

Residual Vector And Solution Mode Analysis Using Semi-Supervised Machine Learning For Mesh Modification And CFD Stability Improvement

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

Novel methods are studied to improve the performance of our previous mesh optimization approach for the stability improvement of unstructured-mesh finite-volume simulations. The residual vector as well as solution modes from Principal Component Analysis of solution vectors are analyzed for this purpose. After sufficient growth, instabilities appear as anomalies in the dominant numerical modes. Using standard classification algorithms, such outliers can be detected readily and much more efficiently compared to the eigenanalysis of the Jacobian matrix, which was required by the forebears of the current study. Further, it is essential to identify the correct local regions on the mesh for possible modification of vertex location and to remove the noise from the non-related cells. Hence, a synthetic vector is constructed from the working vector containing instabilities to simulate the behavior of the right eigenvectors in the solution. The results show the feasibility of residual vector analysis and principal component analysis of solution vectors for the stability improvement of finite-volume simulations. The new approach results in complete automation of the mesh optimization application.

Keywords: Anomaly Detection, Machine Learning, Mesh Optimization, Principal Component Analysis, Residual Vector Analysis, Stability

*Corresponding Author

Email address: mohammad.zandsalimy@ubc.ca (Mohammad Zandsalimy)

Improvement.

1. Introduction

Large-scale 3D flow simulation is an increasingly important area of research in the Computational Fluid Dynamics (CFD) industry with growing requirements for computational power and accuracy of the results. The numerical
5 solution to such problems is resource intensive and peppered with inevitable complications. Numerical stability issues and poor numerical convergence behavior are frequently recurring themes.

Aufiero and Fratoni [1] presented an approach to stabilize and accelerate the convergence of steady-state coupled Monte Carlo/thermal-hydraulics simula-
10 tions, by combining the Newton method and Monte Carlo perturbation theory. Their method was successfully demonstrated in a simplified pressurized water reactor (PWR) multi-physics simulation showing an effective stabilization and convergence acceleration of the coupled problem. Otero and Eliasson [2] implemented an implicit Lower-Upper Symmetric Gauss-Seidel (LU-SGS) relaxation
15 on a flow solver for unstructured meshes based on a multi-grid formulation. They demonstrated a convergence acceleration of three orders for inviscid transonic flows compared to explicit Runge-Kutta smoothing for multi-grid acceleration. FaSTAR [3] is among the well-known unstructured-mesh CFD codes that use multi-grid methods and GMRES to accelerate the convergence behavior.
20 Lang et al. [4] proposed a strategy to develop fast Reduced Order Models based on PCA of commercial CFD results. Their model was robust within the well-sampled input domain and the CPU time was significantly reduced. Citro et al. [5] proposed a method based on the minimization of the residual norm at each iteration with a projection basis to compute the unstable steady states and/or
25 accelerate the convergence to stable configurations. Their algorithm improved the convergence of the existing iterative schemes.

Moreover, some studies in the field utilize Machine Learning (ML) techniques for improved accuracy and/or mesh optimization of CFD problems. Ling et al.

[6] used Deep Neural Networks (DNN) to train a model for the Reynolds stress
 30 anisotropy tensor from high-fidelity simulation data. They demonstrated improved prediction accuracy from the new model compared to previous ones. Novati et al. [7] employed multi-agent reinforcement learning to estimate the unresolved subgrid-scale physics. Their unsupervised approach exhibited favorable generalization properties across grid sizes and flow conditions with the
 35 results presented for isotropic turbulence. Fidkowski and Chen [8] presented a machine-learning approach for determining the optimal anisotropy of unstructured meshes in output-based adaptive solutions. Artificial neural networks were used to predict the desired element aspect ratio from features of the primal and adjoint solutions.

40 Zandsalimy and Ollivier-Gooch [9] presented novel methods to improve the efficiency and effectiveness of a previous mesh optimization approach for unstructured-mesh finite-volume simulations [10]. The vertex selection methodology was improved and a new method was implemented to reduce the overall number of optimization iterations. However, the most computationally intensive aspect
 45 of the algorithm, the solution to the large sparse eigenproblem, remains an issue that makes it infeasible to apply to large-scale 3D simulations. In another work, Zandsalimy and Ollivier-Gooch [11] employed fast and reliable classical unsupervised classification and anomaly detection algorithms to identify unstable solution modes in the residual vector. They presented a novel method to
 50 construct synthetic vectors from the residual vector that echoes the shape and behavior of the unstable right eigenvectors. This enabled them to find highly accurate estimates of the unstable eigenmode and its gradients with respect to mesh vertex movement which fully eliminated the need for the eigenanalysis module for substantial computational savings.

55 The present paper provides a method based on residual/solution vector analysis and classification models to enhance the efficiency of the previous mesh optimization approach. Principal Component Analysis (PCA) is typically used to reduce the dimensionality of a dataset containing a large number of closely related variables while preserving the most important variations in the data. This

60 end is fulfilled by identifying a small set of uncorrelated (orthogonal) principal
 components and the transformation of the data. We normalize each solution
 vector by removing the vector average to shift the focus to more important
 variations in the data. As a result, PCA can help identify the largest solu-
 tion variations which usually correlate to the dominant modes in a numerical
 65 simulation or instabilities in the case of unstable solutions. PCA and residual
 analysis are computationally more efficient than the full eigenanalysis of the Ja-
 cobian matrix. To further reduce the computational complexity of the presented
 methodology, we perform PCA on a small subset of solution vectors in the form
 of a tall skinny matrix. The required computational resources are reduced by
 70 lowering the number of solution vectors in use. However, enough data should
 exist for the successful identification of the dominant solution modes. Singu-
 lar Value Decomposition (SVD) provides a computationally efficient method of
 calculating the principal components.

Solution mode analysis through PCA provides better estimates of the un-
 75 stable modes in comparison with the residual vector. However, PCA is more
 computationally demanding. We are seeking a balance between accuracy and
 computational cost to be able to stabilize or improve the convergence rate of
 finite-volume simulations as effectively, reliably, and quickly as possible. Un-
 stable numerical modes are highly localized in the domain of solution [9] and
 80 identifiable through solution mode analysis. In unstable problems, the domi-
 nant solution modes exhibit instabilities with outlier values and overshoots in
 the vector. Such outlier values can be detected using anomaly detection models.
 The outlier detection module may also present computational complexity and
 memory requirement issues in the final mesh optimization algorithm. Hence, dif-
 85 ferent classification models are studied to identify the one that best conforms to
 the criteria sought herein. In the next step, synthetic unstable eigenvectors are
 constructed based on the approach presented by Zandsalimy and Ollivier-Gooch
 [11] using the anomalous values in the solution mode. Furthermore, synthetic
 eigenmodes and their gradients are calculated using the synthetic vectors for
 90 mesh optimization.

2. Background

2.1. Flow Solver

The finite volume method is utilized for the discretization of the partial differential equations with 2nd-order spatial accuracy. The general conservation
 95 equation can be written as follows.

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{U}) = \mathbf{S}(\mathbf{U}) \quad (1)$$

Here, \mathbf{U} are the conserved quantities, $\mathbf{F}(\mathbf{U})$ is the flux tensor, and $\mathbf{S}(\mathbf{U})$ is the source term. To obtain a numerical solution to Equation (1), the solution domain is discretized into a set of non-overlapping control volumes Ω_i the union of which builds the entire solution domain. We can write the integral form of
 100 Equation (1) by integrating over each control volume as follows.

$$\int_{\Omega_i} \frac{\partial \mathbf{U}}{\partial t} d\Omega + \oint_{\partial\Omega_i} \mathbf{F}(\mathbf{U}) \cdot \hat{n} d(\partial\Omega) = \int_{\Omega_i} \mathbf{S}(\mathbf{U}) d\Omega \quad (2)$$

Here, the Gauss divergence theorem has been used to convert the flux divergence integral into a surface integral over the boundaries of the control volume. In this equation, \hat{n} is the unit outward pointing normal vector from the faces of cell Ω_i . The average of any property \mathbf{U} inside each cell can be expressed as,

$$\bar{\mathbf{U}}_i = \frac{1}{|\Omega_i|} \int_{\Omega_i} \mathbf{U} d\Omega \quad (3)$$

105 Here, $|\Omega_i|$ is the volume of cell Ω_i . Assuming the control volumes are fixed, we can pull the time derivative out of the first integral in Equation (2). In the finite volume method, a numerical solution \mathbf{U}_h is sought on a mesh of characteristic size h that approximates the control volume average $\bar{\mathbf{U}}_i$. As a result, we can rewrite Equation (2) as follows.

$$|\Omega_i| \frac{d\mathbf{U}_h}{dt} = - \oint_{\partial\Omega_i} \mathbf{F}_h(\mathbf{U}) \cdot \hat{n} d(\partial\Omega) + |\Omega_i| \bar{\mathbf{S}}_i = \mathbf{R}(\mathbf{U}_h) \quad (4)$$

110 In this equation, $\mathbf{R}(\mathbf{U}_h)$ is the steady state residual vector.

This methodology is locally conservative, which guarantees global conservation. The surface flux integral in Equation (4) 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. Upwind flux formulas such as Roe's scheme [12] are used for convective fluxes and central flux formulas are used for diffusive terms.
3. Find flux values using Gauss quadrature rules.

The boundary conditions are applied weakly using flux values on the boundaries.

Equation (4) can be discretized in time using an appropriate ODE time integrator. In a number of experiments in this work, the Crank-Nicolson time advance scheme is utilized as presented in Equation (5).

$$\frac{\delta \mathbf{U}_h}{\delta t} = \frac{\mathbf{U}_h^{k+1} - \mathbf{U}_h^k}{\delta t} = \frac{1}{2} (\mathbf{R}(\mathbf{U}_h^{k+1}) + \mathbf{R}(\mathbf{U}_h^k)) \quad (5)$$

where δt is the time-step size. The linearization of this equation results in the following approximation,

$$\left(\frac{1}{\delta t} \mathbf{I} - \frac{1}{2} \frac{\partial \mathbf{R}}{\partial \mathbf{U}_h} \right) \delta \mathbf{U}_h = \mathbf{R}(\mathbf{U}_h^k) \quad (6)$$

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

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} = \frac{\partial \text{FluxInt}}{\partial \text{Flux}} \frac{\partial \text{Flux}}{\partial \text{RecSol}} \frac{\partial \text{RecSol}}{\partial \text{RecCoef}} \frac{\partial \text{RecCoef}}{\partial \text{PVars}} \frac{\partial \text{PVars}}{\partial \text{CVars}} \quad (7)$$

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]. For the purpose of our work, only the matrix-vector products of the Jacobian matrix are required which enables us to use a matrix-free approach when necessary.

The implicit Euler time integration method is also utilized frequently in the current study. In this method, Equation (4) can be discretized in time using,

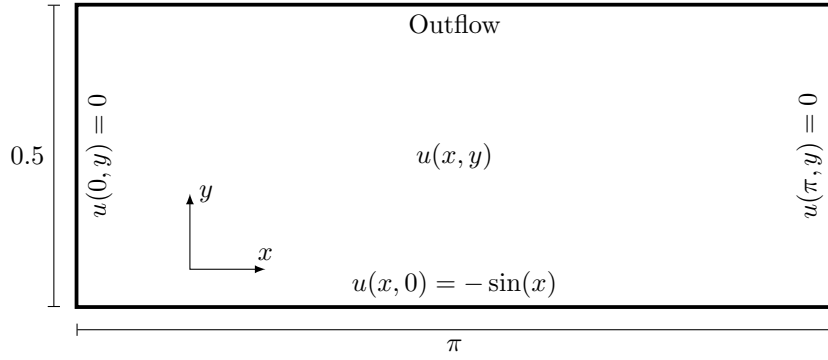
$$\frac{\delta \mathbf{U}_h}{\delta t} = \frac{\mathbf{U}_h^{k+1} - \mathbf{U}_h^k}{\delta t} = \mathbf{R}(\mathbf{U}_h^{k+1}) \quad (8)$$

This implicit method is employed in conjunction with local time-stepping and CFL evolution strategies, wherein the time-step size is dynamically adjusted based on variations in the residual [13]. The incorporation of these sophisticated time integration techniques contributes to enhanced numerical stability by mitigating the impact of unstable modes arising from the initial conditions of the solution. Despite the efficacy of these advanced time marching methods, challenges persist in certain simulation scenarios, where they may exhibit suboptimal numerical stability and convergence behavior.

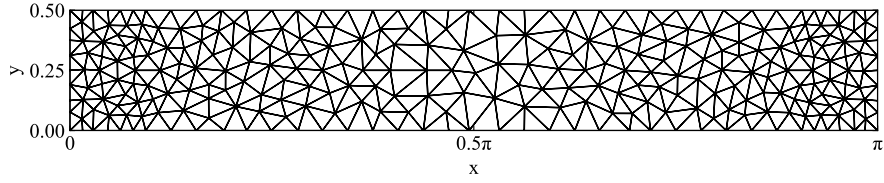
2.2. Standard Test Cases

Two classical flow problems are utilized as the main test cases in the current study which we will refer to at different points during our methodology formulation and results discussions. For this purpose, we have chosen the non-linear inviscid Burgers problem and the Euler problem on different two-dimensional solution domains. This section provides detailed descriptions of the solution domain, boundary conditions, and the converged solution for each problem. Different meshes are generated using the Generation and Refinement of Unstructured Mixed-Element Meshes in Parallel (GRUMMP) toolkit [14] containing geometric high-quality cells for each test case.

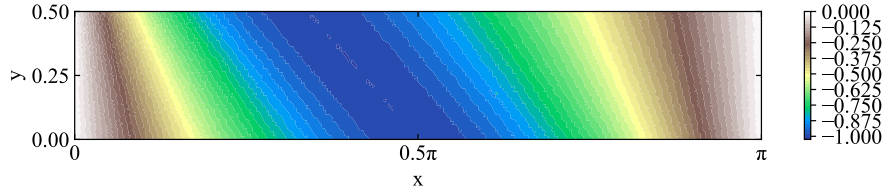
The non-linear one-dimensional inviscid Burgers problem $\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = 0$ is selected as the first test case. Here, we transform the time dimension of the Burgers equation into a second space dimension y for better representation of the solution. As a result, a new unsteady equation is formed to be solved as $\frac{\partial u}{\partial t} + \frac{\partial u}{\partial y} + u \frac{\partial u}{\partial x} = 0$. This problem is solved on a $\pi \times 0.5$ rectangular channel with the boundary conditions presented in Figure 1a. The Crank-Nicolson time-stepping method is used for the solution to this problem throughout the study. An example of the mesh used for the solution is presented in Figure 1b. It is worth noting that this mesh adheres to quality standards as per traditional mesh quality guidelines. The final converged solution for this problem is presented in Figure 1c. The color bar in this figure shows the value of $u(x, y)$ in the domain.



