A Novel Approach of Mesh Optimization to Stabilize Unstructured Finite Volume Simulations

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

The stability of different computational fluid dynamics problems is evaluated and a stabilization approach based on mesh modification is presented. The problematic parts of the unstructured mesh are identified through the unstable eigenvectors and a few vertices are selected to be perturbed. The improved vertex selection methodology ensures the fastest possible optimization procedure. Perturbation vectors are computed utilizing the gradients of unstable eigenmodes with respect to the movement of the selected vertices. The optimization is performed in a single step to reduce the computational time. A new approach is presented to remediate the opposing eigenmodes in larger problems. This method is applied to different CFD problems and the results prove the robustness of our optimization scheme.

Keywords: Stability Analysis, Stability Improvement, Eigenanalysis, Finite Volume Methods, Unstructured Mesh, Mesh Optimization.

1. Introduction

Computational fluid dynamics has been a flexible and powerful tool in design and analysis applications in many areas of science and engineering for the past seven decades. The rapid development of computers has enabled scientists to tackle more complex numerical problems both in size and fidelity of the simu-

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

lations. However, most real-world applications require numerical solutions on complex geometries to capture intricate physics features which in turn compel engineers to seek a balance between accuracy and computational cost despite this lively growth in computing power. Unstructured meshes provide this balance, which offer more accurate results and are preferred over the structured counterpart in certain applications (e.g. ocean modeling [1]). Numerical stability issues in such substantial problems may give rise to difficulties in convergence, adversely affecting the reliability of the computations. Consequently, stability analysis has always been an integral facet of computational fluid dynamics and the main subject of several different studies in this area [2, 3, 4]. Yet, while the stability of numerical methods based on structured grids is well understood, studies conducted on stability analysis and improvement of methods based on unstructured grids are scant.

Constructing the numerical mesh is among the most time-consuming tasks in a numerical simulation process and requires extensive human intervention. Reliable mesh generation techniques encapsulate the knowledge and experience of experts in the scientific field in question. Deficient features in an unstructured mesh can have unfavorable effects on convergence rate as well as the accuracy of the results, aside from the risk of numerical divergence. However, as we will show later, traditional mesh quality measures can sometimes be deceptive and do not necessarily correlate with optimal convergence of the solution. Further, the stability of a numerical simulation depends not only on the mesh but also on the physics of the problem, the boundary conditions, and the reconstruction stencil [5, 6, 7]. As a result, different physics or numerics might produce stable or unstable solutions on the same mesh that is considered to be of high quality by traditional measures. Hence, the development of automated and reliable stability analysis and improvement tools are critical in computational simulations on unstructured meshes to correctly predict the stability of the solution algorithm, and to improve the convergence rate if possible.

Numerical stability studies for problems based on structured grids can be readily performed using Fourier analysis [8] which is not so easily applicable to methods based on unstructured meshes. The Lyapunov theorem [9] with a custom energy function is often utilized for stability analysis in such problems, conveniently called the energy stability analysis [8]. This method provides tools to study the equilibrium properties of a dynamical system, or in our case, the system of differential equations resulting from space discretization. If the custom scalar function (generalized energy) has only one local minimum (near the converged solution) and it is decreasing starting from all non-equilibrium initial conditions, then we expect to have a locally stable solution. Giles [10] used energy stability to analyze the time-step stability of a linearized form of Navier-Stokes equations on a 3D grid with unstructured tetrahedral elements. Moinier and Giles [11] used a similar idea and analyzed the stability of the Euler equations discretized on 3D unstructured grids. Their goal was to use the multi-grid method together with Jacobi preconditioning to accelerate the solution of the compressible Reynolds-averaged Navier–Stokes equations. Haider et al. [12] utilized the same concept to examine the eigenvalue stability of the linear semi-discrete advection equation on general unstructured meshes. They also investigated the influence of the slope reconstruction method and stencil on eigenvalue stability and presented a reconstruction method for stable discretizations. Zangeneh and Ollivier-Gooch [5] utilized the eigenvalue gradients of the semi-discrete Jacobian with respect to local changes in the mesh to stabilize their finite volume solver. Moving several vertices in an unstructured mesh, they were able to stabilize an initially unstable solution (solutions that are unstable on the initially selected mesh given physics, time-integration method, boundary conditions, and other options). In another work, Zangeneh and Ollivier-Gooch [6] investigated the energy stability due to solution reconstruction in cell-centered finite volume methods on unstructured meshes and selectively changed the stencil size of certain parts of the grid which improved energy stability of the solution. Finally, in [7] they studied the effects of boundary conditions on stability in an inviscid compressible finite volume method on unstructured meshes and optimized a hybrid boundary condition for better stability.

The present study seeks to analyze and improve the energy stability of different finite volume solutions on unstructured meshes by leveraging eigenanalysis of the semi-discrete Jacobian. The gradients of eigenvalues with respect to mesh vertex movements are calculated and used to push the eigenvalues with positive real parts to the left open-half of the eigenspectrum as presented in [5]. Contrary to that study, however, we select the minimum number of required vertices to move and stabilize the solution which results in an accelerated optimization procedure. For this purpose, the eigenvalues of the Jacobian matrix with positive real parts are selected and the right eigenvectors associated with them are identified. Each eigenvector has larger components in the cells that are contributing to the instability of that eigenmode. Our approach adds the absolute value of the right eigenvector component in a cell as a weight measure to its vertices. We use the summation of weights on vertices as a plausible approximation of how effective a vertex will be in pushing an eigenvalue with a positive real part to the left side of the complex plane. Finally, we choose a single vertex with the largest weight in the entire domain for modification. The presented algorithm herein is much faster to execute with only a few iterations of the mesh optimization procedure required to reach full stability (often only one iteration of the optimization program). We also present a method to increase the convergence rate for stable problems. Further, we will present evidence on the shortcomings of traditional mesh quality factors in predicting stability.

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, we can apply our approach at one or more intermediate stages of convergence as needed. The new 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.

2. Methods

2.1. Flow Simulation

Consider an arbitrary conservative dependent variable U which is a function of time and a vector of independent variables $\vec{x}$. We can write the conservation equation as

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

in which $\vec{F}$ and f are the flux vector and the source term, respectively. In finite volume methods the differential equations are written in divergence form which is then integrated over control volumes, and with the application of Gauss's theorem, the volume integration is converted into a surface integral across the boundaries. Doing so in an arbitrary cell results in the following equation,

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

in which V_i is the ith cell with a volume of $|V_i|$, $\vec{n}$ is the unit outward-pointing normal vector from the faces of the cell, R is called the residual, and $\bar{U}$ is the average conservative property inside the cell which can be expressed as follows,

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

This methodology is locally conservative (inside each cell), hence global conservation property is guaranteed. A detailed description of finite volume methods in computational fluid dynamics can be found in the work of [13, 14, 15, 16]. The flux integral in Equation 2 is calculated with second-order accuracy through the following procedure;

1. Reconstruct the piece-wise constant control volume averages using the linear least-squares method [17].
2. Compute the flux at each quadrature point on the cell's perimeter. Roe's scheme [18] is utilized for inviscid flux calculation in the present study.
3. Sum up the flux values using Gauss quadrature rules.

The boundary conditions are applied weakly using flux values on the boundaries. Different time integration methods, such as implicit and explicit Euler, can be used to advance the approximate solution in time. In a number of experiments in this work, the Crank-Nicolson time advance scheme is utilized as presented in Equation 4 (the bar notation is dropped for convenience).

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

where $\vec{U} = \{U_1, U_2, \ldots, U_k\}$ is the vector of control volume averages, $\vec{R}$ is the residual vector, and δt is the time-step selected for each iteration. Linearization of this equation results in the following approximation,

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

In Equation 5, $\dfrac{\partial \vec{R}}{\partial \vec{U}}$ is the Jacobian matrix which can be calculated using finite difference or chain rule differentiation and was first used for unstructured finite volume methods by [19] as presented in Equation 6.

$$\boldsymbol{A} := \frac{\partial \vec{R}}{\partial \vec{U}} = \frac{\partial \text{FluxInt}}{\partial \text{Flux}}\frac{\partial \text{Flux}}{\partial \text{RecSol}}\frac{\partial \text{RecSol}}{\partial \text{RecCoef}}\frac{\partial \text{RecCoef}}{\partial \text{PVars}}\frac{\partial \text{PVars}}{\partial \text{CVars}} \tag{6}$$

where 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 [19]. Forming the Jacobian matrix explicitly can be useful; however, only the product of this matrix with different vectors is necessary for the purpose of this study, and this can be computed through a matrix free approach.

2.2. Stability Analysis

Lyapunov stability theory [9] is used for stability analysis of the linear solver in the present study. Consider a dynamical system that satisfies the following equation.

$$\dot{x} = f(x,t), \qquad x(t_0) = x_0, \qquad x \in \mathbb{R}^n \tag{7}$$

Assume that $f(x,t)$ satisfies the standard conditions for the existence and uniqueness of solutions. A point $x^* \in \mathbb{R}^n$ is an equilibrium point of this system if $f(x^*, t) = 0$. Shifting the origin of the system allows us to conveniently have the equilibrium point at $x^* = 0$. Hence, Equation 7 is stable in the sense of Lyapunov at $t = t_0$ if for any $\epsilon > 0$ there exists a $\delta(t_0, \epsilon) > 0$ such that,

$$\|x(t_0)\| < \delta \quad \Longrightarrow \quad \|x(t)\| < \epsilon, \quad \forall t \geq t_0 \tag{8}$$

The second method of Lyapunov (direct method) determines the stability of a system without explicit integration. The method implies that if there is some measure of energy in a system then the rate of change of energy can be studied as a means of stability analysis. Let $E(x, t)$ be a non-negative function with derivative $\dot{E}$ along the trajectories of the system. If $E(x, t)$ is locally positive definite and $\dot{E} \leq 0$ locally in x and for all t, then the origin of the system is locally stable (in the sense of Lyapunov) [9].

Systematic methods for estimating the bounds on the region of attraction of equilibrium points of the non-linear system have been an important area of research that involves searching for the best possible energy function [20, 21]. The present study uses a notion similar to the kinetic energy of dynamic systems as the Lyapunov function. Take the averaged vector of solution in control volume i as U_i. We can define the following energy function (Einstein notation),

$$E \coloneqq \frac{U_i U_i}{2} \tag{9}$$

Then for homogeneous systems of degree one such as the advection, Euler, and Navier-Stokes problems the rate of change in energy is,

$$\frac{dE}{dt} = U_i \frac{dU_i}{dt} = U_i \boldsymbol{A}_{ij} U_j \tag{10}$$

If the energy function of our system of equations is always decreasing, then we have a strictly stable solution.

Furthermore, the indirect method of Lyapunov uses the linearization (if it exists) of a system to determine the local stability of the original non-linear system. Consider the system presented in Equation 7 with $f(0, t) = 0$ for all $t \geq 0$. The Jacobian matrix can be defined as,

$$\boldsymbol{A}(t) \coloneqq \left. \frac{\partial f(x, t)}{\partial x} \right|_{x=0} \tag{11}$$

Now assume,

$$\lim_{\|x\|\to 0} \sup_{t\geq 0} \frac{\|f(x,t) - A(t)x\|}{\|x\|} = 0 \tag{12}$$

Further, let the Jacobian matrix be bounded. If 0 is a uniformly asymptotically stable equilibrium point of the linearized system of equations, then it is a locally uniformly asymptotically stable equilibrium point of Equation 7. The reader is referred to [9] for details of Lyapunov stability theorem. The indirect method implies that if the original system is time-invariant and the eigenvalues of the Jacobian matrix are in the open left half of the complex plane, the system is asymptotically stable. As a result, the eigenanalysis of the Jacobian matrix indicates whether the semi-discretized system is stable or not (or has defective convergence), even before starting the solution procedure. Further, we know that the Jacobian of a discrete system depends on the mesh. This means that changing the vertex locations in the mesh can potentially change the eigenspectrum of the Jacobian which in turn can be utilized as a powerful tool in the stabilization of unstable solutions as demonstrated in [5].

