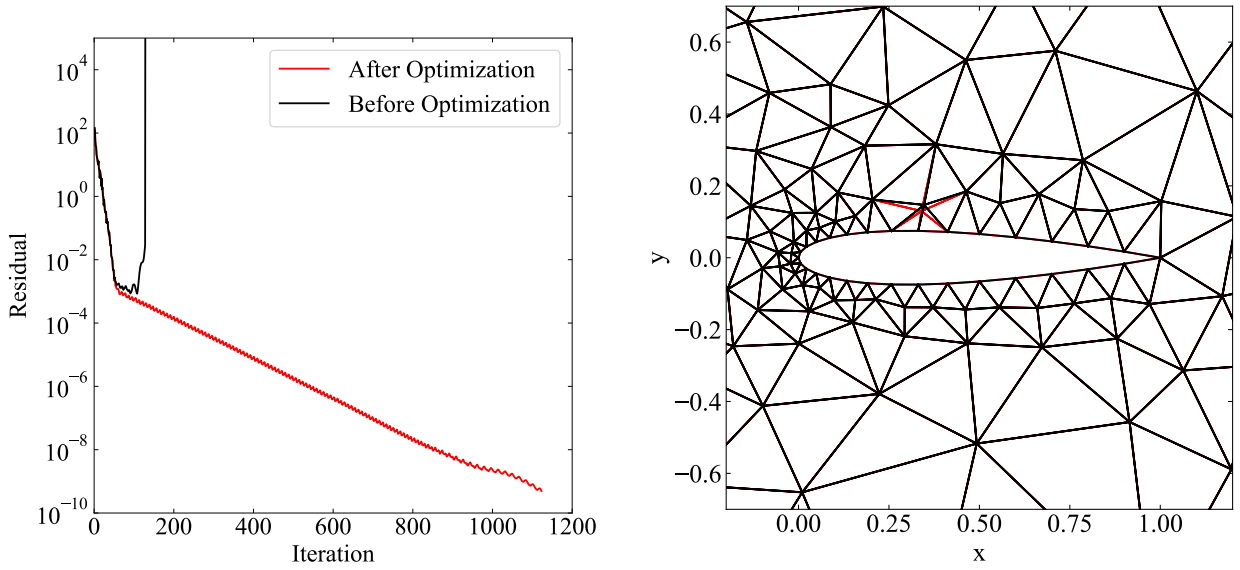


(a) The residual history **(b) Mesh modification**

Fig. 12 Mesh optimization in a Burgers problem using Implicit Euler time integration on an unstructured mesh with 17500 cells

B. Euler Problem

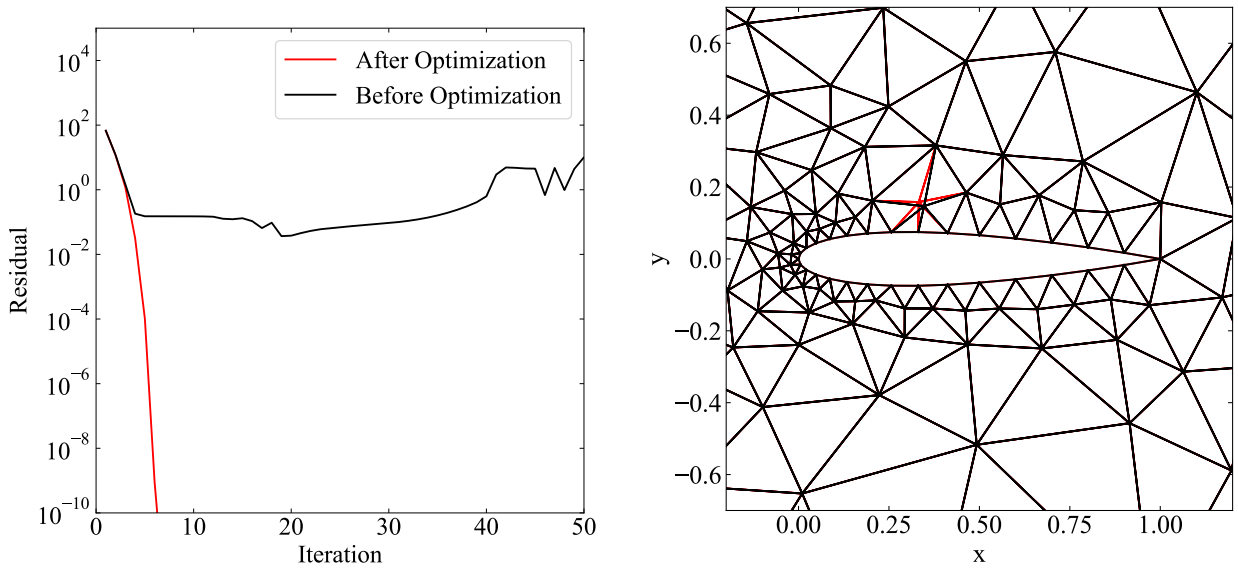
The Euler problem is selected as the next test case in the present study which is solved around the NACA 0015 airfoil inside a circular domain with a radius of 500 chords. In the initial solution for this problem, density is set to 1.0, velocity in (x, y) direction to $M_\infty(\cos(\alpha), \sin(\alpha))$, and pressure to P_{isen}. In these relations, M_∞ is the free-stream Mach number, α is the angle of attack, and P_{isen} is the resulting pressure during isentropic expansion to M_∞. 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 $M_\infty = 0.5$ 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**