(a) Physical domain and boundary conditions



(b) Example mesh

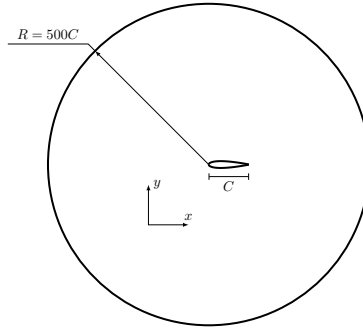


(c) Converged solution

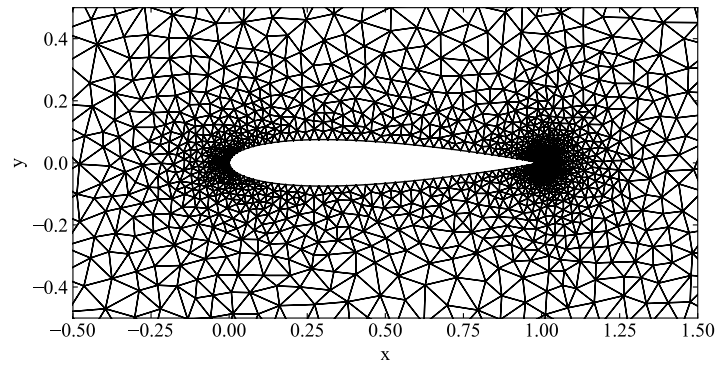
Figure 1: The standard Burgers problem

The Euler problem is selected as the next test case 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 ρ_∞ is set to 1.0, velocity in (x, y) direction to $M_\infty(\cos(\alpha), \sin(\alpha))$, and free-stream pressure P_∞ to $\frac{1}{\gamma}$. In these relations, M_∞ is the free-stream Mach number, α is the angle of attack, P_∞ is the far-field thermodynamic pressure,

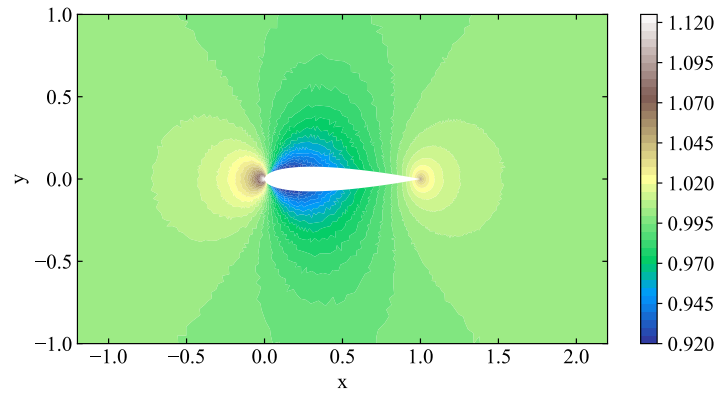
and γ is the heat capacity ratio of the fluid. Further, in this problem, stagnation values are set from the isentropic relations. This numerical simulation is carried
175 out using both Crank-Nicolson as well as Implicit Euler time-stepping methods. CFL evolution strategies [13] are utilized to increase the convergence rate of the problem in the case of Implicit Euler time-stepping. The physical domain of this solution is presented in Figure 2a and a typical triangular mesh that is used for this simulation is depicted in Figure 2b. In all the test cases that solve
180 the Euler problem, the angle of attack is set to zero and a Mach number of 0.5 is used. The final converged solution for this problem is presented in Figure 2c which shows the contours of density.



(a) Physical domain and boundary conditions



(b) Example mesh



(c) Converged solution

Figure 2: The standard Euler problem

2.3. Baseline Mesh Optimization Approach

The stabilization algorithm presented by Zandsalimy and Ollivier-Gooch [9] is used as the main mesh optimization approach in the current study. This algorithm is based on the Lyapunov stability theory [15] for dynamical systems without explicit integration. The Lyapunov theorem of stability implies that the linear time-invariant system $\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$, is locally stable if all the eigenvalues of \mathbf{A} have non-positive real parts. As a result, the system is unstable if any eigenvalue of \mathbf{A} has a positive real part. In other words, for a linear time-invariant system, stability of the equilibrium point can be characterized by the location of the eigenvalues of the Jacobian matrix [16] on the complex plane. 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 [17].

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 modify unstable eigenmodes as a means of stability improvement in CFD [9]. 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 eigenvalues 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 [9] 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 3. As seen in the *Optimization* block, the first step is deciding to perform the stabilization in a certain iteration of the solver. The issue of automating this step is addressed through the novel approach pre-

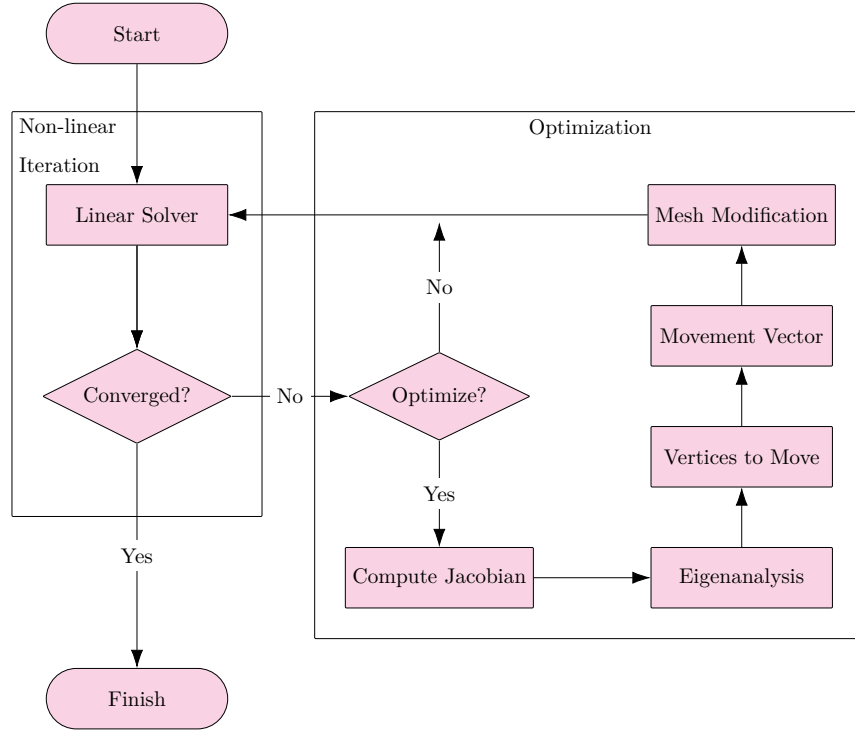


Figure 3: Overview of the stabilization approach presented by Zandsalimy and Ollivier-Gooch [9] coupled with the non-linear solver

sented in the current study. As the solution progresses, the eigenvalues of the
215 semi-discrete Jacobian change and new unstable modes might appear in the
solution. To remediate these cases, Zandsalimy and Ollivier-Gooch [9] 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
220 several iterations of the solver.

2.4. Numerical Versus Physical Instabilities

Unfortunately, no general method has yet been devised to distinguish be-
tween numerical and physical instabilities. This is important for our work, be-
cause our experience shows that numerical instabilities can be easily corrected

225 by mesh movement, whereas physical instabilities are unaffected by mesh move-
 ment. We propose here a qualitative approach to distinguishing between these
 based on the sensitivity of eigenvalues to mesh movement and the localization
 of the eigenvectors. In the case of numerically unstable modes, the gradients of
 the eigenvalues with vertex movement are usually several orders of magnitude
 230 larger than for physically unstable modes. Further, according to our experi-
 ments, the numerical modes are expected to have relatively compact support,
 unlike the physical modes which are more spread out. Vortex shedding and
 shock waves are examples of unstable physical modes while solution instabilities
 caused by local mesh defects are considered unstable numerical modes. These
 235 examples also satisfy the conditions discussed herein to distinguish physical and
 numerical modes in a simulation.

The eigenvalue gradient magnitude versus the number of non-zero values
 in the corresponding eigenvector for each solution mode can be depicted on
 a scatter plot as in Figure 4. Here, a large eigenvalue gradient and a lower
 240 number of non-zero eigenvector entries (bottom right) corresponds with a higher
 probability of a numerical solution mode while a small eigenvalue gradient and a
 larger number of non-zero entries in the eigenvector (top left) suggests a higher
 probability of a physical mode.

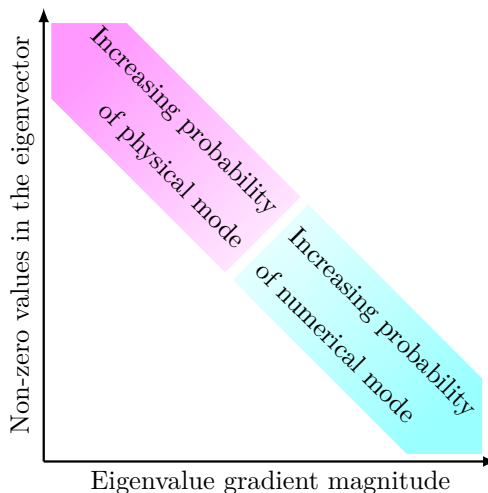


Figure 4: Non-zero values in the eigenvector versus eigenmode gradient magnitude

Figure 5 shows the scatter plot of the gradients of all the eigenmodes in
 245 a Burgers problem with respect to mesh vertex movement. Each dot in this
 figure indicates the eigenvalue gradient versus the number of non-zero values
 in the corresponding eigenvector for a given eigenmode. As seen here, the top
 left corner of the plot shows the eigenmodes with small gradients and large
 number of non-zero values in the eigenvector which are likely to be physical
 250 modes. On the other hand, the bottom right corner shows the numerical modes
 with large gradients and small number of non-zero values in the eigenvector.
 Figure 6 indicates the density of the distribution of entries with the color bar
 depicting the number of entries located in a given region of eigenvalue gradient
 versus eigenvector non-zero values. Note the logarithmic scale on both the
 255 horizontal and vertical axes. As seen in both plots for gradients with respect
 to x and y , most entries are located in the dark area which corresponds to a
 gradient of $O(10^3)$ and the number of non-zero values of $O(10^2)$. Further, we
 can see a few values with larger gradients of around 10^5 and a similar number
 of eigenvector non-zero values of around 10^2 . In practice, these values can be
 260 considered numerical modes of the simulation. Figures 7 and 8 depict the same
 phenomena for an Euler problem. As seen in Figure 8, there is a much better

separation between numerical and physical modes with two concentrated regions of high eigenvalue gradient of $O(10^3)$ versus small number of eigenvector non-zeros of $O(10^2)$ and small eigenvalue gradient of $O(10^0)$ versus large number of eigenvector non-zeros of $O(10^3)$. As a result, the concentrated region close to the bottom right corner of the spectrum can be considered numerical modes while the region located near the top left can be considered physical modes.

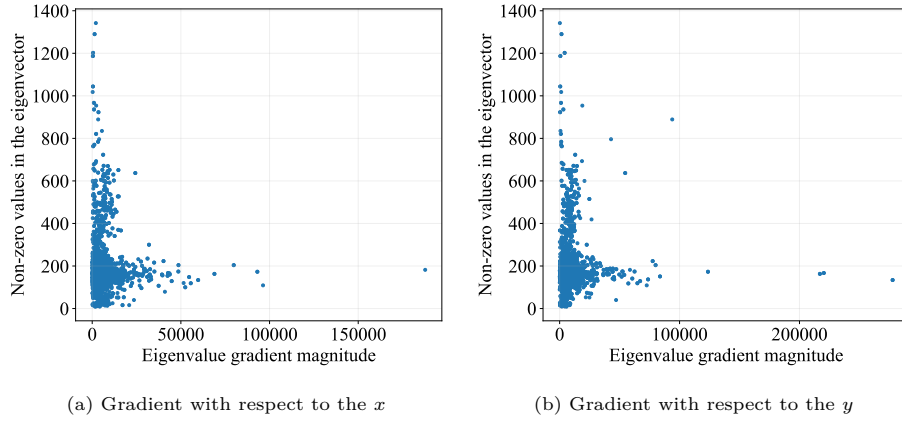


Figure 5: Scatter plot of the eigenvalue gradients versus the number of non-zero values in the eigenvector in a Burgers problem with 4500 cells

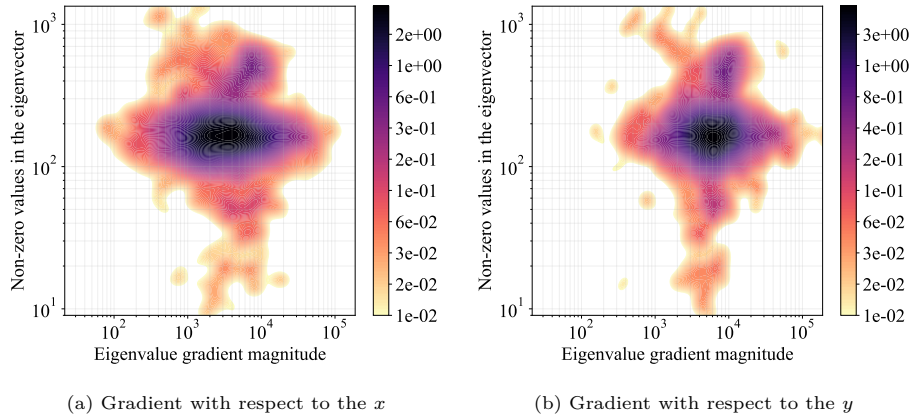


Figure 6: Density contours of the distribution of the eigenvalue gradients versus the number of non-zero values in the eigenvector in a Burgers problem with 4500 cells

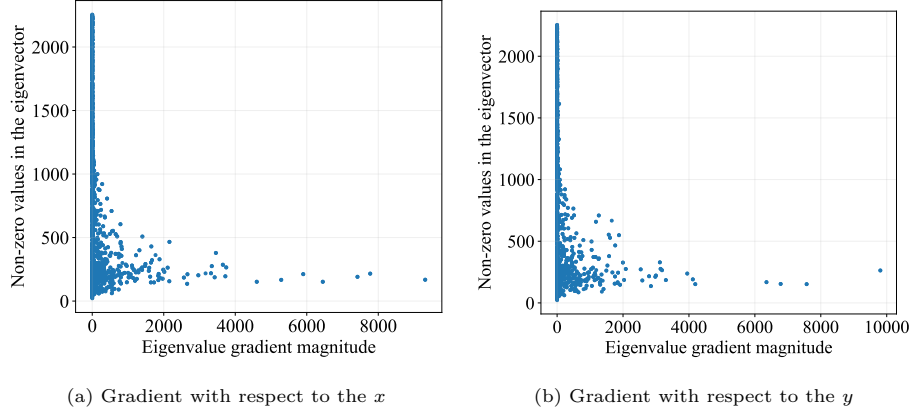


Figure 7: Scatter plot of the eigenvalue gradients versus the number of non-zero values in the eigenvector in an Euler problem with 600 cells

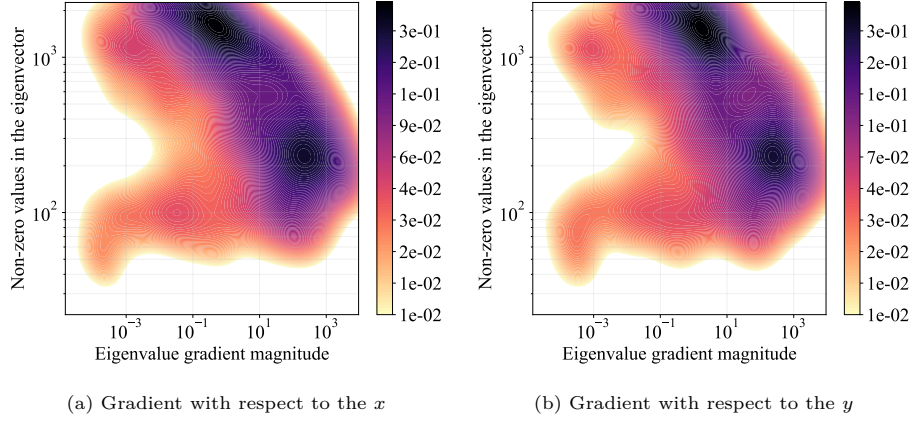


Figure 8: Density contours of the distribution of the eigenvalue gradients versus the number of non-zero values in the eigenvector in an Euler problem with 600 cells

3. Methodology

The most resource-intensive aspect of the presented methodology (Figure 3) is the solution to the large sparse eigenproblem. Removing this module can dramatically reduce the computational cost of the approach. The current work studies the feasibility of residual vector analysis and principal component analysis as alternative tools for eigenanalysis in stability improvement and mesh

optimization applications. We also present tools to construct synthetic vectors
 275 that simulate the behavior of the least stable eigenvectors for vertex selection
 and eigenvalue gradient calculation. The presented methodology utilizes an
 anomaly detection module for automatic solution mode selection. The anomaly
 detection module requires fewer computational resources in comparison to other
 aspects of the algorithm.