2.3. Eigenanalysis

Eigenanalysis has been a useful and essential element in computational fluid dynamics from the beginning of this chapter of science. Optimal mesh generation, stability and convergence analysis, numerical error estimation, and flux vector calculation are a few examples [5, 22, 23]. Most eigenvalue solvers are based on iterative methods that utilize computationally cheap matrix-vector products instead of matrix-matrix operations or finding the matrix inverse. In general, iterative methods produce more accurate results with each iteration.

Efficient and fast eigenanalysis of large sparse matrices (resulting from finite volume discretization) is a critical aspect of the present study and one of our main concerns. In our optimization algorithm, only the eigenvalues with positive real parts are usually required. Utilizing proper methods to search in the right open-half of the eigenspectrum can reduce the solution time substantially. We utilize spectral transformation techniques such as Cayley and shift-and-invert methods to isolate the rightmost eigenvalues from the rest. The Krylov-Schur method [24] is used for the calculation of the eigenvalues. As the primary tool of eigenanalysis in the present study we use the Scalable Library for Eigenvalue Problem Computations, SLEPc [25], a software library that provides powerful tools for the solution of large sparse eigenvalue problems. This package is an extension of the Portable, Extensible Toolkit for Scientific Computation, PETSc [26], and can be used for linear eigenvalue problems with both real and complex arithmetic.

The nonzero vector $\boldsymbol{x} \in \mathbb{C}^n$ is a right eigenvector of $\boldsymbol{A} \in \mathbb{C}^{n \times n}$ if

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x} \tag{13}$$

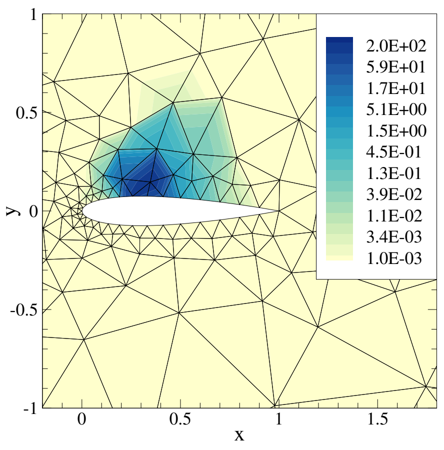
In this equation λ is an eigenvalue associated with the eigenvector $\boldsymbol{x}$. Equation 13 is a standard algebraic eigenvalue problem. Further, a nonzero $\boldsymbol{y} \in \mathbb{C}^n$ is a left eigenvector of $\boldsymbol{A}$ if the following is true,

$$\boldsymbol{y}^H \boldsymbol{A} = \lambda \boldsymbol{y}^H \tag{14}$$

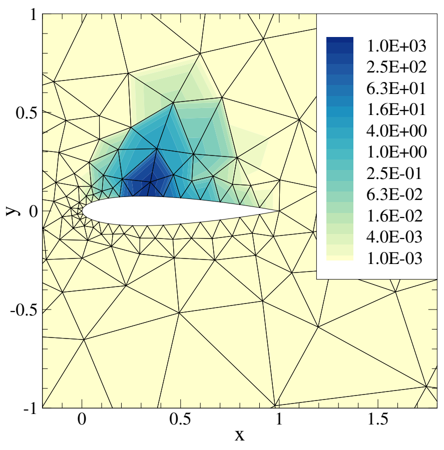
in which, $\boldsymbol{y}^H$ is the conjugate transpose (Hermitian transpose) of $\boldsymbol{y}$. In the present study, the eigenvectors are normalized to have an L_2 norm of 1.0.

2.4. Vertex Selection

The selection of vertices for modification is an important aspect of our optimization approach. The presented algorithm is capable of modifying all the vertices inside the mesh. Doing so on an originally unstable problem results in a stable solution, with the only drawback being the high computational cost. Further, according to our eigenvalue gradient analysis in Section 2.5 only certain parts of the mesh are responsible for the problematic eigenmodes. In our experience, solely local changes in the mesh will result in a stable eigenspectrum which also depicts the point. To quantify this, we have computed the eigenvalue gradient of the Jacobian matrix for an Euler problem with two unstable eigenvalues on a 2D unstructured mesh around the NACA-0015 airfoil with 600 control volumes (Section 2.5 presents details on the computation of eigenvalue gradients). Figure 1 depicts the gradient magnitude of the rightmost eigenvalue, $\lambda = 0.9633 + 0I$, with respect to the movement of each vertex in the mesh. As seen, the gradient is highly concentrated in a certain part of the mesh. The gradients further away from the airfoil are close to zero. As a result, choosing only certain vertices in the mesh is a viable option for mesh optimization. This approach will produce a much faster optimization procedure due to the limited number of vertices chosen to execute the calculations on. The unstable eigenmodes may simply include an unstable physical phenomenon or (more interestingly in the scope of the present study) a defective local mesh structure. It is noteworthy that the defectiveness of the mesh cannot always be explained through traditional mesh quality measures as we will show in Section 2.6.



(a) The gradient with respect to x.



(b) The gradient with respect to y.

Figure 1: The gradient magnitude of $\lambda = 0.9633 + 0I$ with respect to the movement of each vertex in the mesh (note that the color contours are logarithmic).

Finding the right vertices to move in the problematic parts of the mesh is our next challenge. We know that an eigenpair of the Jacobian matrix is a mode of the solution. As a result, the right eigenvector points to cells in which that eigenmode is large. As suggested by [5], the largest component of the eigenvector associated with an eigenvalue with a positive real part can be utilized to find the control volumes of interest. Then, all the vertices of these cells as well as the control volumes in their Jacobian fill are selected for modification. This method, though still not time- and resource-efficient enough, can also produce optimized meshes. The number of vertices in this methodology grows with stencil size. Further, there is an increased chance of finding contradicting eigenmodes, where stabilizing an eigenvalue with a positive real part will destabilize another eigenvalue that was originally stable.

The present study introduces an original approach to vertex selection as part of the novel optimization algorithm herein. The first step in the novel method is to find the problematic eigenvalues (with a positive real part) and extract the right eigenvectors associated with them which we will refer to as the unstable eigenvectors. Each unstable eigenvector is shaped like (pointing to) a different unstable mode in the solution at the current state. As a result, an unstable eigenvector has large components in the cells that contribute to that unstable mode while having small values (negligible) in other cells. As depicted in Figure 1, the instabilities are highly local to a certain part of the mesh which results in these cells being neighbors and sharing faces and vertices. In the next step, the summation of absolute values of the unstable eigenvector component in all the adjacent cells to a vertex is used as a selection weight for that vertex. This measure has proved to be a plausible approximation of the probability of a vertex having the largest eigenvalue gradient and hence being the most effective in eigenspectrum modification for stability. Finally, the single vertex with the largest weight measure is selected for modification and stabilization of the unstable eigenvector in question.

Note that searching the full domain of the solution is not necessary as only the region near the largest eigenvector component can contain the vertex whose movement will most affect stability. It is recommended to only perform the above-mentioned calculations in an area near the cell with the largest unstable eigenvector component. Contrary to [5], selecting only a single vertex for each unstable eigenvalue reduces the computational resources required for the optimization procedure and improves the efficiency of the method. The number of vertices to move in the new methodology does not increase with problem or stencil size which is another advantage over the previous method.

The inviscid Burgers problem, $\dfrac{\partial u}{\partial y}+u\dfrac{\partial u}{\partial x}=0$, is solved on a rectangular domain with the boundary conditions shown in Figure 2. Throughout this paper, $u(x,y)=1$ is selected as the initial conditions for the inviscid Burgers problem on this domain. The Jacobian matrix of this problem on a 2D unstructured mesh with 500 control volumes (Figure 3) has only one unstable eigenvalue of $\lambda = 11.9979 + 0I$. Figure 4 depicts the selection procedure of a single vertex for modification in this problem. In Figure 4a the absolute value of the right eigenvector components in different cells are presented. In Figure 4b the contributing weights from each cell on an incident vertex are summed up and the result points to a single vertex of our interest (indicated with a white circle).

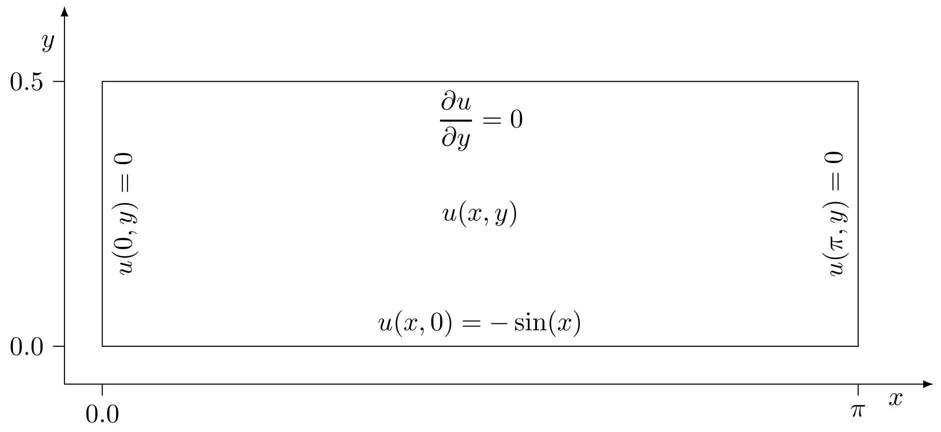


Figure 2: The physical domain and boundary conditions of the Burgers problem.

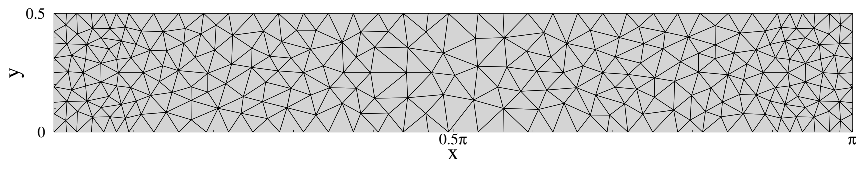
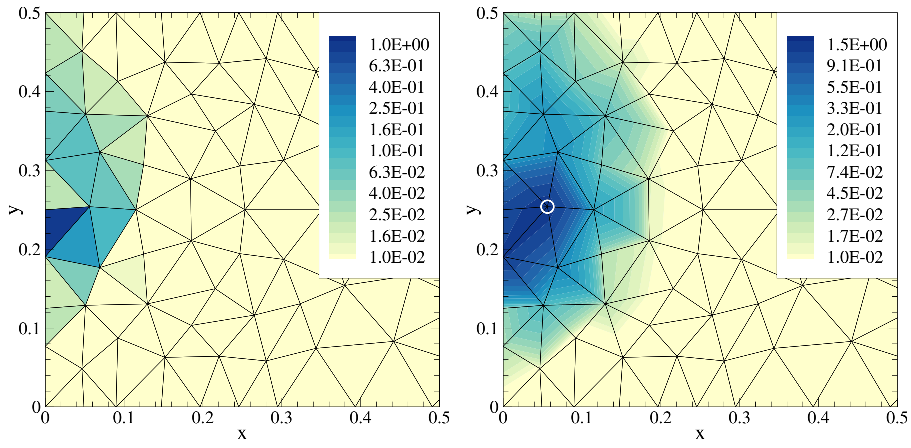


Figure 3: The unstructured mesh with 500 control volumes for the solution of the Burgers problem.



(a) The normalized absolute value of eigenvector components in the cells (each value is divided by the maximum eigenvector component).

(b) The summation of weights for each vertex.