Fig. 13 Mesh optimization in an Euler problem using Crank-Nicolson time integration on an unstructured mesh with 600 cells

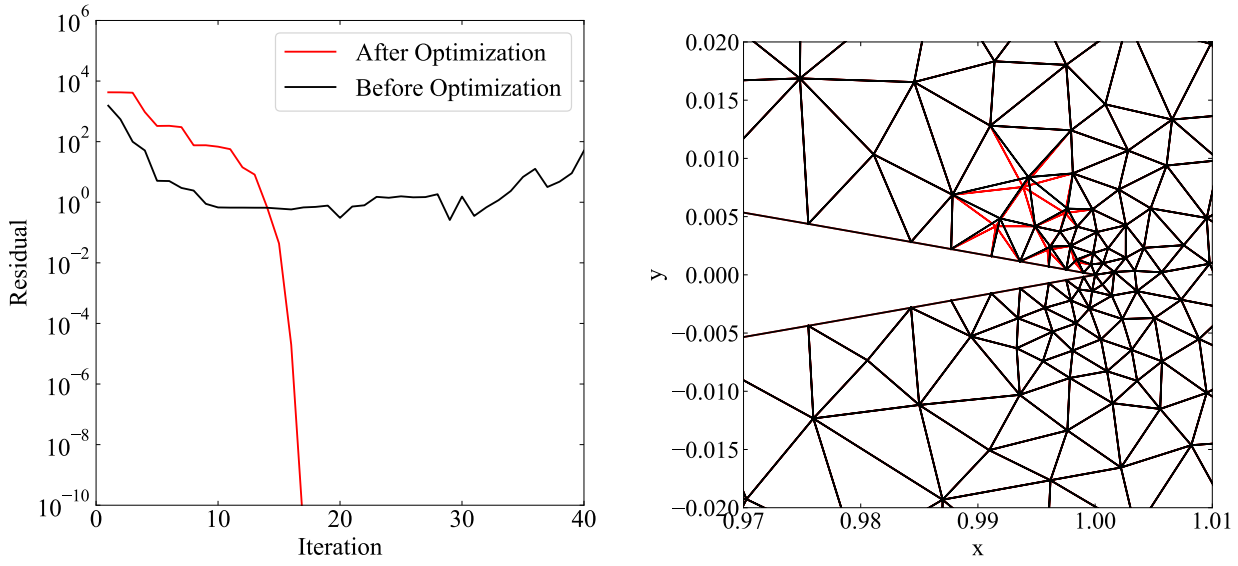
This problem is solved on the same mesh using the Implicit Euler time-stepping method. CFL evolution strategies [15] are utilized to increase the convergence rate of the problem. In this case, unstable DMD modes are detected at iteration 13 of the solver. Performing the optimization at this point results 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**

Fig. 14 Mesh optimization in an Euler problem using Implicit Euler time integration on an unstructured mesh with 600 cells

Another Euler problem is solved using the Implicit Euler time-stepping method on a mesh with 2500 cells. Unstable DMD modes are detected at iterations 11, 18, and 16 (the solution might depict new unstable modes on restart which is the case here) of the solver. Performing the optimization at these iterations results in the residual history presented in Figure 15a. The optimized mesh is presented in Figure 15b depicting the modification of 5 vertices near the trailing edge of the airfoil.