280 3.1. Principal Components

Principal Component Analysis (PCA) of a sequence of solution vectors de-
 termines the dominant modes in the numerical simulation. As we will show, in
 cases where real unstable numerical modes are present, the dominant modes will
 manifest outlier values with anomalous behavior. In such cases, outlier detectors
 285 can help find unstable modes while the simulation is underway. Linear stability
 theory proposes that, during blowup, the solution will increasingly resemble the
 most unstable eigenvector. As a consequence, the dominant mode should re-
 semble this eigenvector and can be used to construct a synthetic vector that is
 pointing in a similar direction as the eigenvector to be used in the stabilization
 290 algorithm. The resulting artificial vectors can help locate the problematic areas
 of the mesh, estimate the unstable eigenvalues, and compute the gradient of the
 synthetic eigenmodes with little computational cost.

The goal of PCA is to extract the most important information from the data
 as a set of new orthogonal variables referred to as principal components. The
 295 principal components can be extracted using the Singular Value Decomposition
 of a matrix $\mathbf{A} \in \mathbb{C}^{m \times n}$ into the product of form $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^H$ in which $m > n$,
 $\mathbf{U} \in \mathbb{C}^{m \times m}$ and $\mathbf{V} \in \mathbb{C}^{n \times n}$ are two unitary matrices, and $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ is a
 diagonal matrix containing the singular values. There exist efficient algorithms
 to perform singular value decomposition such as the work of Golub and Reinsch
 300 [18]. The Scalable Library for Eigenvalue Problem Computations (SLEPc) [19]
 provides systematic tools to perform SVD readily and effectively which is utilized
 in the present study.

We aim to perform SVD on a small selection of solution vectors. For a tall

skinny \mathbf{A} ($m \gg n$) the computational complexity of SVD is of $O(mn^2)$ [20]. This
305 is a great improvement over the eigenanalysis of the full Jacobian matrix which
can have a complexity of up to $O(m^3)$ for a square matrix $\mathbf{B} \in \mathbb{C}^{m \times m}$. The
caveat, however, is the requirement of performing SVD multiple times during
a finite-volume simulation compared to the full eigenanalysis which has a high
chance of succeeding in only one iteration of the approach. As a result, this
310 trade-off needs to be studied accurately to assess the feasibility of the proposed
method.

As mentioned earlier, in simulations that contain unstable numerical modes,
the largest principal components exhibit instabilities manifested as anomalies in
a few cells. As an example, the Burgers problem is solved on a mesh with 500
315 cells using the Crank-Nicolson time-stepping scheme. This solution is unstable
as seen in the residual history plot presented in Figure 9a. Figure 9b shows the
eigenspectrum of the Jacobian matrix in the first iteration of the simulation.
As depicted, the Jacobian matrix includes a single unstable mode on the right
open half of the eigenspectrum. At each iteration of the solution, SVD is per-
320 formed on the 10 most recent solution vectors, which are normalized and the
mean is subtracted from each vector to focus on the more important variations.
The snapshots of the largest solution mode at each iteration are presented in
Figure 10. As seen, the dominant solution mode exhibits anomalous behavior
in a few cells near the left boundary at around iteration 10 and after. As this
325 is a nonlinear problem, the outlier quickly contaminates neighboring cells and
causes the solution to diverge. This anomalous behavior is highly likely to be
generated by the sole unstable numerical mode in this solution.

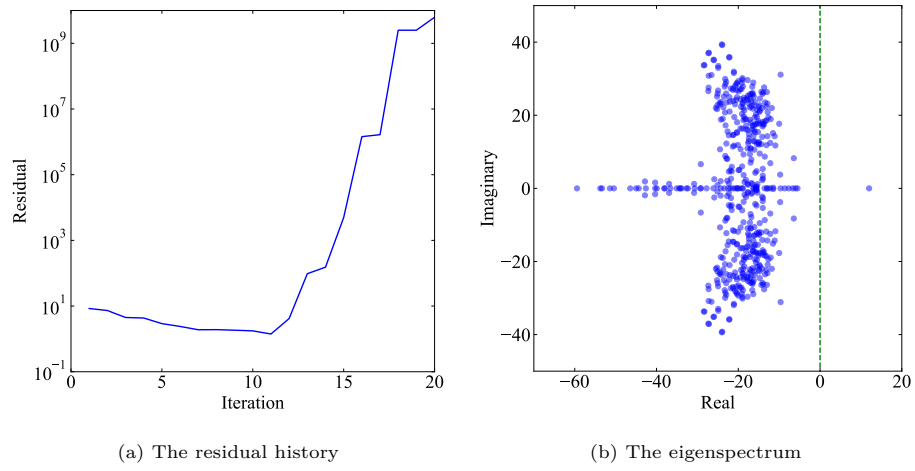


Figure 9: The solution to a Burgers problem using the Crank-Nicolson time-stepping method

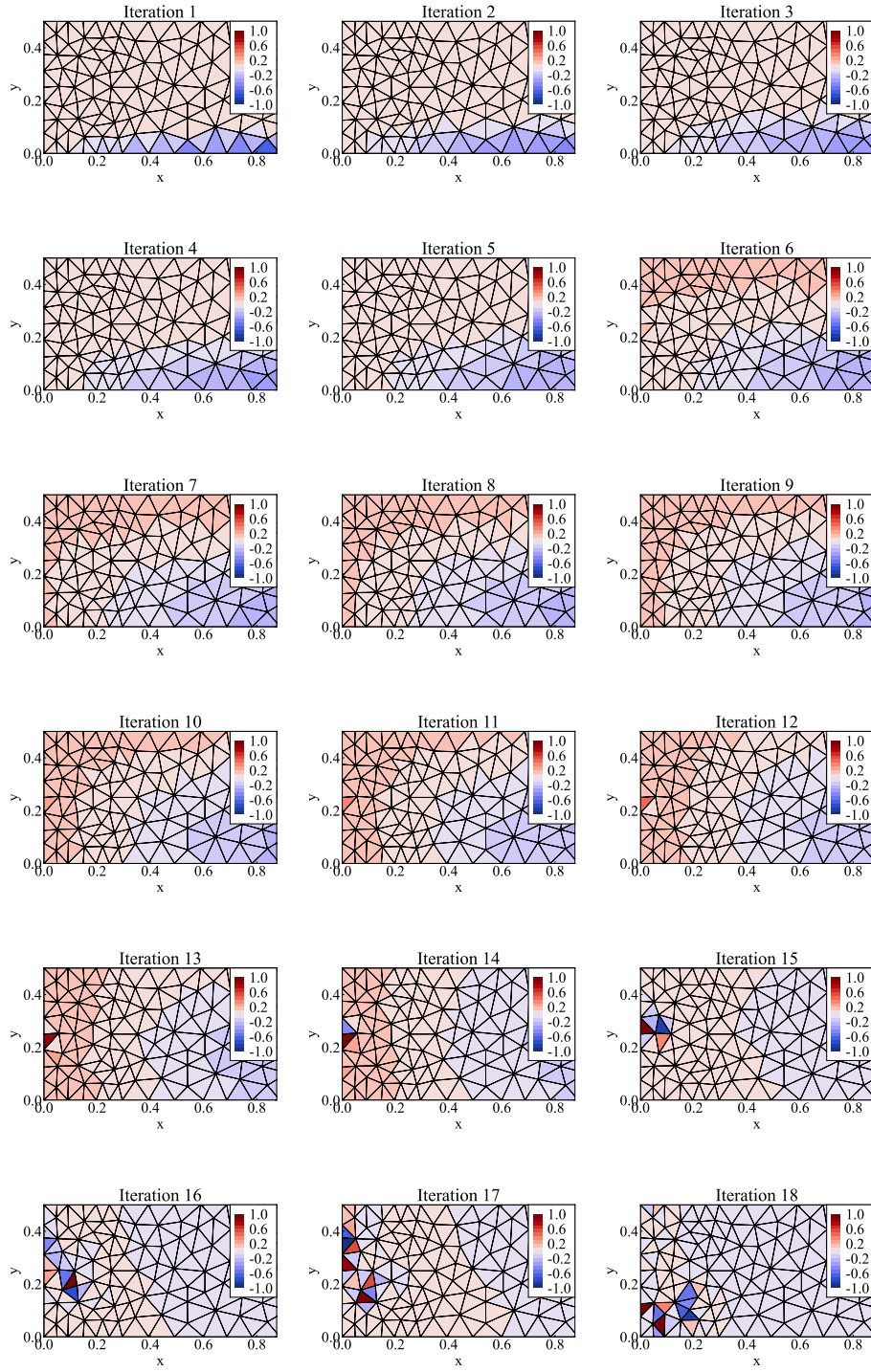


Figure 10: Largest solution mode contours in a Burgers problem

As depicted, the largest solution mode in an unstable problem exhibits anomalous behavior in the problematic control volumes. These outlier values
 330 emerge when the least stable mode of the solution becomes the dominant one. As seen in Figure 9a, the residual is decreasing during iterations 1 to 11 of the simulation. This means that up to iteration 11, the dominant solution mode is a stable one while after iteration 11 the unstable mode becomes dominant and results in a sudden growth in the residual history. These anomalies are
 335 readily recognizable to the naked eye. According to the experiments performed by Zandsalimy and Ollivier-Gooch [11], anomaly detection can be used as a powerful tool in detecting such outlier values to find the correct iteration of the solution to apply the optimization as well as identifying the problematic cells for modification.

340 A similar test is performed on an Euler problem with 600 cells and the implicit Euler time-stepping method with the results presented in Figures 11 and 12. As seen in the residual history of Figure 11a the problem is divergent. Figure 11b shows the eigenspectrum of the simulation with two real unstable eigenmodes. SVD is performed on the 10 most recent solution vectors (pressure
 345 only) and the contours of the largest solution mode are presented in Figure 12. As seen here, an outlier value is visible after iteration 5 adjacent to the top surface of the airfoil.

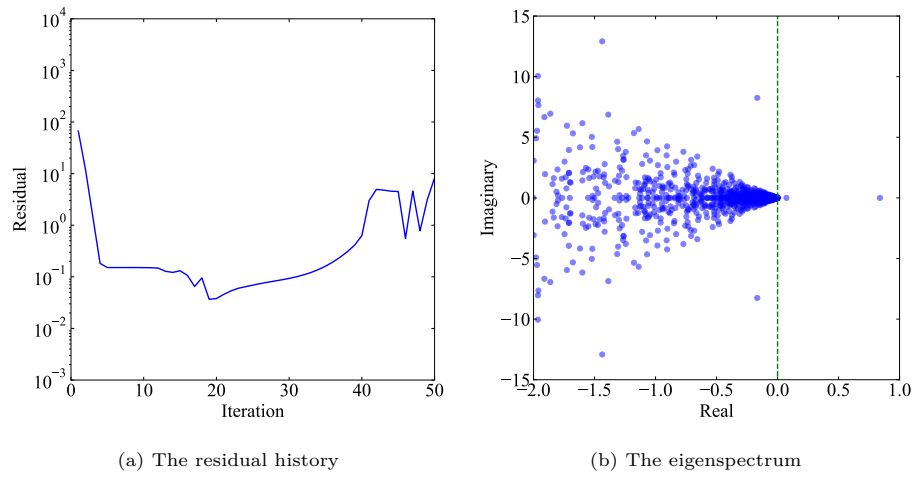


Figure 11: The solution to an Euler problem using the implicit Euler time-stepping method