Figure 4: The vertex selection weight in the Burgers problem on a mesh with 500 control volumes (note that the color contours are logarithmic).

2.5. Eigenvalue Gradient

According to the indirect method of Lyapunov, any eigenvalue of the Jacobian matrix that has a positive real part is an unstable eigenmode, which needs to be mitigated. The idea here is to use movements in certain mesh vertices to push those unstable eigenvalues in the eigenspectrum of the Jacobian to the open left half of the complex plane. This task requires a knowledge of how eigenvalues behave with respect to the movement of those special vertices. As a result, the eigenvalue gradients with respect to vertex movement are required in our optimization procedure. The gradients can then be utilized to find the direction and magnitude of vertex movement to push the unstable modes to the desired stable values. Moreover, the imaginary part of the spatial eigenvalues in the Jacobian matrix causes oscillations in the solution residual [27] and can change the amplification factor (hence the convergence rate) of the fully discrete problem. Our stabilization method can also be utilized to mitigate such unnecessary and potentially dangerous vibrations in the solution error which is a subject for future work.

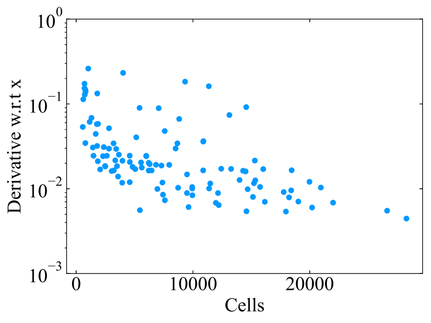
The gradients of eigenvalues with respect to mesh coordinates, ζ, can be computed directly using the finite difference method. However, this approach will require many solutions of Equations 13 and 14. Ideally, we avoid the solution to the eigenproblem as much as possible since this is the most time-consuming part of our mesh optimization program. The Jacobian matrix $\boldsymbol{A}$ is a function of both the solution vector $\vec{U}$ and the mesh coordinates ζ. We can find the gradients of eigenvalues with respect to ζ following the procedure presented by [28, 29, 30].

$$\frac{d\lambda}{d\zeta} = \boldsymbol{y}^H \frac{d\boldsymbol{A}}{d\zeta}\boldsymbol{x} \tag{15}$$

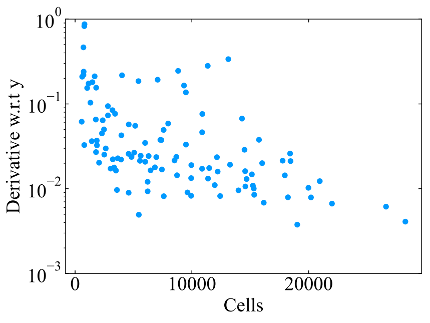
This equation is considerably more efficient to utilize than the direct finite difference approach. In this case, we only need to solve the eigenvalue problem once to find the eigenvectors. The term $\dfrac{d\boldsymbol{A}}{d\zeta}$ in Equation 15 can be computed as follows,

$$\frac{d\boldsymbol{A}}{d\zeta} = \frac{\partial \boldsymbol{A}}{\partial U}\frac{\partial U}{\partial \zeta} + \frac{\partial \boldsymbol{A}}{\partial \zeta} \tag{16}$$

We assume that at steady-state conditions, the changes in solution with respect to changes in the mesh, is much smaller than one. This means the first term in the equation is negligible compared to the second term. To show this, we have performed tests on many different meshes to solve the Euler and Burgers problems. A stable solution is carried out and with the help of our vertex selection method, the vertex corresponding to the least stable eigenmode is chosen. Through modification of this vertex, we can solve the problem until convergence again and find the derivative term in question. Infinity norm (L_∞) of solution change is utilized for this purpose. Figure 5 shows the derivative in both x and y directions versus mesh size for an Euler problem. As depicted here, the derivatives are at least 3 orders of magnitude smaller than the gradients presented in Figure 1 which proves our point. Figure 6 shows the same results for a Burgers problem in which the derivative is of negligible magnitude too.

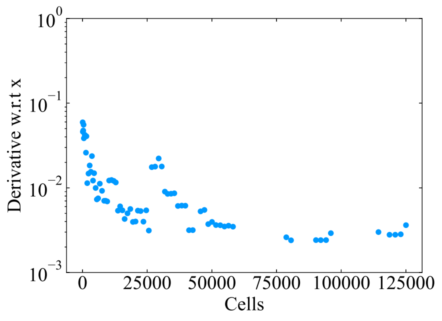


(a) Derivative in x direction.

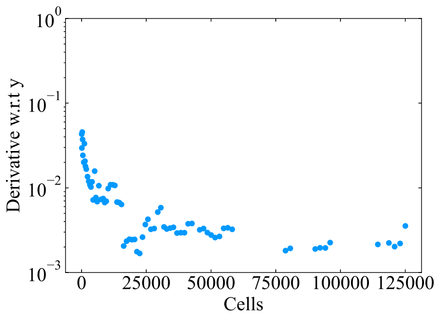


(b) Derivative in y direction.

Figure 5: Derivative of solution with respect to mesh movement versus mesh size in an Euler problem.



(a) Derivative in x direction.



(b) Derivative in y direction.

Figure 6: Derivative of solution with respect to mesh movement versus mesh size in a Burgers problem.

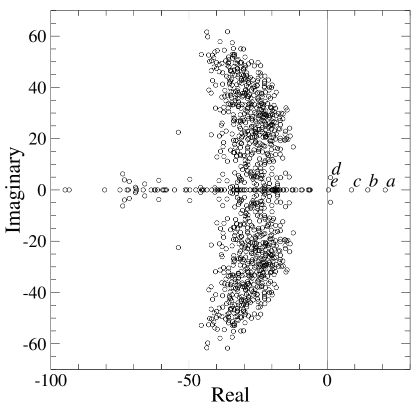
As a result, we can approximate the derivative of the Jacobian matrix with respect to mesh movement, $\frac{d\boldsymbol{A}}{d\zeta}$, readily using the finite difference method.

$$\frac{d\boldsymbol{A}}{d\zeta} \approx \frac{\partial \boldsymbol{A}}{\partial \zeta} = \frac{\boldsymbol{A}(\zeta + \delta\zeta) - \boldsymbol{A}(\zeta)}{\delta\zeta} \tag{17}$$

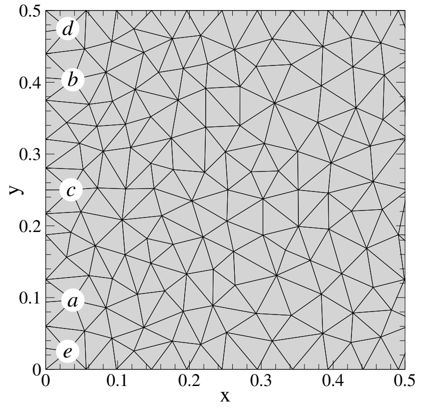
Moving each vertex in the mesh only affects the flux residual of the cells that contain that vertex in their reconstruction stencil. As a result, for the calculation of $\frac{d\boldsymbol{A}}{d\zeta}$ only a small number of rows in the Jacobian matrix are recalculated with each iteration. Such information is available from the sparsity pattern of the

Jacobian matrix. The non-zero elements in each row of the Jacobian represent the control volumes in that specific flux residual.

The next issue in our study is to choose certain vertices (in the suggested list according to Section 2.4) to find the eigenvalue gradients with respect to. Different approaches include finding the gradients with respect to all the suggested vertices, a selected number, or only a single vertex. Choosing the right vertices for this purpose will have a significant effect on the time required for the optimization procedure. To demonstrate this point, we have selected a Burgers problem on a 2D unstructured mesh with 1100 cells and 5 unstable eigenvalues. The eigenspectrum of the Jacobian matrix is presented in Figure 7a and the unstable eigenmodes are indicated with letters. Following the procedure presented in Section 2.4, 5 vertices are selected to be moved which are indicated in Figure 7b. The gradients of the unstable eigenvalues with respect to the movement of all the suggested vertices simultaneously are presented in Table 1. Table 2 presents the gradients of the unstable eigenvalues with respect to the movement of only vertex a. The eigenvector associated with $\lambda_a = 20.9993 + 0I$ was used to find the vertex a. According to Table 2, the gradients of all eigenvalues other than λ_a with respect to the movement of the vertex a are negligible. This shows that finding the gradient of an eigenvalue with respect to the movement of the associated vertex is adequate for use in the stabilization algorithm. Further, this confirms that the gradients of eigenvalues with respect to non-associated vertices are minute and each unstable eigenmode is local to a certain part of the mesh. Furthermore, comparing the results of Tables 1 and 2 shows that finding the gradient of eigenvalues with respect to the movement of a single vertex gives a very similar result to finding the gradients with respect to the movement of all vertices. We utilize the latter since it is much more computationally efficient and the results can be used for individual vertex movement.



(a) The eigenspectrum of the Jacobian matrix.



(b) The selected vertices to move.

Figure 7: A Burgers problem on a mesh with 1100 control volumes and 5 unstable eigenmodes.

Table 1: The gradient of eigenvalues with respect to the movement of all the suggested vertices, simultaneously.

	Eigenvalue	Gradient w.r.t. x	Gradient w.r.t. y
λ_a	$20.9993 + 0I$	$7987.76 + 0I$	$1213.48 + 0I$
λ_b	$14.6780 + 0I$	$8924.53 + 0I$	$1483.23 + 0I$
λ_c	$8.7831 + 0I$	$7658.86 + 0I$	$2656.88 + 0I$
λ_d	$1.1932 + 4.8081I$	$2260.66 - 3976.23I$	$978.00 - 1734.37I$
λ_d^H	$1.1932 - 4.8081I$	$2260.66 + 3976.23I$	$978.00 + 1734.37I$
λ_e	$0.4764 + 0I$	$437.90 + 0I$	$126.02 + 0I$

Table 2: The gradient of eigenvalues with respect to the movement of vertex a.

	Eigenvalue	Gradient w.r.t. x	Gradient w.r.t. y
λ_a	$20.9993 + 0I$	$7987.75 + 0I$	$1213.66 + 0I$
λ_b	$14.6780 + 0I$	$0 + 0I$	$0 + 0I$
λ_c	$8.7831 + 0I$	$0 + 0I$	$0 + 0I$
λ_d	$1.1932 + 4.8081I$	$0 + 0I$	$0 + 0I$
λ_d^H	$1.1932 - 4.8081I$	$0 + 0I$	$0 + 0I$
λ_e	$0.4764 + 0I$	$-1.54 + 0I$	$0.36 + 0I$

2.6. Mesh Modification

Utilizing the eigenvalue gradient with respect to the mesh movement vector, we can modify the real part of the unstable eigenvalue and push it to the stable part of the eigenspectrum through the method of steepest descent. This methodology also enables us to modify the imaginary part of the eigenvalues for controlled oscillations in the residual history. Refer to [27] for the effects of the imaginary part of the eigenvalue on the residual oscillations. A truncated Taylor series is used for mesh modification and to push the eigenvalue λ to the new value λ_{new},

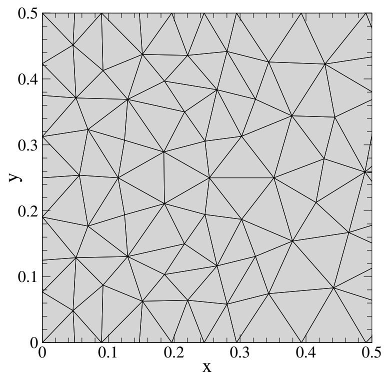
$$\lambda_{\text{new}} = \lambda + \frac{d\lambda}{d\zeta}\delta\zeta \tag{18}$$

The goal is to move the unstable eigenvalues (with positive real parts) to the left side of the eigenspectrum. The dominant eigenmodes (right-most values) dictate the overall behavior of the residual as presented by [27]. As a result, our approach is to move the undesirable eigenmodes of the complex plane to the left side until they are not dominant anymore. One should also pay attention to and account for non-linear effects.