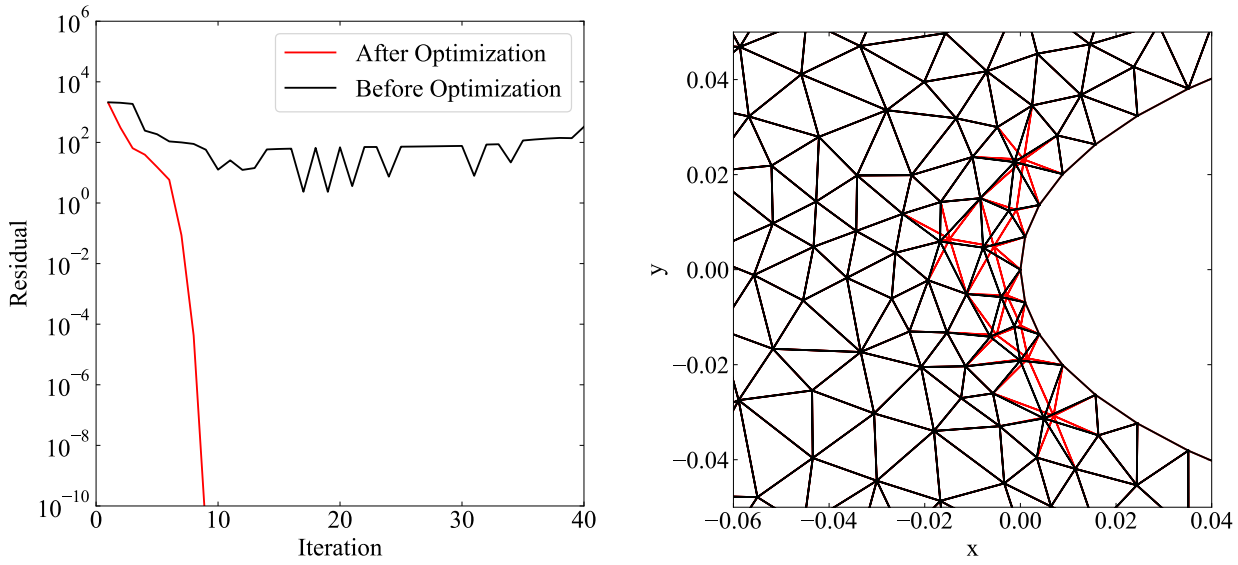


(a) The residual history **(b) Mesh modification**

Fig. 15 Mesh optimization in an Euler problem using Implicit Euler time integration on an unstructured mesh with 2500 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 16a. The optimized mesh is presented in Figure 16b 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**

Fig. 16 Mesh optimization in an Euler problem using Implicit Euler time integration on an unstructured mesh with 7600 cells

C. Computational Cost

This section evaluates and discusses the computational cost of the algorithm presented in this study. One major drawback of previous studies is the high computational complexity of their respective approaches. In this work, our goal is to reduce the runtime of the mesh optimization algorithm by either improving the complexity of the individual modules or eliminating the bottleneck modules. Specifically, we have successfully eliminated the eigenanalysis module from the approach proposed by Zandsalimy and Ollivier-Gooch [1]. Instead, we have adopted dynamic mode decomposition, which offers substantially lower computational complexity and exhibits a smaller increment rate as the mesh size increases. This improvement is represented by the solid blue line in Figure 17.

As a result of these enhancements, the overall optimization process in our work is faster compared to Zandsalimy and Ollivier-Gooch [1] approach, mainly due to the complete elimination of the eigenanalysis module. Furthermore, the overall runtime in our study is similar to that of Zandsalimy and Ollivier-Gooch [2]. However, it is worth noting that the DMD Analysis module in our work exhibits a smaller growth rate with increasing degrees of freedom compared to the Residual Analysis module described in Zandsalimy and Ollivier-Gooch [2]. On the other hand, the Movement Vector, Flow Solver, and Jacobian calculation modules show high similarities across all three papers.