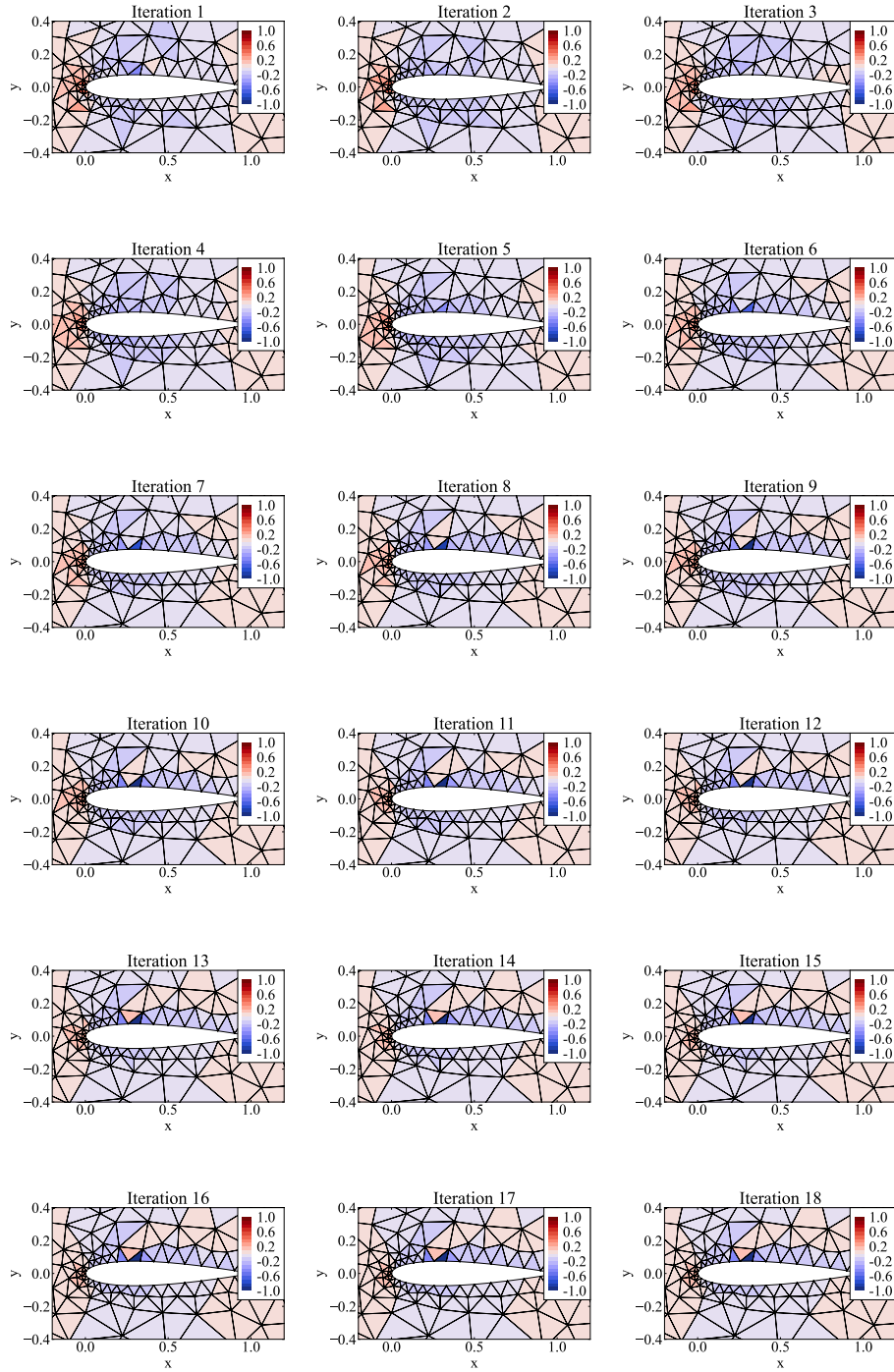


Figure 12: Largest solution mode contours in an Euler problem

3.2. Residual Vector

Similar to the principal components of the solution, the residual vector also carries important information regarding the dominant modes of the simulation. However, the identification of unstable solution modes using the residual can be a perplexing task as this vector often changes shape during the numerical solution. In cases where the dominant mode is a pure real value, the residual keeps a roughly constant shape during its growth for several solution iterations. This vector, similar to the unstable right eigenvector, points to the areas in the mesh which have the largest effect on the solution mode in question. In cases where the dominant mode has a non-zero imaginary part, the magnitude growth is accompanied by a constant phase change per iteration which makes the identification of the problematic areas in the mesh more challenging. Nevertheless, in the former case, anomaly detection models can help identify the cells with anomalous residual behavior. Figure 13 depicts the residual vector snapshots on the mesh for the same Burgers problem presented in Figure 9. As seen, a few outliers can be identified in the vector which move around the domain as the solution progresses in time. At iteration 11 of the non-linear solver, the outlier values point to a small selection of cells near the left boundary of the solution. This is consistent with the results obtained from the SVD test in Figure 10. These anomalous values point to the same cells until iteration 15. The only unstable mode in this experiment is local to the same selection of cells. Figure 14 shows the residual vector contours at every iteration of the non-linear solver for the Euler problem presented in Figure 11. As depicted here, starting at iteration 4 the residual vector exhibits outlier values near the top surface of the airfoil which stay at the same location for the rest of the simulation. This location matches the results from the SVD test performed in Figure 12.

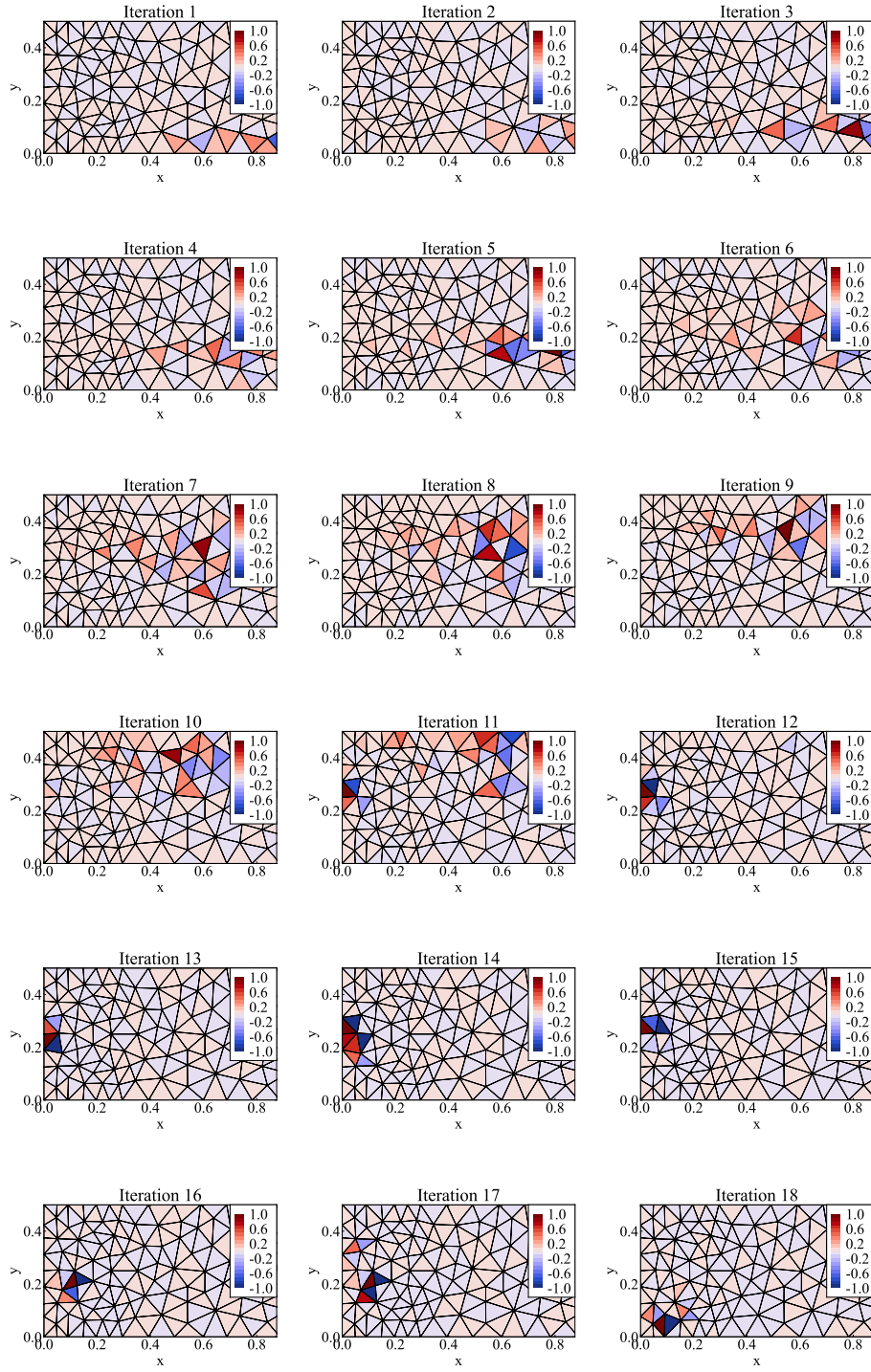


Figure 13: Residual vector contours in a Burgers problem

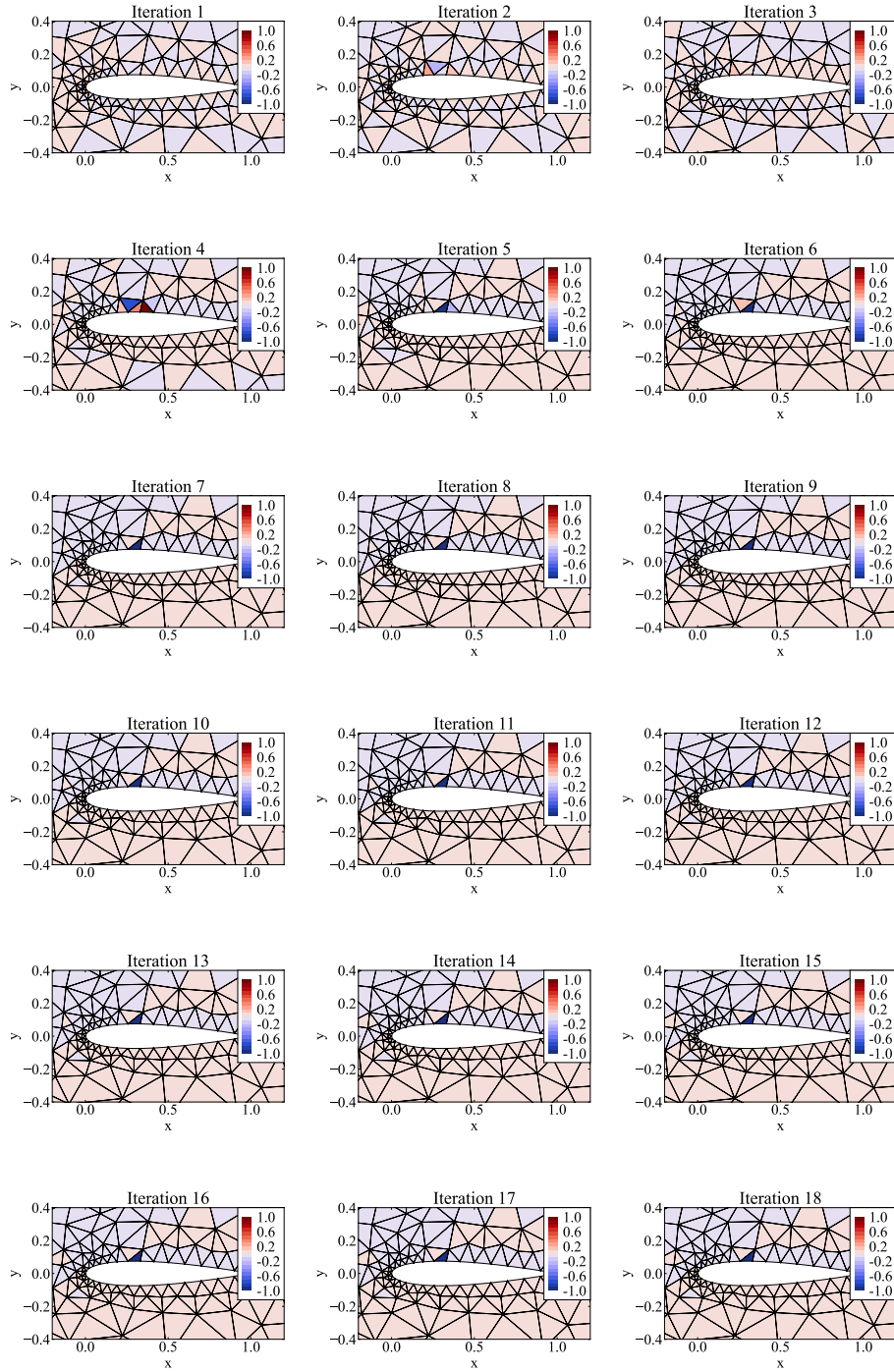


Figure 14: Residual vector contours in an Euler problem

3.3. Anomaly Detection

375 To use the principal components of the solution or the residual to guide mesh
movement to stabilize a simulation, we must be able to identify the location in
the mesh where numerical problems arise. For this purpose, we rely on anomaly
detection methods, which locate patterns in the data that do not conform well to
the overall structure and behavior. Defining the normal behavior with a precise
380 boundary can be a challenging task that hinders our ability to effectively detect
anomalies. The normal region may develop over time and this alters the notion
of outliers. The anomaly detection model selected for a given application de-
pends on different factors such as the availability of labeled data and the type of
anomalies. Such methods include Classification, Clustering, Nearest Neighbors,
385 Statistical, and Deep Learning models [21]. The majority of these methodolo-
gies are developed for low-dimensional data. Deep Learning models are best
suited for high-dimensional spaces such as multivariate datasets, images, and
videos. The present study employs PCA to reduce the dimensionality of the
dataset which enables us to use classical ML models for anomaly detection on
390 the low dimensional space.

Anomalies can be of different types [22] which is another factor in choosing
the right approach for detection. Global anomalies, which are the most common
type of outliers, refer to observations that deviate largely from the rest of the
data. As mentioned, finding the exact amount of deviation (normal behavior
395 boundary) is a challenging task and an active area of research. The observations
considered anomalous in the present study are mainly global values. However,
the vectors used herein can undergo large changes in time which makes our
task even more strenuous. Contextual anomalies are only considered so in a
given circumstance. Take the rainfall in a certain area as an example. A large
400 amount of rainfall in the winter season might be considered normal behavior
while a small amount of rain in the summer can be considered anomalous. In
our case, the changes in the solution mode over time are deemed contextual
and can change our notion of anomalous behavior. Collective anomalies happen
when a subset of the observations form an anomalous cluster. In this case,

405 each observation in the cluster might be considered normal while the group as
a whole is an anomalous observation.

Usually, two distinctions are made regarding anomaly detection applications. Outlier detection refers to the cases where the training data contains anomalies defined as observations that are far away from the rest of the entries. These
410 models fit the regions with the highest concentration of training values and select the deviant observations as outliers. Novelty detection refers to applications in which the training data does not contain anomalies. In this case, the goal is to decide if a new observation belongs to the region suggested by the training data or not. Outlier detection models are usually unsupervised while novelty
415 detection is semi-supervised. Both approaches are tested herein to find the best-performing one for our application.

A fast and reliable anomaly detection module is crucial for the current study as we aim to reduce the computational cost of the mesh optimization and stability improvement approach as much as possible. The working vector (either
420 the residual vector or the solution modes from PCA) is provided as input to the model which is then checked for possible outlier values for further investigation through the methodology presented herein. Furthermore, this module can help automate the stabilization process which is a shortcoming of the previous studies in which human intervention is critical for a successful optimization outcome
425 [9, 10].