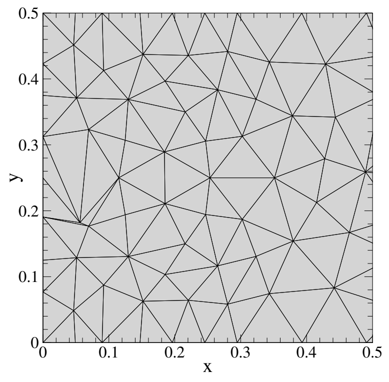
Shifting the eigenvalues as far to the left as possible will result in a more stable numerical solution. However, there should be a limit for mesh vertex movement to prevent the formation of defective cells with negative volumes. A characteristic length scale such as the length of the shortest edge incident on each vertex can be defined and used to limit the vertex movement. This issue is more significant while tackling meshes that contain high aspect-ratio and highly skewed control volumes. Boundary layer and turbulence problems are among these applications in which high aspect-ratio cells are prominent. In such cases, there is a limit on how much a single vertex can be moved before getting a defective control volume. Further, in an adaptive mesh modification approach, error minimization methods are used to modify the elements [31, 32] which might give contradicting vertex movement vectors with the optimization approach. Fortunately, in most problems, movement vectors from mesh optimization are much smaller than those from adaptive methods. This enables us to perform optimization without impeding other modification methods.

For an accurate numerical solution during the mesh optimization process, the boundary geometry must remain intact. For this reason, if a vertex on the boundary is suggested for modification, the movement vector should be restricted to the boundary. This might hinder the effectiveness of a boundary vertex movement in the optimization. However, our approach to finding vertices for modification usually does not return any vertices that are located on the boundaries. The boundary vertices are usually incident on fewer cells than the ones inside the domain which decreases their selection chances. On the other hand, if the suggested vertex is on the boundary and the allowable vertex movement is normal to the optimal gradient (very low probability) our optimization methodology is still applicable with slight modifications. The first solution in such cases is to change the vector direction so that the eigenmodes are pushed to the stable side of the spectrum with possibly different imaginary values. We can also select other vertices to modify as a second solution to such an issue. The presented optimization approach nominates the vertex that has the greatest effect on stability (largest gradient). However, other vertices (usually a local selection) affect the stability of that certain eigenmode as well and can be used for modifications.

Traditionally, different geometric and physical factors are considered in reliable unstructured mesh generation. The best of these techniques are the ones that encapsulate the knowledge and experience of experts in a specific scientific field. An unstructured mesh that is considered “good” for a certain physical problem can be considered “bad” for a different problem in the same domain of solution. Deciding factors include the physical problem, numerical approach (Finite Element, Finite Difference, or Finite Volume), boundary conditions, presence of shocks and other discontinuities inside the domain, and so on. As a result, it is implausible to decide whether a particular mesh is an optimal grid for a specific application solely based on traditional mesh quality measures, such as the minimum and maximum internal angles of a cell, aspect ratio, shortest edge to circumcircle, and Marcum's sliver and skewness measures [33, 34, 35]. Such mesh quality measures can sometimes be deceptive, with meshes considered “good” leading to divergence of the solution. We have devised a test of the Burgers problem of Section 2.4 on a 2D unstructured mesh which is generated utilizing the aforementioned quality measures. Using the Crank-Nicolson time-stepping method, the solution on the “high quality” mesh is unstable. However, changing the location of a single vertex exaggeratedly (without taking the geometric quality measures into account), leads to a stable solution. The traditional high quality mesh and the modified version are presented in Figure 8 and the residual history is presented in Figure 9. The highly skewed cell depicted in Figure 8b has an internal angle close to π which is considered extremely defective according to the traditional mesh quality measures. Nonetheless, the latter mesh results in a stable numerical solution.



(a) The traditional high quality mesh.



(b) The low quality mesh.

Figure 8: Two unstructured meshes for the solution of the Burgers problem.

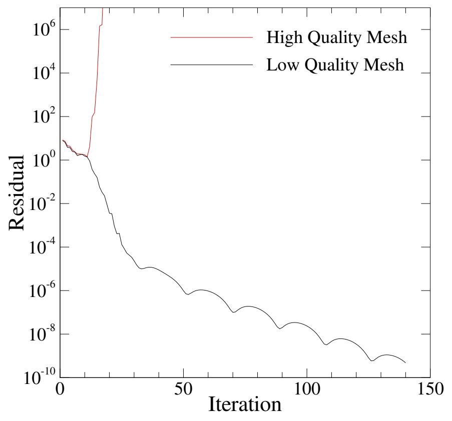


Figure 9: The residual history of the Burgers problem on traditional high quality and low quality meshes.

425 *2.7. Pseudo-Transient Continuation*

Implicit time integration methodologies allow the use of large time steps. A key issue in the implicit upwind method for the solution to the Euler problem is balancing large and small time steps for faster convergence and sufficiently resolving intricate flow features, respectively. Different methods are presented for choosing Courant-Friedrichs-Lewy (CFL) numbers during an implicit solution;

Exponential Progression, Switched Evolution Relaxation, and Residual Difference Method [36]. Further, Newton's method is a famous fixed-point iterative method for the solution of such non-linear problems which results in a quadratic convergence rate [37]. Combining this method with an iterative Krylov solver (e.g. GMRES) for solving the linear system of equations, results in an inexact Newton method [38]. Even in this case, the convergence rate can be super-linear or even quadratic. However, it is established that Newton's method does not converge unless we start from a suitable initial guess. Obtaining the initial guess is known as globalization and can be performed using a continuation method. A global time step or a CFL-based local time step can be utilized to achieve globalization. Implementing a pseudo-time stepping method increases the robustness of globalization. Using a time step that scales with local residual will result in reducing the number of iterations required for convergence [19].

Non-linear problems might exhibit unstable eigenmodes in the early stages of solution due to the initial conditions. The unstable modes resulting from the initial conditions are usually mitigated while using a CFL evolution strategy. However, there can still be other unstable modes near the converged solution that will result in divergence and require stabilization through the methodology discussed herein. In the present paper, the Crank-Nicolson method, as well as the implicit Euler time-stepping with CFL evolution are utilized for time integration in different cases. The results show the robustness of our novel optimization approach in conjunction with multiple implicit methods of time marching.

2.8. Probability of having Unstable Modes

Different problems are tested on numerous meshes to study the probability of unstable modes given certain solver conditions and mesh generation method. We have utilized the Generation and Refinement of Unstructured, Mixed-Element Meshes in Parallel (GRUMMP) software [39] to generate many meshes of various sizes. Different problems were initialized on these meshes and with the help of the stability analysis tool presented herein, the stability of the problems was established. As such, the ratio of unstable cases to stable ones can be determined.

100 different meshes are generated on the domain presented in Figure 2 for the Burgers problem. The number of cells in these meshes ranges from 60 to 125000. Using second-order reconstruction and the boundary conditions presented in the same figure, the number of unstable modes are extracted after one iteration of the solver. Figure 10 shows a histogram of the number of unstable modes present in different cases. Over 80% of the tests in this experiment include unstable modes.

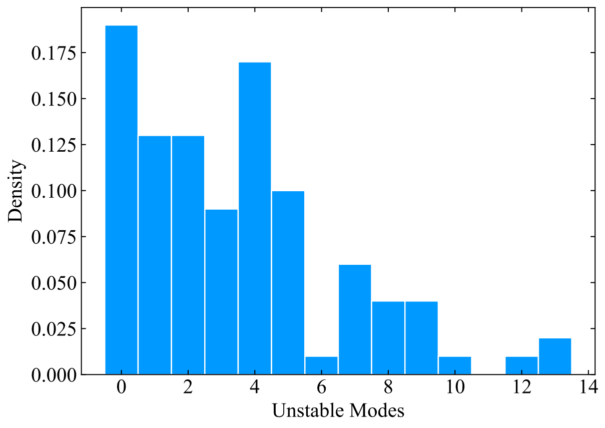


Figure 10: Probability of unstable modes in a second-order Burgers problem.

In another experiment, 100 different meshes are generated around the NACA-0015 airfoil for the Euler problem with the number of cells ranging from 1000 to 50000. Using second-order reconstruction the number of unstable modes are extracted after one iteration of the solver. Figure 11 shows a histogram of the unstable modes present in each test. Over 80% of the cases in this experiment include unstable modes. The Euler problem is non-linear which can exhibit unstable modes resulting from initial conditions as well as undesirable modes near the converged solution. To differentiate between the two, we have utilized our CFL evolution strategy (Section 2.7) to solve the problem on the same 100 meshes. In this test, 23% of the cases exhibited unstable modes near the converged solution. Using a pseudo-transient continuation strategy was to no avail in these cases.

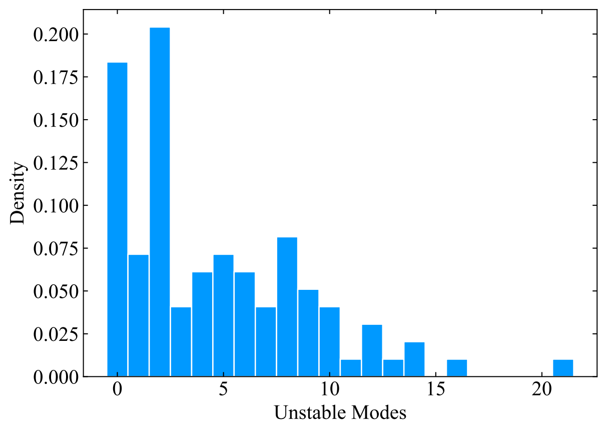
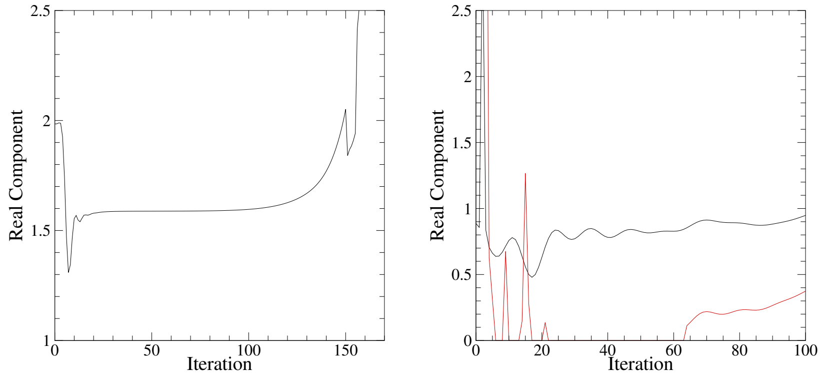


Figure 11: Probability of unstable modes in a second-order Euler problem.