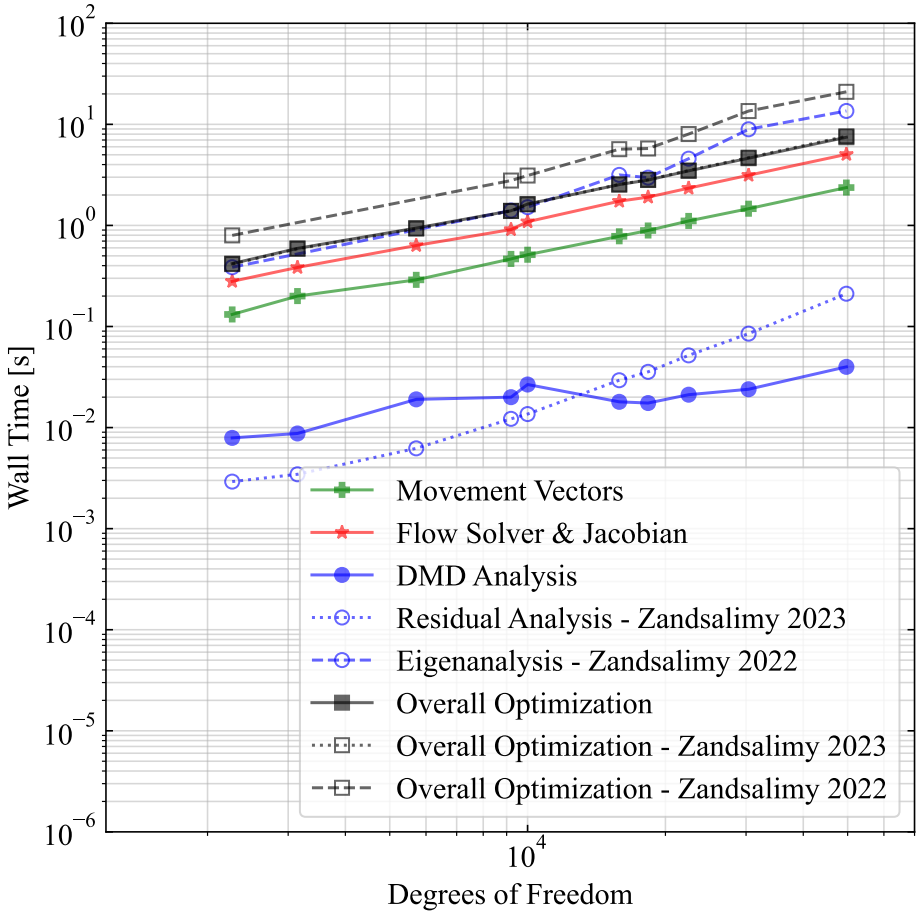


Fig. 17 The optimization run-time of the Euler problem

V. Conclusion

This study introduces innovative strategies to enhance stability and optimize mesh structures in unstructured finite volume methods. Addressing the computational complexity limitations of a previous approach, the researchers propose a new algorithm that eliminates the need for eigenanalysis. Instead, they utilize dynamic mode decomposition to identify unstable modes, analyzing the dominant modes through DMD eigenvectors derived from the latest solution update vectors. The incorporation of automatic unstable mode detection eliminates the need for manual intervention in initiating the mesh optimization algorithm. Additionally, the study presents a novel technique for calculating vertex movement by determining the gradient of the diagonal of the Jacobian matrix in relation to mesh movement. The authors demonstrate that this innovative method not only stabilizes initially unstable computational fluid dynamics solutions but also improves the convergence rate of simulations that exhibit slow convergence.

Acknowledgement

This work is co-sponsored by ANSYS Canada and the Natural Sciences and Engineering Research Council of Canada under the Cooperative Research and Development Grant F19-00204.

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/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 2023 Forum*, 2023, p. 0833. https://doi.org/10.2514/6.2023-0833.

[3] Patankar, S., *Numerical Heat Transfer and Fluid Flow*, Series in computational methods in mechanics and thermal sciences, Taylor & Francis, 1980.

[4] LeVeque, R. J., *Numerical Methods for Conservation Laws*, Vol. 3, Springer, 1992.

[5] Toro, E. F., *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction*, Springer Science & Business Media, 2013.

[6] Eymard, R., Gallouët, T., and Herbin, R., "Finite volume methods," *Handbook of Numerical Analysis*, Vol. 7, 2000, pp. 713–1018.

[7] Barth, T., and Frederickson, P., *Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction*, Aerospace Sciences Meetings, American Institute of Aeronautics and Astronautics, 1990. https://doi.org/10.2514/6.1990-13.

[8] Roe, P., "Approximate Riemann solvers, parameter vectors, and difference schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357 – 372. https://doi.org/https://doi.org/10.1016/0021-9991(81)90128-5.

[9] Murray, R., *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 2017.

[10] Taira, K., Brunton, S. L., Dawson, S. T. M., Rowley, C. W., Colonius, T., McKeon, B. J., Schmidt, O. T., Gordeyev, S., Theofilis, V., and Ukeiley, L. S., "Modal Analysis of Fluid Flows: An Overview," *AIAA Journal*, Vol. 55, No. 12, 2017, pp. 4013–4041. https://doi.org/10.2514/1.J056060.

[11] Zandsalimy, M., and Ollivier-Gooch, C., "Residual Vector And Solution Mode Analysis Using Semi-Supervised Machine Learning For Mesh Modification And CFD Stability Improvement," *In Preparation, Journal of Computational Physics*, ????

[12] Schmid, P. J., "Dynamic mode decomposition of numerical and experimental data," *Journal of Fluid Mechanics*, Vol. 656, 2010, p. 5–28. https://doi.org/10.1017/S0022112010001217.

[13] Tu, J. H., , Rowley, C. W., Luchtenburg, D. M., Brunton, S. L., and and, J. N. K., "On dynamic mode decomposition: Theory and applications," *Journal of Computational Dynamics*, Vol. 1, No. 2, 2014, pp. 391–421. https://doi.org/10.3934/jcd.2014.1.391.

[14] 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/https://doi.org/10.1016/j.sigpro.2021.108316.

[15] Bücker, H. M., Pollul, B., and Rasch, A., "On CFL evolution strategies for implicit upwind methods in linearized Euler equations," *International Journal for Numerical Methods in Fluids*, Vol. 59, No. 1, 2009, pp. 1–18. https://doi.org/https://doi.org/10.1002/fld.1798.