The standard libraries provided in scikit-learn [23] open source software are utilized as the main machine learning tools in the present study. Among the different models, we expect the Local Outlier Factor to perform well on our datasets due to the nature of our mesh optimization approach. The Elliptic
430 Envelope model assumes the data has a Gaussian distribution and fits an ellipse to the training set. One Class SVM has a high sensitivity to outliers and might not perform as well as expected. It requires numerous experiments with hyper-parameters for fine-tuning to prevent over-fitting of the training set. The Isolation Forest model classifies data points by randomly selecting a split be-
435 tween the maximum and minimum values of a randomly selected feature. In

such models, outliers are detected based on the depth of the branch; deeper branches reduce the likelihood of anomalous values.

We have conducted experiments on different outlier detection models to select the best-performing one for the application of the current study. The Burgers problem presented in Figure 9 and the Euler problem of Figure 11 are
440 selected as our test cases. We aim to detect the anomalous behavior before it becomes too large in magnitude to give the algorithm enough time for effective mesh adjustments. The elected anomaly detection methods include the Robust Covariance [24], Radial Basis Function (RBF) One-Class Support Vector
445 Machine (SVM) [25], Stochastic Gradient Descent (SGD) One-Class SVM [26], Isolation Forest [27], and Local Outlier Factor [28]. Model training in this test is performed fully unsupervised. The γ value in the SVM models is set to 0.1 and the number of neighbors in the Local Outlier Factor is set to 10. The contamination factor in the Robust Covariance, Isolation Forest, and Local Outlier
450 Factor models is set based on their respective original papers.

The γ parameter in SVM intuitively dictates the extent of influence exerted by an individual training example, where lower values imply a more extensive reach, while higher values signify a more localized influence. Conceptually, γ serves as the reciprocal of the radius of influence encompassing samples designated as support vectors by the model. A very small γ constrains the model
455 excessively, rendering it unable to grasp the intricacies of the data. In such instances, the influence region of any selected support vector spans the entire training set, resulting in a model that emulates a linear model with hyperplanes segregating the density centers of each class pair.

Conversely, an excessively large γ confines the area of influence solely to the support vector itself, inevitably leading to overfitting. The optimal balance lies in selecting an appropriate γ that captures the relevant information without succumbing to either extreme. Meanwhile, the contamination factor serves as a metric for gauging the degree of data impurity. Specifically, it represents the
460 proportion of outliers within the dataset. During the fitting process, this factor is instrumental in establishing the threshold on the scores assigned to individual

samples, helping to delineate the boundaries for effective model performance.

SVD is performed on the 10 latest solution vectors in the simulation of the Burgers problem presented in Figure 9. At iteration 11, the largest solution mode is extracted and presented in Figure 15a. We have manually selected a single vector entry as the outlier, indicated with a red X, and considered the true label for the data set. Note that the indicated anomalous value is for evaluation only and is not introduced in the training phase. Figure 15b depicts the residual value in each cell at iteration 12 of the same problem with four values selected as true outliers. Each of the five models is trained on both of these vectors and evaluated using the true values with the results presented in Tables 1 and 2. Both SVM models perform poorly while the LOF model has the highest accuracy of all with a value of 1.0. Herein, anomalous values are the primary target for classification. As presented, the Local Outlier Factor model has the highest performance in anomaly classification with 100% precision and recall. As shown in Table 2 for the residual test, the SVM SGD test gives the highest accuracy with a value of 0.99. Here, once again the LOF model is one of the top performers with an accuracy of 0.93.

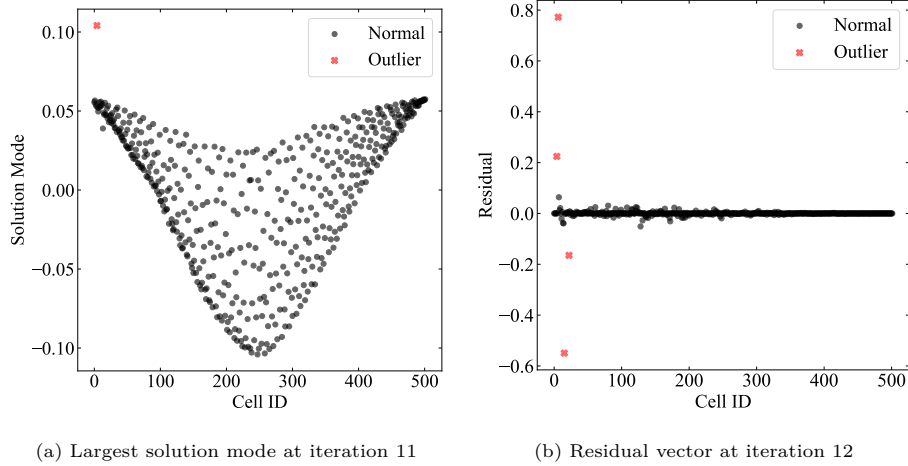


Figure 15: Two working vectors in a sample Burgers problem

Table 1: Outlier detection evaluation on the solution mode in a sample Burgers problem

Method	True Label	Precision	Recall	F1-Score	Support	Accuracy
Robust Covariance	anomaly	0.00	0.00	0.00	1	0.90
	normal	1.00	0.90	0.95	501	0.90
SVM RBF	anomaly	0.00	1.00	0.01	1	0.50
	normal	1.00	0.50	0.67	501	0.50
SVM SGD	anomaly	0.00	1.00	0.01	1	0.58
	normal	1.00	0.58	0.73	501	0.58
Isolation Forest	anomaly	0.01	1.00	0.01	1	0.69
	normal	1.00	0.68	0.81	501	0.69
Local Outlier Factor	anomaly	1.00	1.00	1.00	1	1.00
	normal	1.00	1.00	1.00	501	1.00

Table 2: Outlier detection evaluation on the residual in a sample Burgers problem

Method	True Label	Precision	Recall	F1-Score	Support	Accuracy
Robust Covariance	anomaly	0.08	1.00	0.15	4	0.91
	normal	1.00	0.91	0.95	498	0.91
SVM RBF	anomaly	0.04	1.00	0.08	4	0.83
	normal	1.00	0.83	0.90	498	0.83
SVM SGD	anomaly	0.36	1.00	0.53	4	0.99
	normal	1.00	0.99	0.99	498	0.99
Isolation Forest	anomaly	0.09	1.00	0.17	4	0.92
	normal	1.00	0.92	0.96	498	0.92
Local Outlier Factor	anomaly	0.10	1.00	0.19	4	0.93
	normal	1.00	0.93	0.96	498	0.93

The next test is performed on the Euler problem presented in Figure 11. At
iteration 5, the largest solution mode is extracted and presented in Figure 16a.
We have manually selected five vector entries as outliers which are indicated
with red X's and are considered the true labels for this data set. Figure 16b
depicts the residual value in each cell at iteration 6 of the same problem with
five values selected as true outliers. Each of the five models is trained on the raw
data and evaluated using the true values with the results presented in Tables 3

and 4. As depicted in Table 3 for the solution mode test, the LOF model has the highest accuracy of all. The precision score is much lower than the previous test, however, the recall is 100%. As seen in Table 4 for the residual test, SVM SGD is the best performing of all models. Here, the performance of the LOF model is acceptable with an accuracy of 0.98.

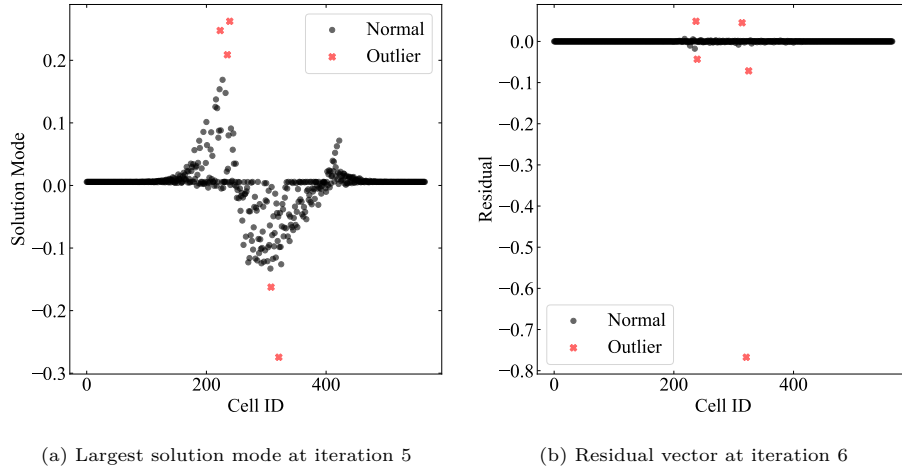


Figure 16: Two working vectors in a sample Euler problem

Table 3: Outlier detection evaluation on the solution mode in a sample Euler problem

Method	True Label	Precision	Recall	F1-Score	Support	Accuracy
Robust Covariance	anomaly	0.09	1.00	0.16	5	0.91
	normal	1.00	0.91	0.95	561	0.91
SVM RBF	anomaly	0.02	1.00	0.05	5	0.63
	normal	1.00	0.63	0.77	561	0.63
SVM SGD	anomaly	0.03	1.00	0.06	5	0.74
	normal	1.00	0.74	0.85	561	0.74
Isolation Forest	anomaly	0.05	1.00	0.09	5	0.82
	normal	1.00	0.82	0.90	561	0.82
Local Outlier Factor	anomaly	0.22	1.00	0.36	5	0.97
	normal	1.00	0.97	0.98	561	0.97

Table 4: Outlier detection evaluation on the residual in a sample Euler problem

Method	True Label	Precision	Recall	F1-Score	Support	Accuracy
Robust Covariance	anomaly	0.09	1.00	0.16	5	0.91
	normal	1.00	0.91	0.95	561	0.91
SVM RBF	anomaly	0.17	1.00	0.29	5	0.96
	normal	1.00	0.96	0.98	561	0.96
SVM SGD	anomaly	0.83	1.00	0.91	5	1.00
	normal	1.00	1.00	1.00	561	1.00
Isolation Forest	anomaly	0.10	1.00	0.18	5	0.92
	normal	1.00	0.92	0.96	561	0.92
Local Outlier Factor	anomaly	0.31	1.00	0.48	5	0.98
	normal	1.00	0.98	0.99	561	0.98

In Tables 1 to 4, the evaluation metrics *Precision* (also referred to as positive predictive value) and *Recall* (commonly known as sensitivity) are employed to assess the performance of the models. Precision represents the proportion of relevant instances among those retrieved, emphasizing the accuracy of the model’s positive predictions. On the other hand, recall measures the fraction of relevant instances that were successfully retrieved, providing insight into the model’s ability to capture all pertinent instances. Both precision and recall hinge on the concept of relevance, and the f1-Score is a comprehensive metric that combines these two measures. The f1-Score is calculated as the harmonic mean of precision and recall, as expressed in the following equation.

$$\text{F1-Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (9)$$

Further, *Support* shows the number of instances of anomaly and normal while *Accuracy* depicts the ratio of correct predictions to the total number of samples.

We have conducted tests to analyze the sensitivity of the classification accuracy with respect to changes in γ in the SVM RBF and SGD outlier detection models to assess the robustness of the selected methods. Figure 17a shows the sensitivity of the models when applied to the solution mode in the sample Burgers problem. In this case, the accuracy from the SVM RBF model undergoes

very little change while the result from SVM SGD is highly sensitive to the value of γ . Figure 17b reveals a similar test for both of these models when applied to the residual vector. Here, the SVM RBF shows smooth changes in accuracy with respect to the values of γ while the SVM SGD depicts higher accuracy with more variations. As seen, the accuracy of both models reduces with increasing γ .

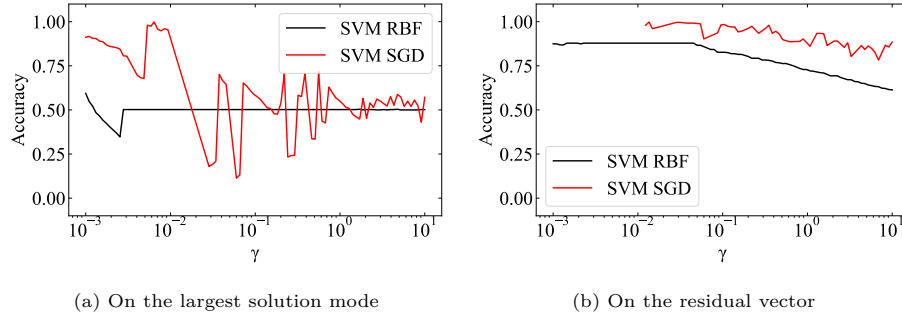


Figure 17: Sensitivity of accuracy with respect to γ

Figure 18a shows the sensitivity of the models when applied to the solution mode in the sample Euler problem. In this case, the accuracy from the SVM RBF model undergoes smooth changes, reducing with increasing γ . On the other hand, the result from SVM SGD follows the same trend while being highly sensitive to the value of γ . Figure 18b reveals a similar test for both of these models when applied to the residual vector. Here, the SVM RBF shows a sudden reduction in accuracy in a range of γ values while the SVM SGD depicts higher accuracy and fewer variations.

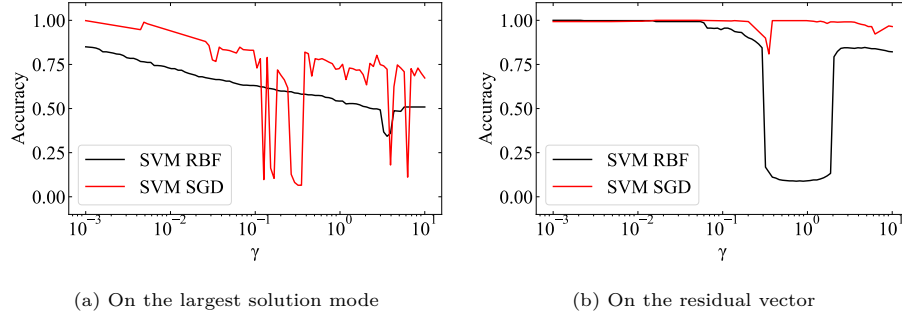


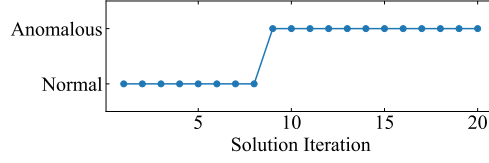
Figure 18: Sensitivity of accuracy with respect to γ

As depicted in the previous tests on the solution mode, Local Outlier Factor outperforms the other models in anomaly detection on the datasets provided herein. The SVM SGD model is best performing on the residual vector tests. However, the LOF model also has an acceptable performance in these experiments. As a result, the Local Outlier Factor model is selected for further analysis. In the current study, the detection of anomalous values in the working vector is of greater importance than finding the exact number of outliers. This fact can help improve the efficiency of the outlier detection model and reduce computational time. In other words, we only need to check the anomalous status of the maximum and minimum entries in the working vector. It is assumed that all entries, except for the minimum and maximum, are normal values in the training of the LOF model. In the next step, the minimum and maximum entries are tested to see if they belong to the normal region or not. This is a semi-supervised approach and is usually referred to as novelty detection which is preferred in the present study.