3. Run-Time Mesh Optimization

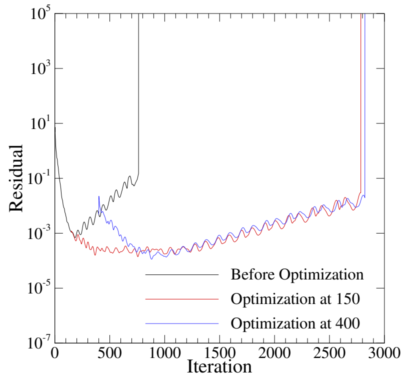
Our optimization scheme is usually capable of stabilizing an originally unstable solution in only one iteration at the beginning of the simulation. However, for non-linear problems the eigenvalues of the Jacobian matrix change as the solution advances in time. As a result, a problem can develop new unstable eigenmodes in the later stages of the solution. Figure 12a shows the real part of the largest eigenvalue of a Burgers problem with solution iteration. Figure 12b depicts the same phenomenon in a 2D Euler problem where eigenmodes are becoming stable and unstable without any specific pattern. This calls for a more flexible stabilization scheme that can cope with such changes in the eigenspectrum during the simulation. Further, there are solutions in which stabilizing unstable eigenmodes result in destabilizing other originally stable eigenvalues with an example Euler problem presented in Figure 13. As seen in Figure 13a, the original solution is unstable and diverges at iteration 760. Pushing the real part of all the unstable eigenvalues to -10 at iterations 150 and 400 produces solutions that are still unstable. This process moves some originally stable eigenvalues to the right side of the spectrum. It is noteworthy that both of the new solutions follow similar trends after iteration 1000 which shows the same new unstable mode in these residual histories. The eigenspectrum of the Jacobian matrix is presented in Figure 13b and depicts the new unstable eigenvalues after the first iteration of the optimizer. Such opposing eigenmodes have deleterious effects on convergence and at first sight, they might seem impossible to remediate. However, in our experience, these new unstable eigenmodes result in instability at a later iteration which can be stabilized then.



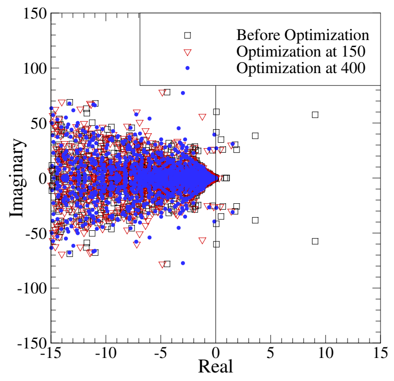
(a) One positive eigenvalue in a 2D Burgers problem.

(b) Two positive eigenvalues in a 2D Euler problem.

Figure 12: The changes in real part of the positive eigenvalues with solution iteration.



(a) The residual history.



(b) The eigenspectrum of the semi-discrete Jacobian matrix.

Figure 13: An Euler problem in which optimizing one eigenmode destabilizes another one.

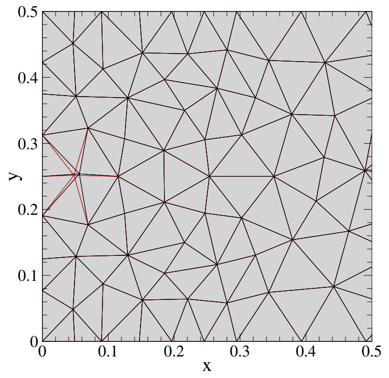
Considering the aforementioned issues, we propose a novel approach for run-time mesh optimization. The idea is to optimize the numerical grid as we progress in the solution. In this outlook, the stabilization program has the state of the solution with an acceptable residual from a few iterations back saved and ready to be called upon when required. Generally, the iteration with the lowest residual works most effectively in the optimization. When divergence occurs, the optimizer performs one iteration of stabilization on the saved state and then starts iterating from there. The stabilization process involves pushing eigenvalues with positive real parts to the left side of the eigenspectrum. This methodology is a robust approach when considering poorly conditioned eigenmodes in an unstable solution and handles the above-mentioned issues very well. This technique can also be effective when applied to slow converging solutions. In such cases, the problematic eigenmodes, which usually have small negative real parts, are identified and pushed further to the left side of the spectrum for faster convergence.

4. Results

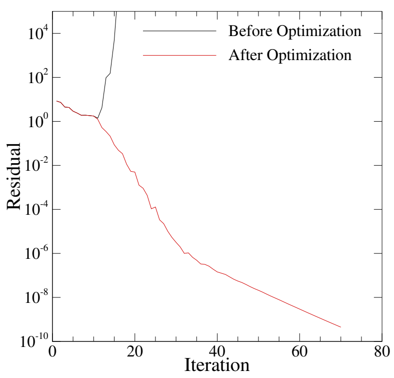
This section provides a deep insight into the results obtained from the novel unstructured mesh optimization application. Mesh modification is utilized on different computational problems, the feasibility of run-time mesh optimization approach is evaluated, and lastly, the improvements in computational time compared with the results of [5] are presented.

4.1. Mesh Optimization

The test case presented in Section 2.6 and Figure 8a is utilized as our initial experiment. An overview of the original high-quality mesh is presented in Figure 14a. The original solution on this mesh is unstable as depicted in Figure 14b. Only one iteration of the optimization program at the beginning of the solution pushes the sole unstable eigenmode to the negative real parts of the spectrum, by moving a single vertex in the mesh, and results in a stable solution. The modified mesh and new residual history are presented in Figures 14a and 14b, respectively. The eigenspectrum of these two solutions are presented in Figure 15. In this test, the only unstable eigenvalue is pushed to $\lambda = -10 + 0I$.



(a) The numerical mesh.



(b) The residual history.

Figure 14: The optimization of a Burgers problem with 500 control volumes.

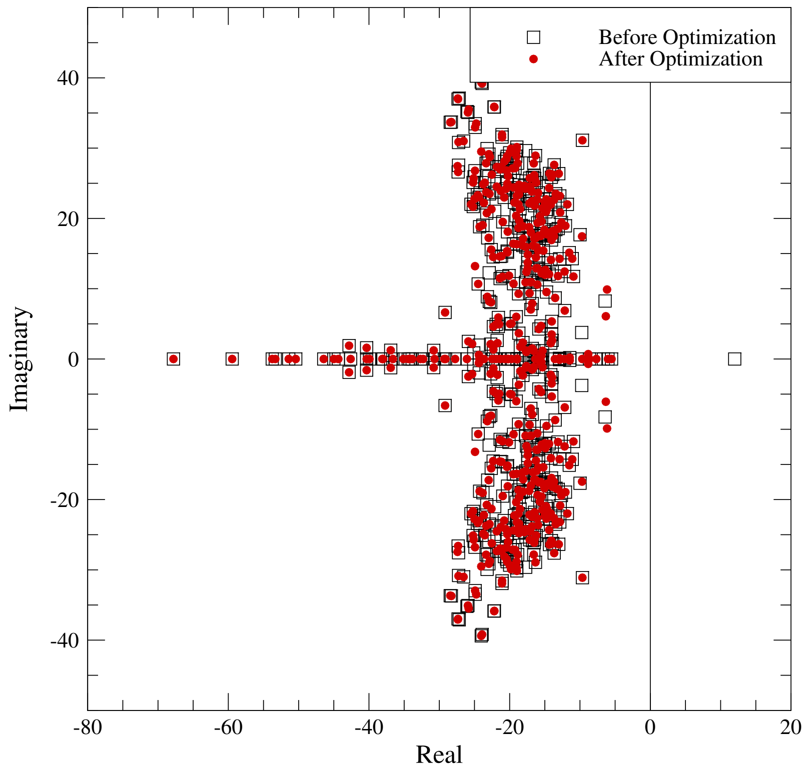


Figure 15: The eigenspectrum of the semi-discrete Jacobian matrix of a Burgers problem with 500 control volumes.

In another experiment, the test case of Figure 6 in [5] is evaluated. This inviscid Burgers problem is solved in a rectangular channel on an unstructured mesh with 1400 control volumes. The eigenspectrum of the original semi-discrete Jacobian matrix is presented in Figure 16. There are two unstable eigenmodes in the spectrum and pushing them to $\lambda = -5 + 0I$ results in a stable solution. The residual history of the solution before and after the first optimization is presented in Figure 17a. The comparison with the results of [5] shows similar convergence behavior with a slight difference that is likely due to the different approach in choosing vertices to move and the general optimization strategy.

Interested readers can refer to [5] for the details of their work. However, the convergence rate is still quite slow and can be improved. The main cause of this behavior is the single eigenvalue with a small negative real part located on the left side of the imaginary axis. Using our optimization program to modify this eigenvalue and push it to $\lambda = -5 + 0I$ in a second optimization step results in a much faster convergence rate which is depicted in Figure 17a. The original and optimized meshes are presented in Figure 17b.

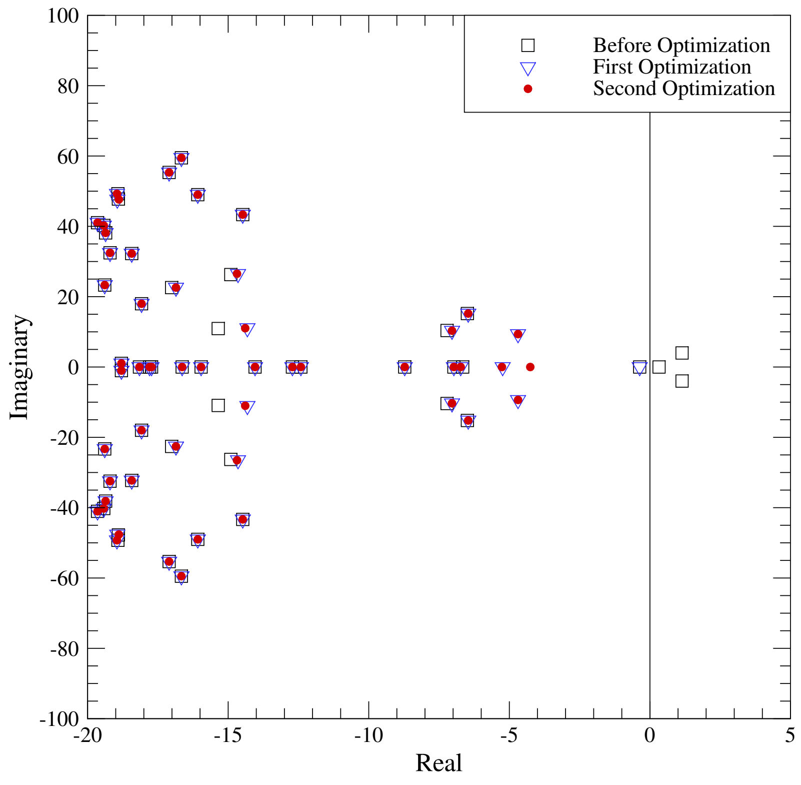
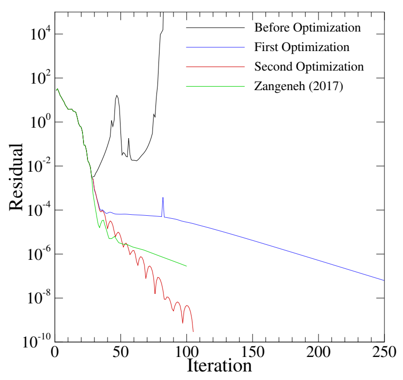
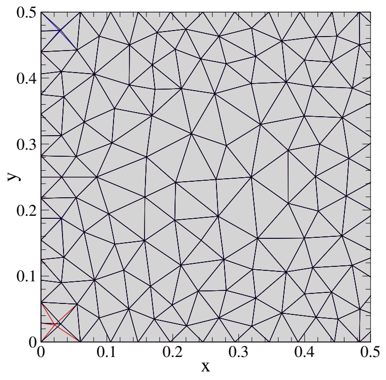


Figure 16: The eigenspectrum of the semi-discrete Jacobian matrix of a Burgers problem with 1400 control volumes.



(a) The residual history.



(b) The numerical mesh.

Figure 17: The optimization of a Burgers problem with 1400 control volumes.

