MEL2040 Project
Data-Driven Analysis of Fluid Flows

Project Report

Ansh Bansal (B22ME012)

Suralkar Nanasaheb Pranav (B22ME066)

15 April 2024

1 Review of Machine Learning in Fluids

1.1 Introduction

The field of fluid mechanics is rapidly advancing, driven by unprecedented volumes of data from experiments, field measurements, and large-scale simulations at multiple spatiotemporal scales. Machine learning (ML) offers a wealth of techniques to extract information from data that can be translated into knowledge about the underlying fluid mechanics.ML provides a modular and agile modeling framework that can be tailored to address many challenges in fluid mechanics, such as reduced-order modeling, experimental data processing, shape optimization, turbulence closure modeling, and control.

1.2 Overview of Machine Learning in Fluid Mechanics

Semi-supervised learning techniques involving RL and generative ML have been applied toward controlling flows and automating turbulence modeling.When applied to fluid mechanics, regression can be categorized into pure and hybrid approaches, which seek to treat a modeling problem either entirely or partially, respectively.The most popular application of pure regression is the prediction of spatiotemporal dynamics in flow configurations, while inverse modeling and symbolic regression enable deducing unknown parameters and formulate algebraic models, respectively.The most popular application of hybrid regression is turbulence closure modeling.

1.3 Physics-informed machine-learning (PIML)

Physics-informed machine-learning (PIML) [3] enables the integration of domain knowledge with machine learning (ML) algorithms, which results in higher data efficiency and more stable predictions.PIML was first proposed as a framework that can account for physical domain knowledge in every stage of ML for improving predictive accuracy in conditions where data are sparse or insufficient.

1.3.1 Physics-Informed Neural Network (PINNs)

A major development in PIML, which deserves extensive discussion due to its proliferation and influence, is the Physics-Informed Neural Network (PINNs), which was first applied to solving a 1D Burgers equation and later applied to solving the Navier–Stokes equations in a 2D flow over a cylinder.The PINN framework was later extended to specifically tackle several laminar and in compressible fluid flow problems.Later, the PINN framework was demonstrated to solve the incompressible Navier–Stokes equations in turbulent flow conditions.

1.3.2 PDE-preserving Neural Network (PPNN) architecture

The PPNN works by forming residual connections between input velocities downsampled on the right-hand side(RHS) of the Navier–Stokes equations to result in more stable predictions than a vanilla CNN model.The PPNN architecture involves two components: a trainable baseline convolutional residual network and a PDE-preserving part. Both of them are connected in an encoder–decoder fashion.Overall, PPNN (an example of a PIML approach) demonstrates significantly better generalizability, stabilityand robustness in terms of long-term prediction than the baseline (vanilla) ML model.

1.4 Flow Modeling With Machine Learning

ML provides new avenues for dimensionality reduction and reduced-order modeling [2] in fluid mechanics by providing a concise framework that complements and extends existing methodologies.

• Dimensionality reduction involves extracting key features and dominant patterns that may be used as reduced coordinates where the fluid is compactly and efficiently described.

• Reduced-order modeling describes the spatiotemporal evolution of the flow as aparametrized dynamical system, although it may also involve developing a statistical map from parameters to averaged quantities, such as drag.

1.5 Neural network modeling

Over the last three decades, NNs have been used to model dynamical systems and fluid mechanics problems. Early examples include the use of NNs to learn the solutions of ordinary and partial differential equations. In fluid mechanics,NNs were widely used to model heat transfer , turbomachinery , turbulent flows , and other problems in aeronautics.

2 Any New Ideas

2.1 Optimizing Collective Swimming Efficiency via Vortex Exploitation: DRL Approach

Deep reinforcement learning (DRL) can be used to optimize the collective swimming behavior of a group of self-propelled swimmers by leveraging vortices. We can aim to develop a control strategy that allows a group of swimmers to collectively navigate through a fluid environment while minimizing energy consumption and maximizing swimming speed. By understanding and replicating the principles of collective swimming observed in nature, we can develop more efficient and agile underwater vehicles and robotic systems.

How it can be done?

By using a combination of computational fluid dynamics (CFD) simulations and deep reinforcement learning techniques to optimize the swimming behavior of a group of swimmers. We can employ a neural network-based RL agent to learn control policies that dictate the swimmers' actions based on sensory inputs and environmental feedback.

2.2 Bio inspired Sensors Trained via Machine Learning for Fluid Flow Classification

Many organisms in nature possess sophisticated sensory systems that enable them to detect and classify various environmental stimuli, including fluid flows. By mimicking these biological sensors, we can aim to develop artificial sensors with enhanced capabilities for flow classification and recognition. We can employ a bio inspired approach to design artificial sensors that mimic the sensory mechanisms observed in biological organisms. This can be done by utilizing machine learning algorithms to train these sensors to classify different types of fluid flows based on sensory inputs.

How it can be done?

The primary objective is to develop bio inspired sensors capable of accurately distinguishing between different flow patterns, such as laminar flow, turbulent flow, and vortical flow. By training the sensors using labeled flow data, we can aim to enhance their classification performance and robustness in real-world applications.