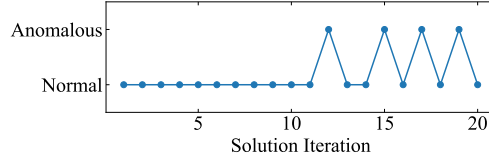
The working vector can have highly dense areas where even small deviations from the average distance between neighbors can result in the classification of an outlier. In other regions the vector might be more sparse where even larger average distances can be considered normal. To prevent false positive predictions (normal entries considered outliers) we have devised a simple workaround. In this method, the average distance of the minimum and maximum entries from

their respective neighbors is calculated. The entry is considered an outlier only if it has the largest average distance of the two. In the outlier detection context herein, “neighbors” are denoting the most similar values in a vector to the entry in question which is different from geometric cell neighbors on the mesh. Another possible solution is to set the number of neighbors (in the anomaly detection context) equal to the length of the vector. This results in the model ignoring the local deviations and only focusing on the global outliers. The latter adversely affects the computational complexity of the model and is disfavored.

The first novelty detection test is performed on the Burgers problem presented in Figure 9. The largest solution mode is used to train the LOF model for novelty detection with the maximum and minimum entries omitted. After training, the two test values are passed to the model for predictions with the results presented in Figure 19a. As seen, iteration 9 is selected as the onset of outliers in the working vector. This conforms well with our observation from the solution modes presented in Figure 10. Further, the largest anomalous entry points directly to the problematic cell in the mesh which will be modified in the next step of the algorithm. The same test is repeated on the residual vector with the results presented in Figure 19b. In this case, some outliers can be seen in the early iterations of the solver which are not pointing to the proper modes for modification. In such cases, checking the history of the location of the outlier value can help with identifying the correct non-linear iteration for optimization. No change in the outlier location is an indication of the correct unstable mode for modification. Nevertheless, the optimization iterations suggested by the novelty detection in Figure 19b are correct which will result in a stable solution.



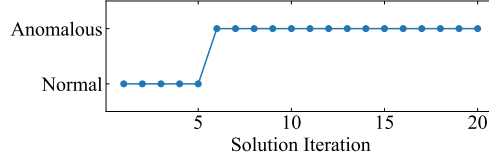
(a) Anomalous solution mode classification



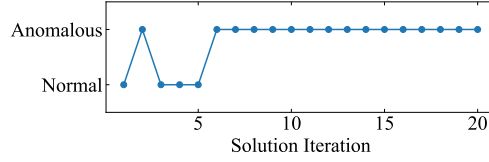
(b) Anomalous residual vector classification

Figure 19: Anomalous vector classification in the solution of a model Burgers problem

The next novelty detection experiment is performed on the Euler problem presented in Figure 11. The maximum and minimum vector entries are eliminated from the largest solution mode and the remaining is used to train the LOF model. Next, the two test values are passed to the model for predictions with the results presented in Figure 20a. As seen, iteration 6 is selected as the onset of anomalies in the vector. Once again, this matches our observation from the solution modes presented in Figure 12. The largest anomalous entry points directly to the problematic cell in the mesh. The same test is repeated for the residual vector in Figure 20b. In this case, some early outliers can be seen which are not pointing to the correct cells on the mesh for modification. Here, iteration 2 has been erroneously suggested as the onset of the mesh optimization process.



(a) Anomalous solution mode classification



(b) Anomalous residual vector classification

Figure 20: Anomalous vector classification in the solution of a model Euler problem

585 3.4. Synthetic Vectors

Solution mode outlier identification helps uncover the correct simulation iteration to apply the stability improvement algorithm. Further, anomalous solution modes are used to find the correct vertices for modification and to calculate proper movement vectors. In our application, the anomalies in the working vector are the most important characteristic. However, such vectors usually contain noise which may hinder their full potential usage. As a result, it is important to identify and remove the noise from the working vectors. A novel method for noise reduction in an anomalous vector was introduced by Zandsalimy and Ollivier-Gooch [11]. The unstable numerical modes are highly local to certain areas on the mesh and only the largest magnitude entries in the vector are used for vertex selection. This means the small entries are not important for mesh optimization and can be ignored.

The noise reduction method starts with selecting the largest magnitude vector entry and identifying the corresponding cell in the mesh. We retain this cell and all the neighbors on the mesh that contribute to its flux integral, and the rest are omitted. As discussed before, the numerical modes are usually highly local to a small area of the solution domain. This makes the presented noise

reduction approach a plausible method for achieving vectors that are similar to the numerical modes.

605 The largest solution mode at iteration 9 of the Burgers problem in Figure 9 is compared to the single unstable right eigenvector in Figure 21a. As seen, the spike in the eigenvector is successfully resolved in the synthetic vector shown with a red dashed line. This means the synthetic vector effectively points to the correct area of the mesh that requires modification for stability. Further, the
610 cosine of the angle between the two vectors is 0.750 which shows their similarity. This figure also depicts the synthetic vector resulting from the residual vector (shown with a dashed green line) at iteration 12 of the solver compared to the eigenvector. Once again, the spike in the eigenvector is correctly resolved and the cosine of the angle between the two vectors is 0.933. The next test is
615 conducted on the Euler problem presented in Figure 11 at iteration 6 of the solver. The largest solution mode in this problem (shown with a dashed red line) is compared to the largest unstable eigenvector in Figure 21b. As seen, the spike in the eigenvector is correctly resolved in the synthetic vector. In this case, the cosine of the angle between the two vectors is 0.813 which shows the two
620 vectors are once again similar. This figure also depicts the synthetic vector from the residual vector at iteration 6 of the solver (shown with a dashed green line) compared to the right eigenvector. The spike in the eigenvector is successfully resolved and the cosine of the angle between the two vectors is 0.791.

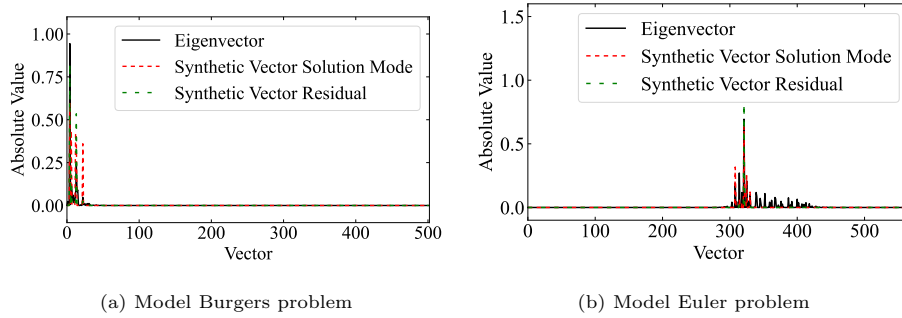


Figure 21: Synthetic vector compared to the right eigenvector

3.5. Synthetic Eigenvalue Gradients

625 Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a Hermitian Jacobian matrix depending smoothly on a real vector $\boldsymbol{\zeta} \in \mathbb{R}^n$, that is differentiable with respect to $\boldsymbol{\zeta}$. An example for $\boldsymbol{\zeta}$ can be mesh coordinates in the discretization of a finite-volume approach. If \mathbf{x} is the eigenvector associated with the eigenvalue λ , then we can compute the derivative of the eigenvalues with respect to $\boldsymbol{\zeta}$ [29],

$$\frac{d\lambda}{d\boldsymbol{\zeta}} = \mathbf{x}^H \frac{d\mathbf{A}}{d\boldsymbol{\zeta}} \mathbf{x} \quad (10)$$

630 The term $\frac{d\mathbf{A}}{d\boldsymbol{\zeta}}$ in Equation 10 can be computed as follows,

$$\frac{d\mathbf{A}}{d\boldsymbol{\zeta}} = \frac{\partial \mathbf{A}}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \boldsymbol{\zeta}} + \frac{\partial \mathbf{A}}{\partial \boldsymbol{\zeta}} \quad (11)$$

In this equation, \mathbf{U} is the solution vector. We assume, and past numerical experiments have confirmed, that the derivative of solution with respect to mesh coordinates is small [9]. As a result, the first term in Equation 11 is negligible compared to the second term. Hence, we can approximate the derivative of the Jacobian matrix with respect to mesh movement, $\frac{d\mathbf{A}}{d\boldsymbol{\zeta}}$, readily using the finite difference method.

$$\frac{d\mathbf{A}}{d\boldsymbol{\zeta}} \approx \frac{\partial \mathbf{A}}{\partial \boldsymbol{\zeta}} = \frac{\mathbf{A}(\boldsymbol{\zeta} + \delta\boldsymbol{\zeta}) - \mathbf{A}(\boldsymbol{\zeta})}{\delta\boldsymbol{\zeta}} \quad (12)$$

Changes in each vertex location only affect the residual of the cells whose flux integral depends on cells with that vertex in their reconstruction stencil. As a result, for the calculation of $\frac{d\mathbf{A}}{d\boldsymbol{\zeta}}$ only a small number of rows in the Jacobian matrix are recalculated at each iteration. Finally, using the eigenvalue gradients we can compute proper movement vectors to make favorable changes to the eigenmodes for improved stability of the numerical simulation. Note that we apply this approximation in our approach even though the Jacobian matrix herein can be non-symmetric.

645 The eigenvalue gradients with respect to changes in the mesh are computed for the model Burgers and Euler problems. Table 5 presents the eigenvalue gradients calculated using the synthetic vector as well as using the actual eigenvectors. Each major column in this table includes two sub-columns referring to

the gradients with respect to mesh movement in x and y directions. In the case
of the Burgers problem, the cosine of the angles between the actual gradients
and the synthetic vectors based on the solution mode and the residual are 0.994
and 1.000, respectively, which means each pair of vectors are quite similar. In
the case of the Euler problem using the solution mode, the cosine of the angle
is -0.955 which means the two vectors are pointing in almost exactly opposite
directions. However, in the case of the Euler problem using the residual, the co-
sine is 0.951. Further, we note that the magnitudes of synthetic gradients differ
from the actual ones. However, as the smaller gradients are still substantially
larger than the unstable eigenvalues in question, this method results in stability
improvement given proper vertex modification.

Table 5: Synthetic eigenvalue gradients compared to the actual gradients

Test	Working Vector	Synthetic Gradients		Actual Gradients	
		w.r.t. x	w.r.t. y	w.r.t. x	w.r.t. y
Burgers	Solution Mode	197.080+0I	30.476+0I	2665.025+0I	721.758+0I
	Residual Vector	518.990+0I	125.740+0I	2460.110+0I	657.230+0I
Euler	Solution Mode	-39.292+0I	-65.797+0I	164.505+0I	678.633+0I
	Residual Vector	85.060+0I	137.750+0I	164.505+0I	678.633+0I

4. Algorithm Summary

The modules discussed previously are assembled and a novel stability im-
provement approach is proposed. The architecture is depicted in Figure 22 pre-
sented in conjunction with the non-linear solver. At each non-linear iteration,
depending on the approach, either SVD is performed on a collection of solution
vectors to find the largest solution mode or simply the residual vector is stored
in memory. The solution modes/residual vector are used as training data in the
classification module. The maximum and minimum vector entries are tested
for anomalies. Hence, instabilities are detected automatically and a decision is
made to continue the optimization approach. The elimination of human inter-

670 vention from the optimization procedure is an important improvement over the
previous techniques. Going forward in the optimization, synthetic vectors and
eigenvalues are constructed. The synthetic vectors closely resemble the right
eigenvectors, pointing in similar directions. Synthetic vectors are then utilized
to find proper vertices and to calculate movement vectors for effective mesh
675 modification. The next innovation of the presented architecture is the complete
elimination of the eigenanalysis module. As mentioned earlier, the eigenvalue
problem was the bottleneck and limiting factor for mesh optimization in previ-
ous studies. The elimination of this module brings us one step closer to a more
feasible stability improvement software that can be readily incorporated into
680 current CFD solvers.

It is crucial to highlight that our proposed method relies on the linear ap-
proximation of the numerical simulation to assess the dynamic behavior of the
solution. As a result, the anticipated stability enhancement is expected to man-
ifest primarily in the linear proximity of the current numerical solution. It
685 is worth noting that in simulations characterized by high nonlinearity, where
the solution undergoes significant changes from one time step to the next (e.g.
flutter or buffet), the effectiveness of the mesh modification suggested by our
method may diminish over time. In such instances, if the simulation encounters
issues with the linear solution at a particular time step, our method can be
690 applied to modify the mesh and enhance numerical stability for that specific in-
terval. As the simulation progresses, if necessary, our method can be reapplied
iteratively to address evolving stability needs throughout the simulation. This
adaptive approach ensures that our method remains flexible and responsive to
dynamic changes in the simulation environment, consistently contributing to
695 improved numerical stability when required.

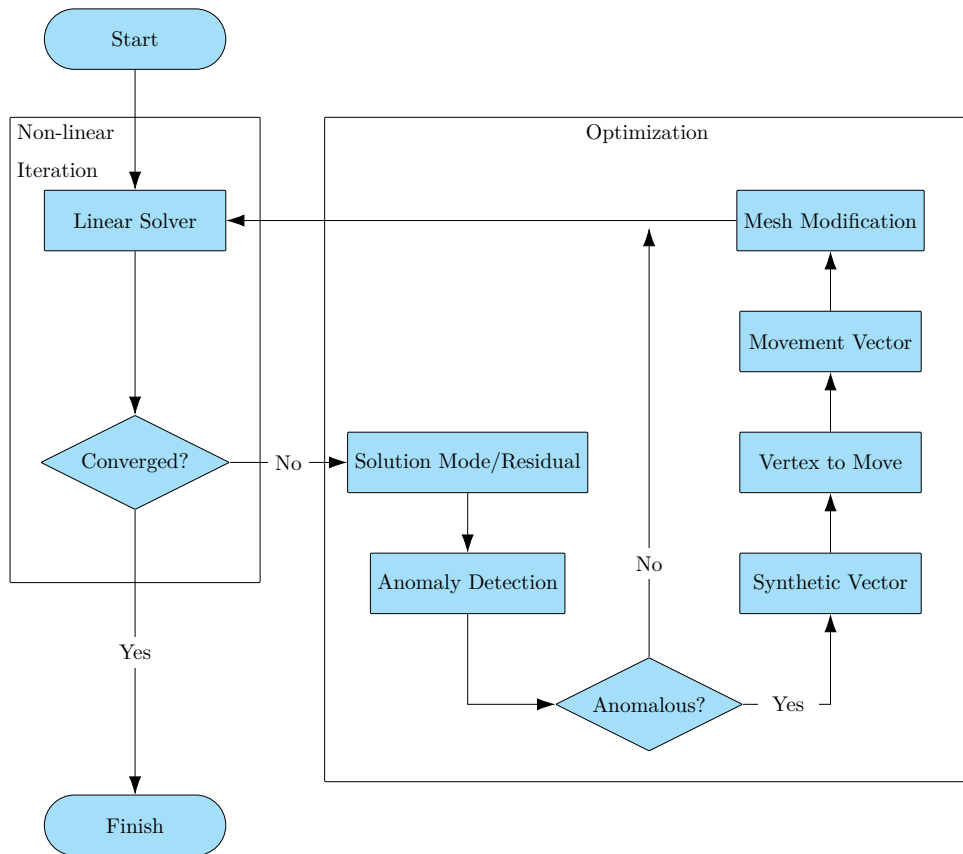


Figure 22: Overview of the novel stabilization approach

5. Results

The results of the novel mesh optimization approach are depicted in the current section. Experiments are performed on different physics, boundary conditions, and meshes. First, the effectiveness of the presented methodology is discussed. Next, the computational cost and resource savings are illustrated and compared to previous studies.