The next experiment is conducted on an Euler problem solved over the NACA-0015 airfoil on an unstructured mesh with 600 control volumes. This problem includes four unknowns, density, momentum in x and y directions, and energy. In the initial solution for this problem throughout the paper, density is set to 1, velocity in (x, y) direction to $M_\infty(\cos(\alpha), \sin(\alpha))$, and pressure to P_{iso}. In these relations, M_∞ is the free stream Mach number, α is the angle of attack, and P_{iso} is the resulting pressure during isotropic expansion to M_∞. The semi-discrete Jacobian contains two unstable eigenmodes which can be seen in Figure 18. The original solution is unstable at iteration 460 using the Crank-Nicolson time-stepping method and a CFL of 1. The optimization at iteration 100 of the Krylov solver pushes the unstable eigenmodes to $\lambda = -10 + 0I$ and results in a stable solution as depicted in the residual history of Figure 19a. The original and modified meshes are presented in Figure 19b. This problem contains unstable modes that are not emerging from the initial conditions. As a result, using the implicit Euler time integration and CFL evolution strategy still results in an unstable behavior as seen in Figure 20a. The application of our optimization approach at the first iteration of the implicit solver gives a stable solution that converges in 7 iterations in total. The original and modified meshes are presented in Figure 20b. Further, in such non-linear problems, significant variations in the state of the solution (e.g. angle of attack, Mach number, and geometry) will cause changes in the eigenspectrum of the Jacobian matrix. As a result, the presented methodology only guarantees local stability and we must perform the eigenanalysis again for large changes in solution state.

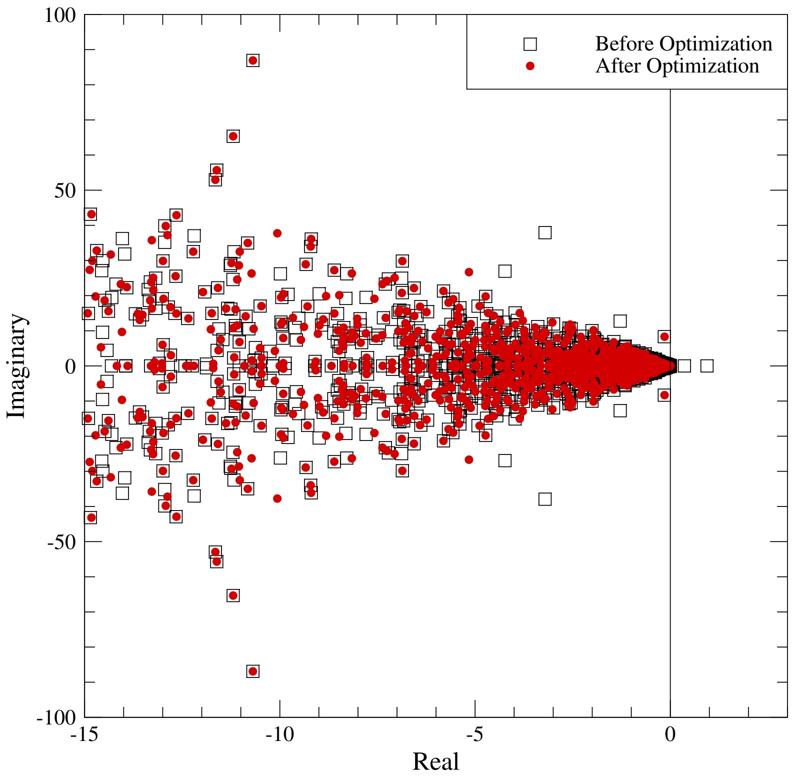
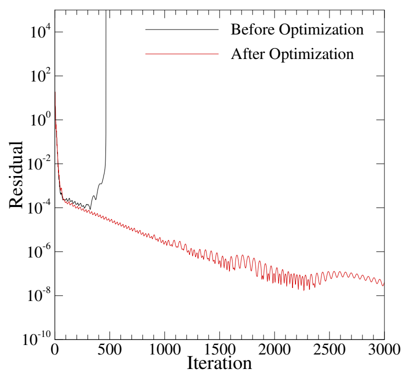
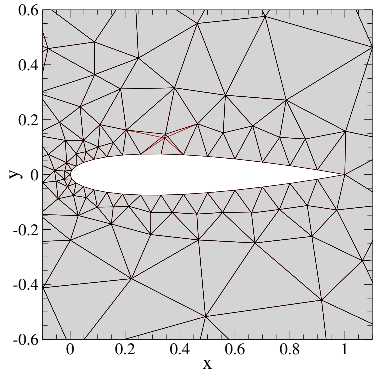


Figure 18: The eigenspectrum of the semi-discrete Jacobian matrix of an Euler problem with 600 control volumes.

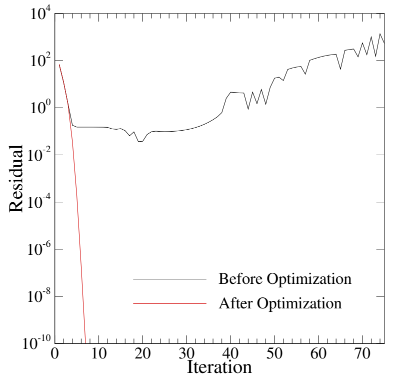


(a) The residual history.

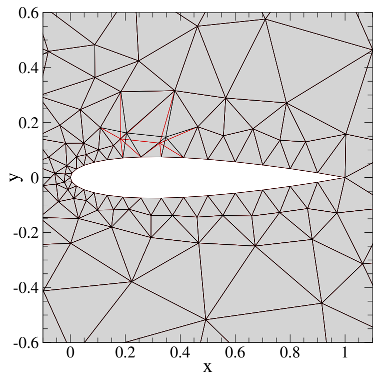


(b) The numerical mesh.

Figure 19: The optimization of an Euler problem with 600 control volumes and Crank-Nicolson time integration.



(a) The residual history.



(b) The numerical mesh.

Figure 20: The optimization of an Euler problem with 600 control volumes and implicit Euler time integration with CFL evolution.

To illustrate the impact of the physical problem on stability, we look at two different problems on the same mesh. A rectangular channel containing an unstructured high-quality mesh with 1100 cells is selected. The Burgers and Advection problems are solved on this mesh and the residual history is presented in Figure 21. The eigenspectrums of these two solutions are presented in Figure

22. As seen, the original Burgers problem has five unstable eigenmodes while the Advection problem does not contain any. This confirms that problem type affects the stability of the solution and should be considered in the optimization scheme. One iteration of the optimizer at the beginning of the solution to the Burgers problem modifies all the unstable modes and gives a stable solution as presented in Figure 21.

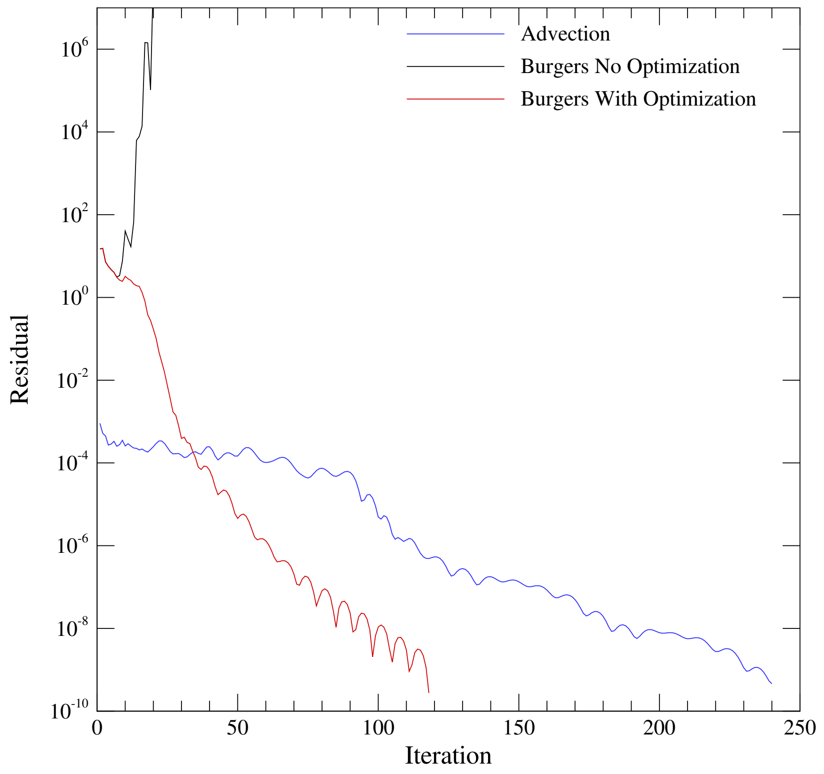
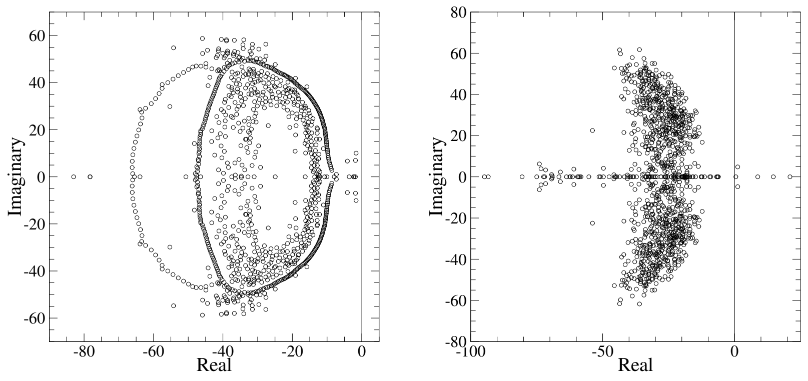


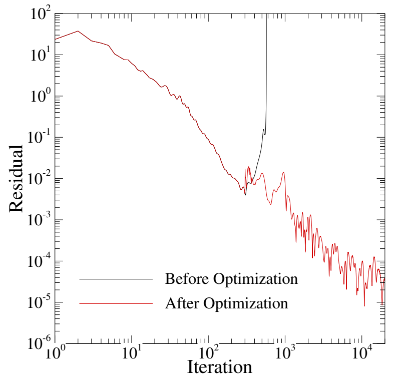
Figure 21: The residual history of two different problems on the same mesh.



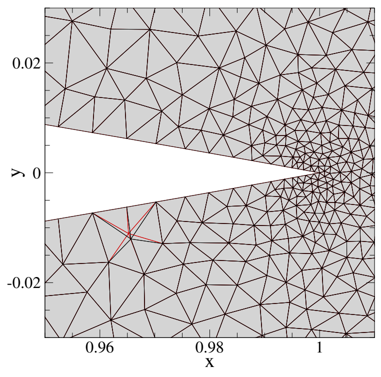
(a) The Advection problem. (b) The Burgers problem.

Figure 22: The eigenspectrum of two problems on the same mesh.

The optimization scheme is also applicable to larger problems with a higher number of control volumes. An Euler problem around the NACA-0015 airfoil, at zero angle of attack and Mach number 0.5, on an unstructured mesh with 7600 control volumes is selected for this experiment. The solution includes four unknowns which makes for 30400 equations in the eigensystem. The original solution is unstable with one eigenmode of $\lambda = 23.7330 + 0I$. Pushing this eigenmode to $\lambda = -100 + 0I$ (for a large vertex perturbation) at the iteration with minimum residual of the original solution (iteration 300) produces a stable solution as depicted in the residual history of Figure 23a. The original and modified meshes are presented in Figure 23b.



(a) The residual history.



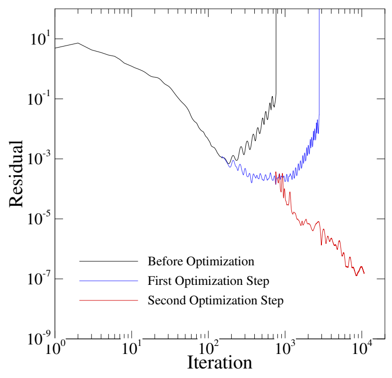
(b) The mesh modification.

Figure 23: The optimization on an Euler problem with 7600 control volumes.

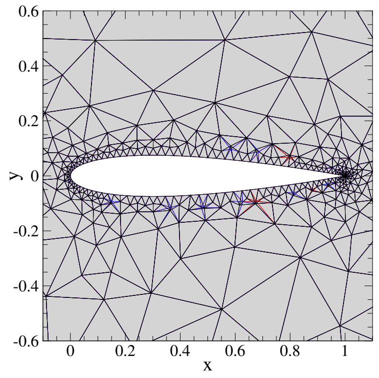
4.2. Run-Time Mesh Optimization

Figure 13 in Section 3 shows an Euler problem in which the application of the stabilization process to push the real part of the unstable eigenvalues to -10 destabilizes other originally stable modes. One might conclude that pushing the unstable eigenmodes to -10 is a sudden large change in the eigenvalues and that might be the cause of the new instabilities. However, moving the unstable modes to a smaller value, such as -1, still produces new instabilities. The main cause of this phenomenon is the opposing eigenmodes in that specific state of the solution. In such cases moving a vertex that stabilizes a certain unstable mode, has deleterious effects on some other originally stable eigenvalue. To overcome these difficulties, the idea of run-time mesh optimization is utilized in which the program has the state of the solution with the lowest residual check pointed. When the solution diverges, we can go back to the saved state and run the optimization program once to push as many positive-real-part eigenvalues to the left side of the spectrum as possible. Going forward, we will save the next minimum residual state and look out for divergence. This approach stabilizes all the cases in our study that contain opposing eigenmodes and prevents unexpected divergence due to the solution-based changes in the eigenspectrum. It also ensures the minimum number of optimization iterations required for a stable solution. Typically, the optimization scheme presented herein is capable of optimizing an unstable solution in only one iteration of the program. For a similar Euler problem, the method presented in [5] takes 34 iterations.

Let us go back to the Euler problem presented in Figure 13 around the NACA-0015 airfoil on an unstructured mesh with 1000 control volumes, which is originally unstable with the lowest residual at iteration 150. Running the optimization program to push the real part of the unstable modes to -10 introduces new unstable eigenvalues. The new minimum residual is at iteration 755 at which performing the second optimization step produces a stable solution with no eigenvalues on the right side of the spectrum. The residual history of the optimized solution and the mesh modification are presented in figures 24a and 24b, respectively.



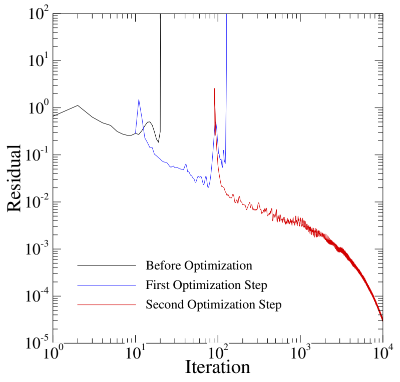
(a) The residual history.



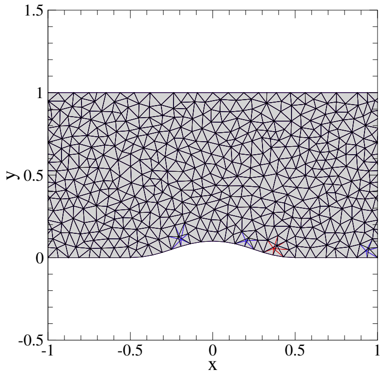
(b) The numerical mesh.

Figure 24: The optimization of an Euler problem with 1000 control volumes.

The Euler problem is solved in a rectangular channel over a bump on a mesh with 1500 triangular cells. The solution using Crank-Nicolson time-stepping method is unstable in only 20 iterations. The optimization program pushes the eigenvalues with positive real parts to the left side of the spectrum and the new solution is stable for about 130 iterations. This problem also includes opposing eigenmodes which means the first stabilization iteration destabilizes some other modes. However, our approach vigorously remediates this behavior and pushes the newly destabilized modes to the left side of the spectrum at iteration 90. The residual history of the optimized solution and the mesh modification are presented in figures 25a and 25b, respectively.



(a) The residual history.



(b) The numerical mesh.

Figure 25: The optimization of an Euler problem with 1500 control volumes.

An issue that might occur for the stabilization of very large problems is the increased computational cost of eigenanalysis which can hinder the optimization process. In such problems, we can utilize the idea of run-time mesh optimization to our benefit. In this approach, we can partially solve the eigenproblem and use any unstable eigenvalues that are found for the optimization. Then, at later stages of the solution, we can solve the eigenproblem again to find the remaining unstable modes. Here, we are not focusing on finding all the unstable modes at once. Instead, we try to find a subset of these modes. In each iteration of the optimization, we find as many unstable modes as computationally feasible which help us spread the workload over multiple iterations of the linear solver.

To depict this approach, we have performed the stabilization on an Euler problem on a mesh around the NACA-0015 airfoil with 29000 cells. The original solution is unstable as depicted in Figure 26. Implicit Euler time integration with CFL evolution is used for the solution of this problem. The first step of the optimization is carried out at iteration 10 of the solution. To stabilize the 2 unstable modes found here, a single vertex is relocated near the trailing edge of the airfoil as depicted in Figure 27a. The new solution is unstable because not all the modes were found and mitigated to save computational cost during eigenanalysis. After 20 more iterations of the solution, the optimization is performed to find another unstable mode. A single vertex near the leading edge of the airfoil is relocated to stabilize this mode as shown in Figure 27b. The solution is stable after 2 iterations of the optimization algorithm as depicted in Figure 26.

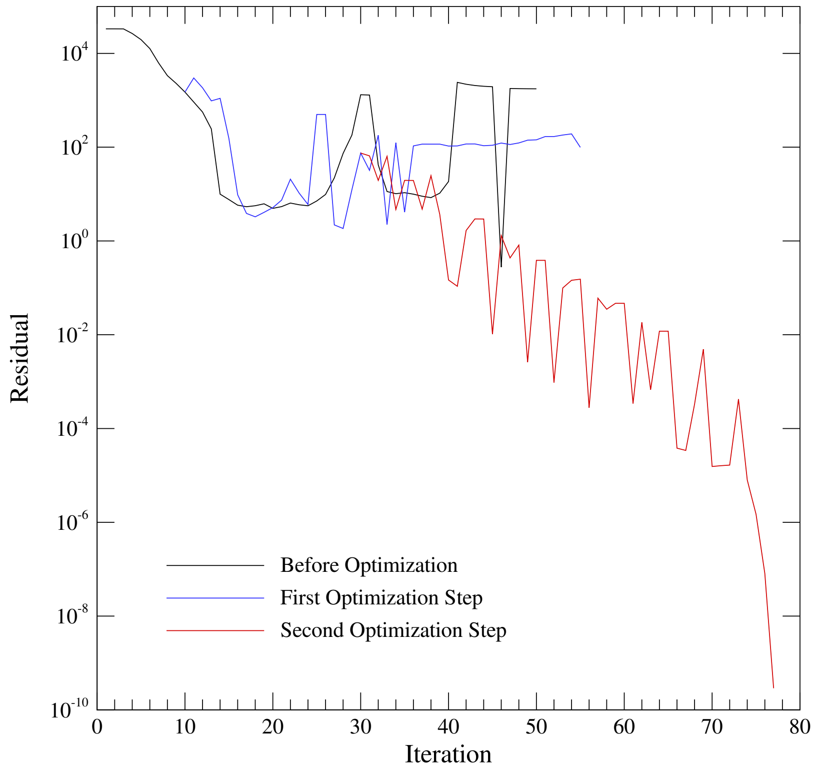
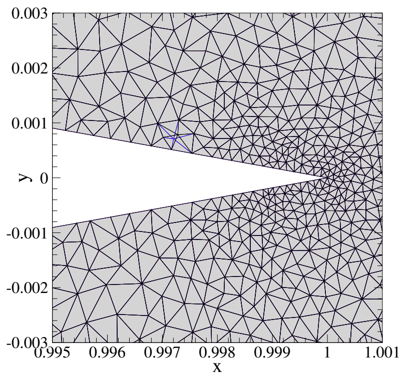
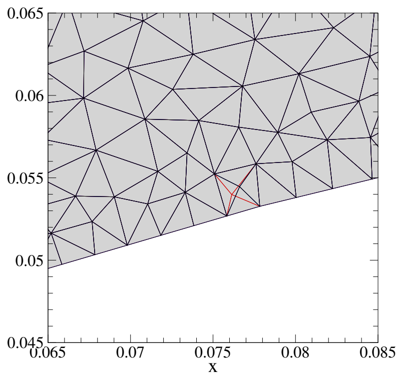


Figure 26: The residual history of an Euler problem with 29000 control volumes.



(a) First vertex relocate.



(b) Second vertex relocate.

Figure 27: The optimization of an Euler problem with 29000 control volumes.

4.3. Computational Cost

The computational cost improvement compared to the previous studies is another contribution of the present paper. The optimization process includes three primary steps which require the most computational time and resources: the solution of the flow field and Jacobian computation, computation of the eigenvalues, and computation of the gradients of eigenvalues with respect to vertex movement. Other less time-consuming steps include finding vertices to move and perturbation of vertices. The time required for the optimization process is evaluated for the Euler problem on meshes of different sizes.

Figure 28 summarizes the computational time of different steps in the optimization algorithm for an Euler problem versus the number of control volumes. The CPU time requirement of the method presented herein is compared to the work of [5]. The time graphs reported in [5] are showing only one iteration of their optimization algorithm. The actual time for their optimization process would be higher than the reported values. For the same unstable problem, their approach will require many optimization iterations (of order 10) in comparison to the approach presented herein which performs the optimization in a single iteration. The CPU time is expected to grow linearly with mesh size as improving the single iteration performance is not in the scope of the present study. While this is true for a single iteration of the optimization algorithm, the overall computational time is reduced substantially by reducing the number of optimization iterations required for a stable problem. Further, the time required for finding the right-most eigenvalue depends on the separation of the eigenvalues in that region. This explains why less time is taken to solve for the right-most eigenvalue in the two meshes of size 800 and 1400 in our experiment. In these two cases, the eigenvalues were better separated by chance which resulted in less computational time for the eigensolution. The time improvement in the present study is due primarily to our improved vertex selection method and a more efficient optimization approach.

To remediate each unstable eigenmode, one vertex is selected to be moved in the present study, while [5] chooses all the vertices in the reconstruction stencil of a cell which on average for a second-order reconstruction are six vertices. The number of vertices in our methodology does not increase with stencil size and it has proved to be just as effective in the optimization process while taking less time to execute. Another aspect of computational time improvement is the overall optimization approach. The present study stabilizes eigenmodes by moving the suggested vertices in a single step, while [5] takes an iteration-based approach. As seen in Figure 28, the time consumption in one iteration of the optimization algorithm of the present study is comparable to [5]. Utilizing our approach in many different cases, this single step is adequate to reach a stable solution. In contrast, similar problems in [5] take more than 30 iterations of the optimization process. Furthermore, the eigensolution for the Euler problem on a mesh with 7600 cells takes about 8.9 seconds while each iteration of the flow solver for the same problem on average takes about 0.7 seconds. As a result, in this case the time required for the solution of the eigenproblem is almost equal to 13 iterations of the flow solver which makes it computationally feasible.

The most time-consuming step of the optimization algorithm is the solution to the large, sparse eigensystem. For a large sparse matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, computing m eigenvalues requires a few considerations regarding both storage and computational cost [25], including the following.

1. Storage of at least m subspace vectors of length n.
2. Orthogonalization of the basis vectors, with a computational cost of $O(m^2n)$.
3. Storage of at least one dense projected eigenproblem of size $m \times m$.
4. The solution to the projected eigenproblem with a computational cost of $O(m^3)$.

Here, the first two points are the most computationally expensive. Requesting the solution to many eigenvalues increases the computational cost and storage requirements of the problem substantially. However, our experience shows that the optimization of unstable problems usually requires the computation of only a few eigenvalues on the right side of the spectrum which are often well separated from the others. This makes finding such eigenvalues of interest less computationally demanding.