5.1. Mesh Optimization

The inviscid Burgers problem is selected as the first test case in this section as presented in Section 2.2. An unstructured mesh with 500 cells is generated using the GRUMMP software [14]. The problem is solved using the finite-volume method paired with Crank-Nicolson time integration with a fixed CFL number. Figure 23a presents the original mesh in black. As seen in the residual history of Figure 23b, the original solution before the optimization is unstable. The classification module identifies an anomalous solution mode at iteration 9 of the solver. After constructing the synthetic vectors and calculating movement vectors, the mesh is modified to the red one. It is possible to continue the simulation from the iteration where mesh optimization was performed or restart the solution completely. Herein, we choose to restart the solution for a better comparison with the previous studies. The restarted solution on the new optimized mesh is stable and converges to 10^{-10} in 40 iterations (presented in solid red). Comparing this to the work of Zandsalimy and Ollivier-Gooch [9] shows the convergence in the previous method happens at around 70 iterations. The residual history in both studies follows the same path up to iteration 30, after which, the convergence rate is higher in the current method. It is concluded that after optimization the least stable mode in the current study has a smaller real component compared to the previous work. The optimization on the same Burgers problem and mesh combination is repeated using the residual vector. According to the classification module, the residual vector is anomalous at iteration 12 of the solver. The mesh optimization at this iteration results

725 in full stabilization of the problem with the results presented in Figure 23 in blue. As seen, the new residual history closely follows the work of Zandsalimy and Ollivier-Gooch [9] with mesh movement similar to the solution mode case. As seen, using the solution modes for mesh optimization resulted in a faster convergence rate.

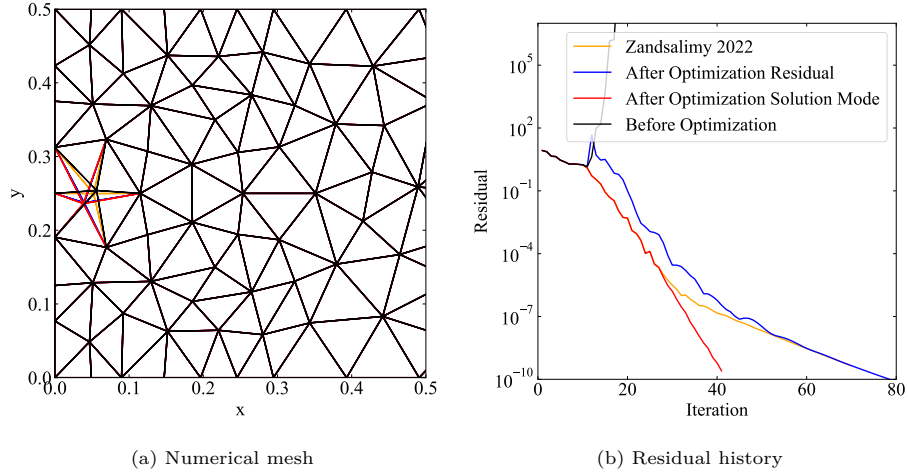


Figure 23: Mesh optimization in a Burgers problem with 500 cells

730 The same experiment is conducted on a mesh with 1100 control volumes presented in Figure 24a. The original solution is unstable as seen in Figure 24b. The first anomalous solution mode is identified by the novelty detection module at iteration 6 of the simulation. The mesh optimization approach results in generating the magenta mesh. As presented in the residual history plot, the so-
 735 lution after the first optimization iteration is still unstable. The next anomalous solution mode is identified at iteration 14 on the new solution. The optimization algorithm results in the cyan mesh and residual history. As seen, the second iteration of the optimization is also unstable. The next automatic optimization iteration occurs at iteration 15 on the latest mesh and results in the generation
 740 of the blue mesh. According to the residual history plot, the new simulation is stable and converges to 10^{-10} in less than 250 iterations. At this point, a better convergence rate is achieved in comparison to the work of Zandsalimy and

Ollivier-Gooch [11] which utilized the residual vector for mesh optimization.

This algorithm is capable of finding anomalous values even for stable simulations as PCA and anomaly detection is performed at every iteration of the simulation. The next anomalous solution mode is identified at iteration 23 of the simulation. Figure 24a presents the new mesh in green and Figure 24b shows the residual history. As seen, the convergence is initially faster than the blue residual history, however, they end up converging to 10^{-10} at the same non-linear iteration. The next anomalous solution mode is identified at iteration 66 of the simulation. This optimization iteration results in the red residual history with a much greater convergence rate in comparison to the previous iteration. The new simulation converges in 50 non-linear iterations which is a considerable improvement over the previous methodology.

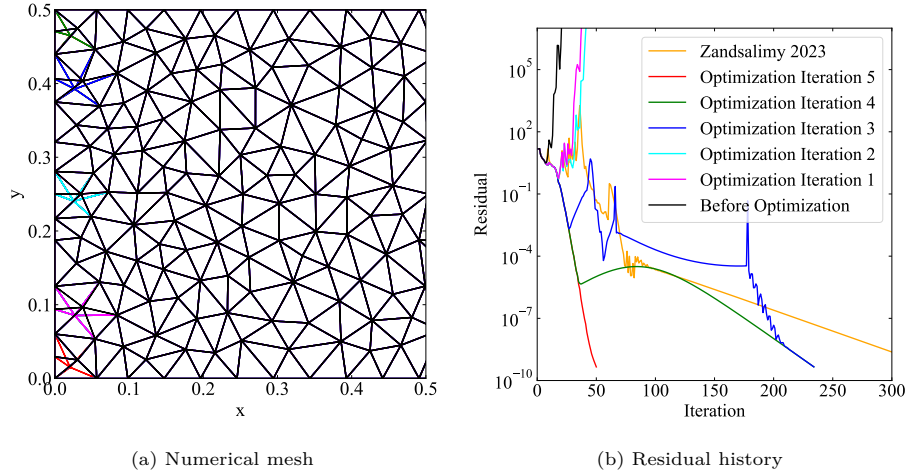


Figure 24: Mesh optimization in a Burgers problem with 1100 cells

The Burgers problem is solved on a mesh with 1400 control volumes presented in Figure 25a. The original solution is unstable as seen in Figure 25b. The first anomalous residual vector is identified by the novelty detection module at iteration 32 of the simulation. The mesh optimization approach results in generating the blue mesh. As seen in the residual history plot, the solution after the first optimization iteration is still unstable. The next anomalous resid-

ual vector is identified at iteration 40 on the new solution. The optimization algorithm results in the red mesh and residual history. As seen, the second iteration of the optimization is stable. Comparing this result with the work of Zandsalimy and Ollivier-Gooch [11] we have achieved a stable solution with a
765 slower convergence rate, but without the need for human intervention.

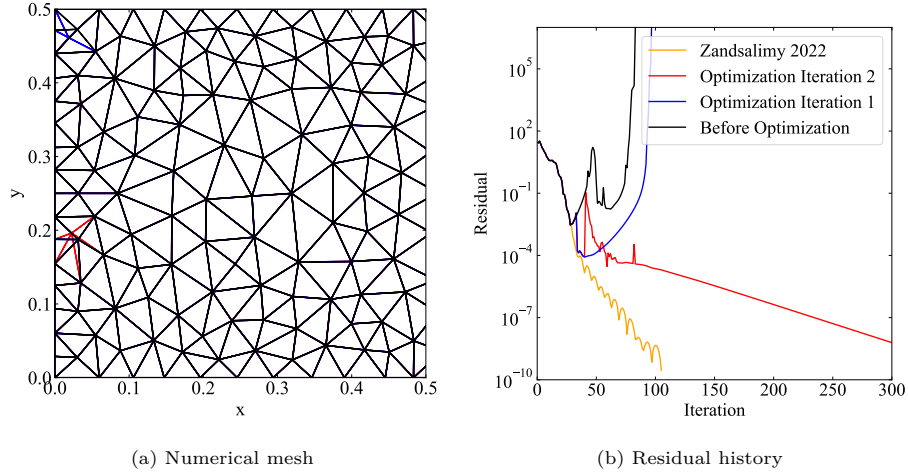


Figure 25: Mesh optimization in a Burgers problem with 1400 cells

The same Burgers problem is solved on a mesh with 1500 cells using the Crank-Nicolson time-stepping method. The initial simulation is stable as seen in Figure 26b. However, it seems that at around iteration 35, a slow converging mode dominates the simulation which is identified with a reduction in the convergence rate in the residual history. The first anomalous solution mode is
770 discovered at iteration 32 of the simulation. The optimization program modifies the mesh to the red one presented in Figure 26a. As seen in the residual plot, the convergence rate is improved substantially and the slow converging mode is eliminated by moving a single vertex in the mesh.

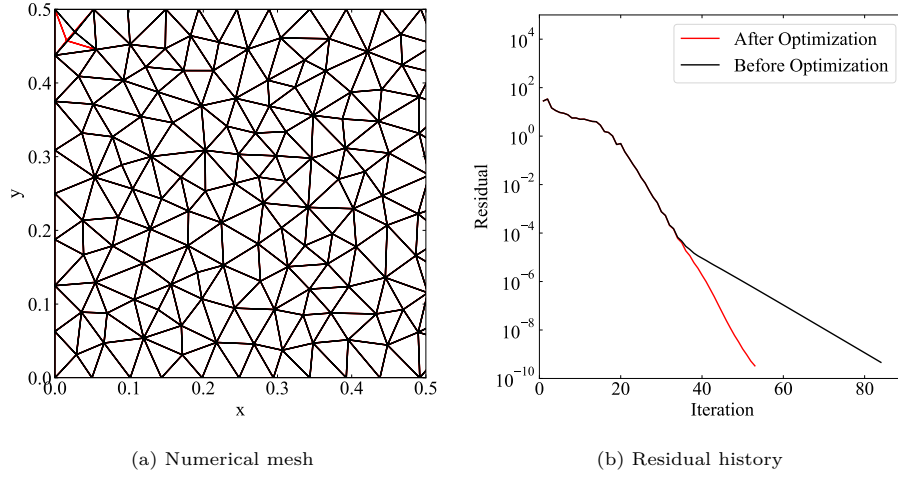


Figure 26: Mesh optimization in a Burgers problem with 1500 cells

775 The Burgers problem is repeated on a mesh with 4500 cells in Figure 27. As
depicted in the residual history plot, the initial simulation on the black mesh is
stable and converges in just over 150 iterations of the non-linear solver. Once
again, the slow convergence behavior in the residual history is seen. SVD is
performed on a collection of the 10 most recent solution vectors and the largest
780 solution mode is passed through the anomaly detection module. The maxi-
mum and/or minimum entries in the working vector are classified as novelties
at iterations 54, 56, and 61 of the solver. The optimization approach at these
iterations results in the blue, green, and red meshes, respectively. The residual
history depicts the effectiveness of the approach in eliminating the slow con-
785 verging modes from the simulation. The latest residual history in red shows
much faster convergence compared to the original simulation.

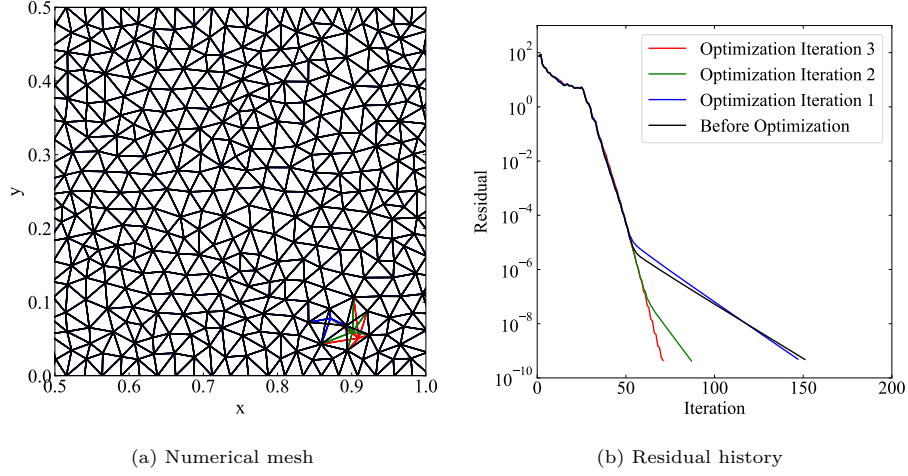


Figure 27: Mesh optimization in a Burgers problem with 4500 cells

The Euler problem is solved around the NACA-0015 airfoil inside a cylindrical outer domain with a radius of 500 airfoil chords as presented in Section 2.2. The first simulation is performed with Crank-Nicolson time integration on a mesh with 600 cells and the results are presented in Figure 28. As depicted in the residual history in black, the original simulation is unstable. The first anomalous solution mode is detected at iteration 391 on cell 314. The application of our mesh optimization methodology results in the mesh presented in blue. The residual history indicates that the new solution is stable and converges in around 2200 iterations of the non-linear solver. Compared with the result of Zandsalimy and Ollivier-Gooch [9], the current approach provides a better convergence rate while making changes to the same vertex in the mesh. The next anomalous solution mode is detected at iteration 116 of the solution on cell 413 which is located near the trailing edge of the airfoil. The latest optimization iteration results in the mesh and residual history presented in red. The convergence rate has once again improved substantially compared to the previous approach. Repeating the same experiment using the residual vector, the first outliers are detected at iteration 350 of the solver. The new mesh and residual history are presented in solid green. As seen in this case, the convergence rate

805 closely follows that of [9].