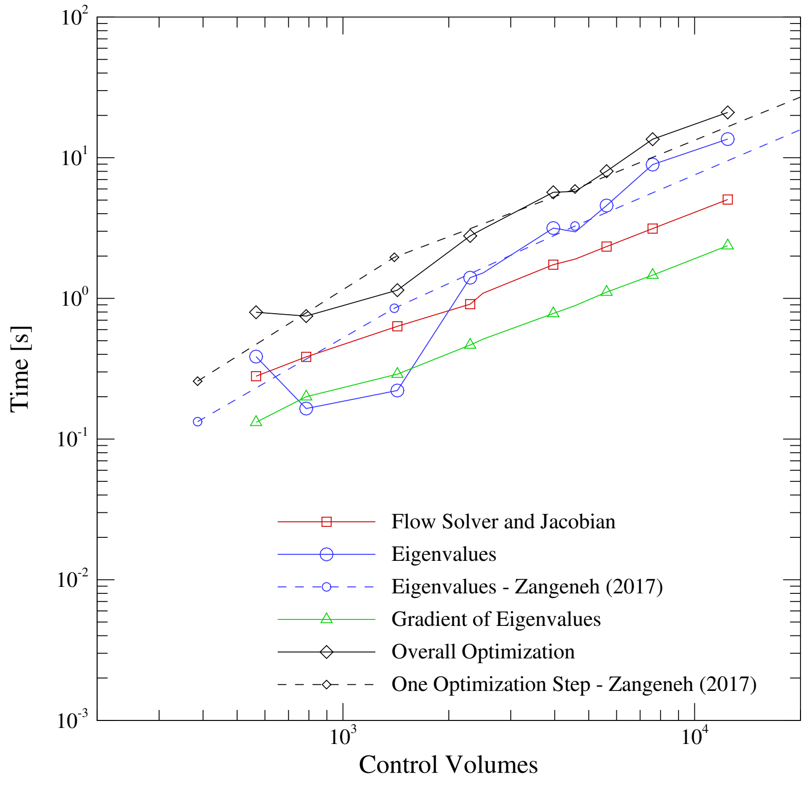


Figure 28: The optimization time of the Euler problem versus mesh size.

For a given problem, the linear solver requires working with much fewer subspace vectors than the eigensolver. Developed by Saad and Schultz [40], GMRES, or generalized minimal residual method, is one of the most well-known iterative methods for solving large, sparse, nonsymmetric systems. This method is used for the solution to the linear system of equations in the present study. The convergence of GMRES closely depends on the distribution of the eigenvalues of the original matrix. At the kth iteration of this method, one matrix-vector multiplication, $k+1$ axpy operations (i.e. $a\boldsymbol{x}+\boldsymbol{y}$), and $k+1$ inner products are required. In total, GMRES requires $2n(\ell+2k+2)$ floating-point operations per iteration and the storage of $k+5$ vectors in addition to the matrix itself. In this equation, the average number of nonzero elements of the matrix is denoted with ℓ which in our case is of $O(1)$.

5. Conclusion

A novel mesh optimization scheme for the solution of different computational fluid dynamics simulations, including the inviscid Burgers and Euler problems, was presented and evaluated in the present study. Utilizing the Lyapunov theorem of stability, we identified the unstable eigenmodes in the semi-discrete Jacobian matrix of the solution. The unstable eigenvectors point to the places in the mesh where that mode is growing. A new approach for selecting vertices to perturb is presented herein. In this methodology, a single vertex is moved to stabilize each unstable mode. The number of selected vertices does not grow with problem or stencil size which helps with the computational resource requirements. Further, using the gradients of the unstable eigenmodes with respect to vertex perturbation, these eigenvalues are pushed to the stable side of the spectrum in one step. This single iteration of the optimization is often enough to stabilize the solution. While stabilizing certain unstable modes in some problems, other modes may become unstable. A new approach for run-time mesh optimization is introduced to remedy such opposing eigenmodes. In this procedure, the optimization is applied at the solution state with the lowest residual when running into divergence. This method has proved to be robust and can remediate the opposing eigenmodes at intermediate iterations of the solution.

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] C. Pain, M. Piggott, A. Goddard, F. Fang, G. Gorman, D. Marshall, M. Eaton, P. Power, C. de Oliveira, Three-dimensional unstructured mesh ocean modelling, Ocean Modelling 10 (1) (2005) 5 – 33. doi:https://doi.org/10.1016/j.ocemod.2004.07.005.

[2] B. Gustafsson, The Godunov-Ryabenkii condition: The beginning of a new stability theory, in: Godunov Methods, Springer, 2001, pp. 425–443.

[3] S. C. Reddy, L. N. Trefethen, Lax-stability of fully discrete spectral methods via stability regions and pseudo-eigenvalues, Computer Methods in Applied Mechanics and Engineering 80 (1-3) (1990) 147–164.

[4] P. D. Lax, R. D. Richtmyer, Survey of the stability of linear finite difference equations, Communications on Pure and Applied Mathematics 9 (2) (1956) 267–293.

[5] R. Zangeneh, C. F. Ollivier-Gooch, Mesh optimization to improve the stability of finite volume methods on unstructured meshes, Computers & Fluids 156 (2017) 590 – 601. doi:https://doi.org/10.1016/j.compfluid.2017.04.020.

[6] R. Zangeneh, C. F. Ollivier-Gooch, Stability analysis and improvement of the solution reconstruction for cell-centered finite volume methods on unstructured meshes, Journal of Computational Physics 393 (2019) 375 – 405. doi:https://doi.org/10.1016/j.jcp.2019.05.002.

[7] R. Zangeneh, C. F. Ollivier-Gooch, Boundary condition optimization to improve the stability of inviscid and compressible finite volume methods on unstructured meshes, Computers & Fluids 199 (2020) 104418. doi:https://doi.org/10.1016/j.compfluid.2019.104418.

[8] R. Richtmyer, L. Bers, R. Courant, Difference Methods for Initial Value Problems: Interscience Tracts in Pure and Applied Mathematics, Literary Licensing, LLC, 2013.

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

[10] M. Giles, Stability analysis of a Galerkin/Runge–Kutta Navier–Stokes discretisation on unstructured tetrahedral grids, Journal of Computational Physics 132 (2) (1997) 201 – 214. `doi:https://doi.org/10.1006/jcph.1996.5616`.

[11] P. Moinier, M. Giles, Stability analysis of preconditioned approximations of the Euler equations on unstructured meshes, Journal of Computational Physics 178 (2) (2002) 498 – 519. `doi:https://doi.org/10.1006/jcph.2002.7038`.

[12] F. Haider, J.-P. Croisille, B. Courbet, Stability analysis of the cell centered finite volume MUSCL method on unstructured grids, Numerische Mathematik 113 (4) (2009) 555–600. `doi:10.1007/s00211-009-0242-6`.

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

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

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

[16] R. Eymard, T. Gallouët, R. Herbin, Finite volume methods, Handbook of Numerical Analysis 7 (2000) 713–1018.

[17] L. Chen, R. Li, An integrated linear reconstruction for finite volume scheme on unstructured grids, Journal of Scientific Computing 68 (3) (2016) 1172–1197.

[18] P. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, Journal of Computational Physics 43 (2) (1981) 357 – 372. `doi:https://doi.org/10.1016/0021-9991(81)90128-5`.

[19] C. Michalak, C. Ollivier-Gooch, Globalized matrix-explicit Newton-GMRES for the high-order accurate solution of the Euler equations, Computers & Fluids 39 (7) (2010) 1156 – 1167. doi:https://doi.org/10.1016/j.compfluid.2010.02.008.

[20] K. Tanaka, T. Hori, H. O. Wang, A multiple Lyapunov function approach to stabilization of fuzzy control systems, IEEE Transactions on Fuzzy Systems 11 (4) (2003) 582–589.

[21] M. Diehl, R. Amrit, J. B. Rawlings, A Lyapunov function for economic optimizing model predictive control, IEEE Transactions on Automatic Control 56 (3) (2010) 703–707.

[22] M. Sharbatdar, C. F. Ollivier Gooch, Eigenanalysis of truncation and discretization error on unstructured meshes, in: 21st AIAA Computational Fluid Dynamics Conference, 2013, p. 3089. doi:10.2514/6.2013-3089.

[23] A. Jalali, M. Sharbatdar, C. Ollivier-Gooch, Accuracy analysis of unstructured finite volume discretization schemes for diffusive fluxes, Computers & Fluids 101 (2014) 220 – 232. doi:https://doi.org/10.1016/j.compfluid.2014.06.008.

[24] G. W. Stewart, A Krylov-Schur algorithm for large eigenproblems, SIAM Journal on Matrix Analysis and Applications 23 (3) (2002) 601–614. doi:10.1137/S0895479800371529.

[25] V. Hernandez, J. E. Roman, V. Vidal, SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems, ACM Trans. Math. Software 31 (3) (2005) 351–362.

[26] S. Abhyankar, J. Brown, E. M. Constantinescu, D. Ghosh, B. F. Smith, H. Zhang, PETSc/TS: A modern scalable ODE/DAE solver library (2018). arXiv:1806.01437.

[27] L. Chen, Stability analysis and stabilization of unstructured finite volume method, Ph.D. thesis, University of British Columbia (2016). doi:http://dx.doi.org/10.14288/1.0300002.

[28] J. N. Juang, P. Ghaemmaghami, K. B. Lim, Eigenvalue and eigenvector derivatives of a non-defective matrix, Journal of Guidance, Control, and Dynamics 12 (4) (1989) 480–486.

[29] D. V. Murthy, R. T. Haftka, Derivatives of eigenvalues and eigenvectors of a general complex matrix, International Journal for Numerical Methods in Engineering 26 (2) (1988) 293–311.

[30] C. S. Rudisill, Derivatives of eigenvalues and eigenvectors for a general matrix, AIAA Journal 12 (5) (1974) 721–722.

[31] M. J. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, Journal of Computational Physics 82 (1) (1989) 64–84.

[32] P. MacNeice, K. M. Olson, C. Mobarry, R. De Fainchtein, C. Packer, PARAMESH: A parallel adaptive mesh refinement community toolkit, Computer Physics Communications 126 (3) (2000) 330–354.

[33] P. M. Knupp, Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part II–A framework for volume mesh optimization and the condition number of the Jacobian matrix, International Journal for Numerical Methods in Engineering 48 (8) (2000) 1165–1185.

[34] S. Gosselin, C. Ollivier-Gooch, Tetrahedral mesh generation using Delaunay refinement with non-standard quality measures, International Journal for Numerical Methods in Engineering 87 (8) (2011) 795–820. doi:https://doi.org/10.1002/nme.3138.

[35] D. L. Marcum, N. P. Weatherill, Unstructured grid generation using iterative point insertion and local reconnection, AIAA Journal 33 (9) (1995) 1619–1625. doi:10.2514/3.12701.

[36] H. M. Bücker, B. Pollul, A. Rasch, On CFL evolution strategies for implicit upwind methods in linearized Euler equations, International Journal for Numerical Methods in Fluids 59 (1) (2009) 1–18.

[37] D. A. Brown, D. W. Zingg, A monolithic homotopy continuation algorithm with application to computational fluid dynamics, Journal of Computational Physics 321 (2016) 55–75.

[38] R. S. Dembo, S. C. Eisenstat, T. Steihaug, Inexact Newton methods, SIAM Journal on Numerical Analysis 19 (2) (1982) 400–408.

[39] C. Ollivier-Gooch, GRUMMP version 0.7.0 user's guide, Department of Mechanical Engineering, University of British Columbia.

[40] Y. Saad, M. H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal on Scientific and Statistical Computing 7 (3) (1986) 856–869.