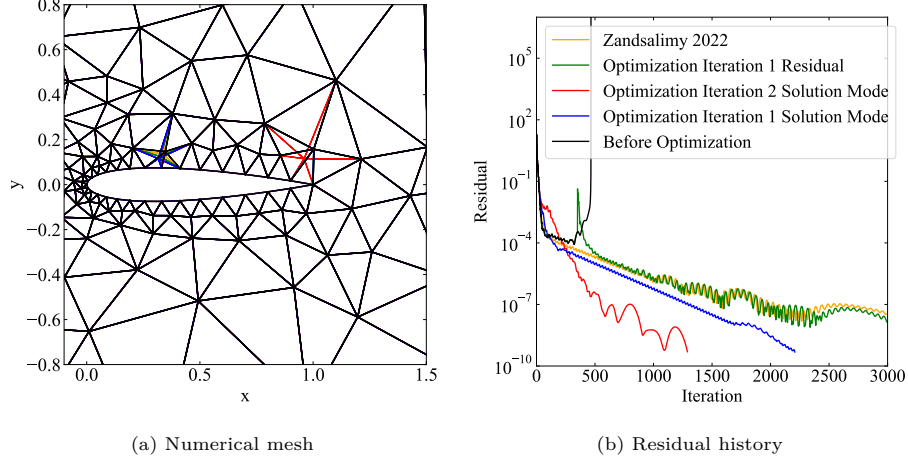


Figure 28: Mesh optimization in an Euler problem with 600 cells

The same Euler problem is solved on the same mesh with 600 cells using the implicit Euler time integration method and the results are presented in Figure 29. The solver implements local time-stepping with CFL evolution strategies for the fastest possible convergence. Put simply, the time step is increased when the solution gets closer to the final answer (decrease in residual) and vice versa.

810 The residual history plot shows that the original simulation is unstable on the black mesh. The first anomalous solution mode is detected at iteration 6 on cell 321 and the optimization program modifies the mesh to the red one. The residual history in red shows the simulation is stable, converging in 7 iterations

815 which is on par with our previous approach. Repeating the same test with the residual vector, iteration 6 is selected as the onset of outliers. Performing the optimization at this iteration results in the mesh and residual history presented in blue. Once again, a stable solution is achieved with a convergence rate comparable to the previous tests.

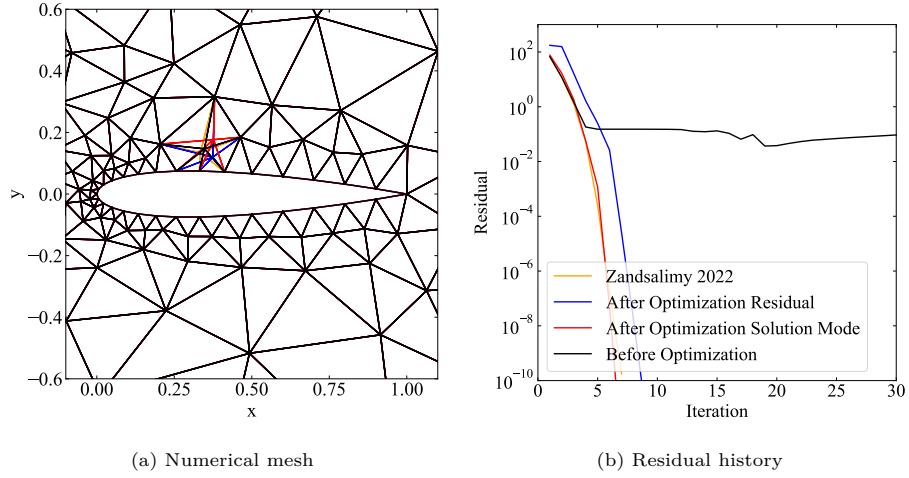


Figure 29: Mesh optimization in an Euler problem with 600 cells

820 The same Euler problem is solved on a mesh with 7600 cells using the implicit Euler time-stepping method paired with CFL evolution strategies. The anomaly detection module indicates the first novelties in the largest solution mode at iteration 10 of the implicit solver on cell 2526. Only one optimization iteration results in a stable solution presented in the red residual history of Figure 30b.

825 The modified mesh is shown in red in Figure 30a.

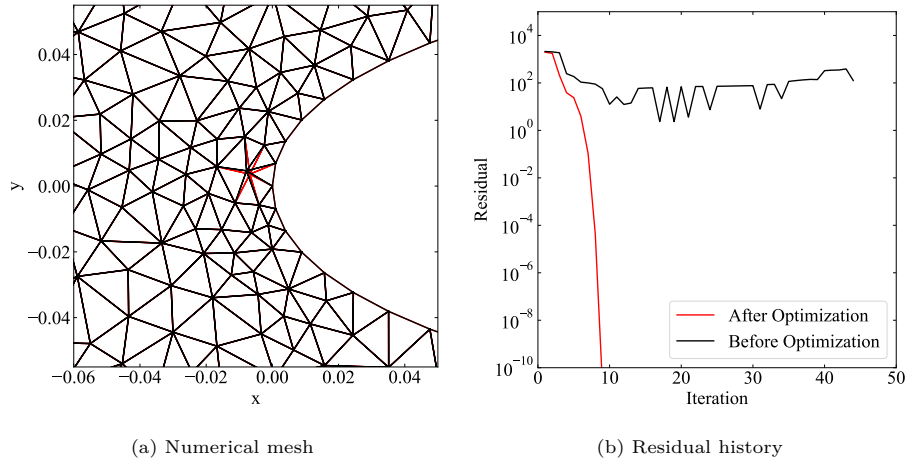


Figure 30: Mesh optimization in an Euler problem with 7600 cells

5.2. Computational Cost

One of the main drawbacks of the previous studies is the high computational complexity of their respective approach. The current work aims to reduce the run-time of different modules in the mesh optimization algorithm by improving the module complexity or eliminating the bottleneck modules. Figure 31 summarizes the run-time of different components in the presented approach for an Euler problem on different mesh sizes. In this figure, the solid lines represent the results from the current study, while non-solid lines indicate previous works. On the other hand, each symbol represents a different computational module in the algorithm. Movement vector calculation, flow solver, and Jacobian computations are the same across all three studies. Principal component analysis replaces the eigenanalysis module from the approach by Zandsalimy and Ollivier-Gooch [9]. PCA is 300 times faster than the eigenanalysis and its cost increases at a slower rate with increasing mesh size. Further, the comparison of the anomaly detection modules from the current study and the work of Zandsalimy and Ollivier-Gooch [11] depicts an improvement of one order of magnitude. This module occupies a negligible amount of computational resources in both studies. The anomaly detection component utilized in the present work is a Local Outlier Factor type with a time complexity of $O(n^2)$ [30]. The overall optimization complexity shows that the current work is 2.5 times faster compared to the work of Zandsalimy and Ollivier-Gooch [9] which is mainly due to the complete elimination of the eigenanalysis unit. On the other hand, the current work is just as efficient compared to the work of Zandsalimy and Ollivier-Gooch [11]. Because the current approach is fully automated, it can be applied to a wider variety of finite-volume simulations with more confidence.

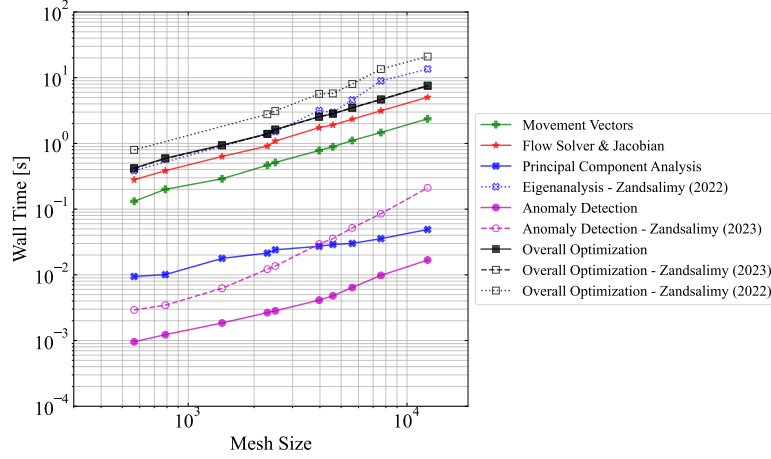


Figure 31: The optimization run-time of an Euler problem

6. Conclusion

This study prototypes novel methods for stability improvement and mesh optimization in unstructured finite-volume methods. The new algorithm is based on a previous approach in which computational complexity is one of the main shortcomings. A novel approach is presented to distinguish numerical and physical modes in a CFD simulation. The new methodology eliminates the eigenanalysis module and utilizes principal component analysis/residual vector for unstable mode identification. PCA is performed on a collection of the latest solution vectors to calculate the dominant modes in the simulation. The solution modes/residual vectors are used as training data in the classification module and the maximum and minimum entries are checked for outliers. In the next step, synthetic vectors are constructed using the anomalous solution modes which resemble the eigenvectors. These vectors are then used for mesh vertex identification and movement vector calculation. It is shown that the novel method can be utilized to stabilize initially unstable CFD solutions and also to improve the convergence rate of slow converging simulations. The novel mesh optimization methodology presented herein is 2.5 times faster than the previous approach which utilized eigenanalysis of the Jacobian matrix for stability improvement.

870 The next major contribution of the current study is the complete automation of the optimization approach. Automatic anomaly detection removes the human intervention required to initiate the mesh optimization algorithm. The new method automatically identifies diverging solution modes and provides remedies before they have a chance to completely blow up in the solution.

Acknowledgement

875 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.

References

1. Aufiero M, Fratoni M. A new approach to the stabilization and convergence acceleration in coupled Monte Carlo–CFD calculations: The Newton method via Monte Carlo perturbation theory. *Nuclear Engineering and Technology* 2017;49(6):1181–8. doi:<https://doi.org/10.1016/j.net.2017.08.005>.
880
2. Otero E, Eliasson P. Convergence acceleration of the CFD code Edge by LU-SGS. In: *3rd CEAS European Air & Space Conference*. CEAS/AIDAA; 2011:606–11.
885
3. Hashimoto A, Murakami K, Aoyama T, Ishiko K, Hishida M, Sakashita M, Lahur P. Toward the fastest unstructured CFD code “FaSTAR”. In: *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. 2012:1075. doi:10.2514/6.2012-1075.
890
4. Lang Yd, Malacina A, Biegler LT, Munteanu S, Madsen JI, Zitney SE. Reduced order model based on principal component analysis for process simulation and optimization. *Energy & Fuels* 2009;23(3):1695–706. doi:10.1021/ef800984v.

- 895 5. Citro V, Luchini P, Giannetti F, Auteri F. Efficient stabilization and acceleration of numerical simulation of fluid flows by residual recombination. *Journal of Computational Physics* 2017;344:234–46. doi:<https://doi.org/10.1016/j.jcp.2017.04.081>.
6. Ling J, Kurzawski A, Templeton J. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics* 2016;807:155–166. doi:[10.1017/jfm.2016.615](https://doi.org/10.1017/jfm.2016.615).
7. Novati G, de Laroussilhe HL, Koumoutsakos P. Automating turbulence modelling by multi-agent reinforcement learning. *Nature Machine Intelligence* 2021;3(1):87–96. doi:[10.1038/s42256-020-00272-0](https://doi.org/10.1038/s42256-020-00272-0).
- 905 8. Fidkowski KJ, Chen G. Metric-based, goal-oriented mesh adaptation using machine learning. *Journal of Computational Physics* 2021;426:109957. doi:<https://doi.org/10.1016/j.jcp.2020.109957>.
9. 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:<https://doi.org/10.1016/j.jcp.2022.110959>.
- 910 10. Zangeneh R, Ollivier-Gooch CF. Mesh optimization to improve the stability of finite-volume methods on unstructured meshes. *Computers & Fluids* 2017;156:590–601.
- 915 11. 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](https://doi.org/10.2514/6.2023-0833).
12. Roe P. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics* 1981;43(2):357–72. doi:[https://doi.org/10.1016/0021-9991\(81\)90128-5](https://doi.org/10.1016/0021-9991(81)90128-5).
- 920 13. Michalak C, Ollivier-Gooch C. Globalized matrix-explicit Newton-GMRES for the high-order accurate solution of the Euler equations. *Comput-*

ers & Fluids 2010;39(7):1156–67. doi:<https://doi.org/10.1016/j.compfluid.2010.02.008>.

- 925 14. Ollivier-Gooch C. GRUMMP version 0.7.0 user’s guide. *Department of Mechanical Engineering, University of British Columbia* 2016;.
15. Murray R. A Mathematical Introduction to Robotic Manipulation. CRC Press; 2017. ISBN 9781351469784.
16. Khalil HK. Lyapunov’s Stability Theory. London: Springer London. ISBN
930 978-1-4471-5058-9; 2015:685–90. doi:10.1007/978-1-4471-5058-9_77.
17. Taira K, Brunton SL, Dawson ST, Rowley CW, Colonius T, McKeon BJ, Schmidt OT, Gordeyev S, Theofilis V, Ukeiley LS. Modal analysis of fluid flows: An overview. *AIAA Journal* 2017;55(12):4013–41.
18. Golub GH, Reinsch C. Singular value decomposition and least squares
935 solutions. *Numerische Mathematik* 1970;14(5):403–20. doi:10.1007/BF02163027.
19. 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.
- 940 20. Vasudevan V, Ramakrishna M. A hierarchical singular value decomposition algorithm for low rank matrices. 2017. doi:10.48550/ARXIV.1710.02812.
21. Chandola V, Banerjee A, Kumar V. Anomaly detection: A survey. *ACM Computing Survey* 2009;41(3). doi:10.1145/1541880.1541882.
22. Cook AA, Misirlı G, Fan Z. Anomaly detection for IoT time-series data: A
945 survey. *IEEE Internet of Things Journal* 2020;7(7):6481–94. doi:10.1109/JIOT.2019.2958185.
23. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A,

- 950 Courneau D, Brucher M, Perrot M, Duchesnay E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 2011;12:2825–30.
24. Rousseeuw PJ, Driessen KV. A fast algorithm for the minimum covariance determinant estimator. *Technometrics* 1999;41(3):212–23. doi:10.1080/00401706.1999.10485670.
- 955 25. Yang K, Kpotufe S, Feamster N. An efficient one-class SVM for anomaly detection in the internet of things. *CoRR* 2021;abs/2104.11146. arXiv:2104.11146.
26. Nguyen M, Vien NA. Scalable and interpretable one-class svms with deep learning and random fourier features. *CoRR* 2018;abs/1804.04888. 960 arXiv:1804.04888.
27. Liu FT, Ting KM, Zhou ZH. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data* 2012;6(1). doi:10.1145/2133360.2133363.
28. Breunig MM, Kriegel HP, Ng RT, Sander J. LOF: Identifying density-based local outliers. *ACM SIGMOD Record* 2000;29(2):93–104. doi:10.1145/335191.335388. 965
29. Overton ML, Womersley RS. Second derivatives for optimizing eigenvalues of symmetric matrices. *SIAM Journal on Matrix Analysis and Applications* 1995;16(3):697–718. doi:10.1137/S089547989324598X.
- 970 30. Alghushairy O, Alsini R, Soule T, Ma X. A review of local outlier factor algorithms for outlier detection in big data streams. *Big Data and Cognitive Computing* 2021;5(1). doi:10.3390/bdcc5010001.