Possible Application: Development of autonomous underwater vehicles equipped with bio inspired sensors trained via machine learning for real-time detection and classification of underwater currents, enabling more efficient navigation and exploration of marine environments.

2.3 Enhancing Turbulent Flow Prediction with Machine Learning: A Focus on Airfoil Separation

Due to their complex and fluctuating nature, turbulent separated flows over airfoils are a significant challenge for conventional computational modeling techniques. We seek to enhance the power of predictive modeling through machine learning and improve the accuracy of flow predictions. The primary objective is to create prediction models that accurately describe the behaviour of turbulent separated flows over airfoils, including phenomena like flow separation, reattachment, and vortex shedding.

How it can be done?

A data-driven framework comprising of full-field inversion and machine learning can be used to develop predictive capabilities for the modeling of turbulent separated flows over airfoils. This framework is embedded in a traditional RANS solver to improve the applicability of the SpalartAllmaras (SA) turbulence model to strong adverse pressure gradients present in flow past airfoils pre- and post-stall.

Possible Application: The development of autonomous aerial vehicles equipped with machine-learning-augmented predictive modeling systems for real-time adaptation to turbulent airflow conditions, enabling enhanced stability and maneuverability during flight.

2.4 Efficient Modeling of Mixing Layers via Cluster-Based Reduced-Order Techniques

The primary objective is to develop reduced-order models that accurately capture the essential dynamics of the mixing layer while reducing the computational complexity associated with high-fidelity simulations. By identifying coherent structures and extracting dominant flow modes, the approach aims to provide insights into the underlying flow physics and facilitate efficient flow prediction

How it can be done?

We can propose a cluster-based reduced-order modeling framework that leverages proper orthogonal decomposition (POD) and dynamic mode decomposition (DMD) techniques. The mixing layer flow data is clustered into coherent structures, and reduced-order models are constructed based on the dominant flow modes identified through POD and DMD analysis.

Possible Application: Engineering applications including turbulent combustion, jet propulsion, and environmental pollution dispersion depend on understanding and predicting mixing layer dynamics.

3 Proper Orthogonal Decomposition

3.1 Image Generation

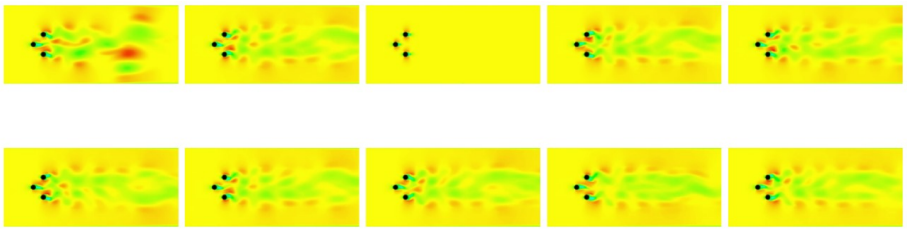


Figure 1: Frames Obtained From Flow Video

3.2 Execute POD

Mathematical Formulation Used :

$$\text{images} = \begin{bmatrix} px & px & .. \\ px & px & .. \\ ..... & & \end{bmatrix}$$

$$\text{mean} = \frac{\sum imagepixels}{\text{total number of pixels}}$$

fluctuation matrix(F) = image pixels - mean

$F = U \sum V^T$

where:

- U is an m×m orthogonal matrix containing the left singular vectors, also known as spatial modes in the context of POD. These vectors represent spatial patterns or structures in the data.
- $\sum$ is an m×n diagonal matrix containing the singular values in decreasing order along its diagonal. The singular values represent the importance or energy associated with each spatial mode.
- V^T is an n×n orthogonal matrix containing the right singular vectors, also known as temporal modes in the context of POD. These vectors represent temporal variations corresponding to each spatial mode.

Compute the covariance matrix C:

$$C = \frac{1}{m} F^T F$$

Perform eigenvalue decomposition (EVD) on the covariance matrix C to obtain the eigenvalues λ_i and eigenvectors v_i:

$$Cv_i = \lambda_i v_i$$

The left singular vectors U are obtained by normalizing the eigenvectors of C:

$$u_i = \frac{Fv_i}{\lambda_i}$$

where u_i corresponds to the i-th column of U.

The singular values Σ are the square roots of the eigenvalues of C, sorted in descending order:

$$\sigma_i = \sqrt{\lambda_i}$$

The right singular vectors V^T are computed using the following relation:

$$V^T = \frac{1}{\sigma_i} F^T U$$

where V^T contains the temporal modes as its rows.

This completes the mathematical formulation of computing U, Σ, and V^T using Singular Value Decomposition for the Proper Orthogonal Decomposition of image data. These matrices are crucial for representing the dominant spatial and temporal structures in the dataset and for reconstructing the images using a reduced set of modes.

3.3 Analyse POD Modes

In Energy matrix $\sum$ Each diagonal repesents amplitude of the corresponding mode.And energies are in decreasing order along the diagonal of this matrix.First Mode have really high energy and higher order modes have less energy.We can see from the following graph.

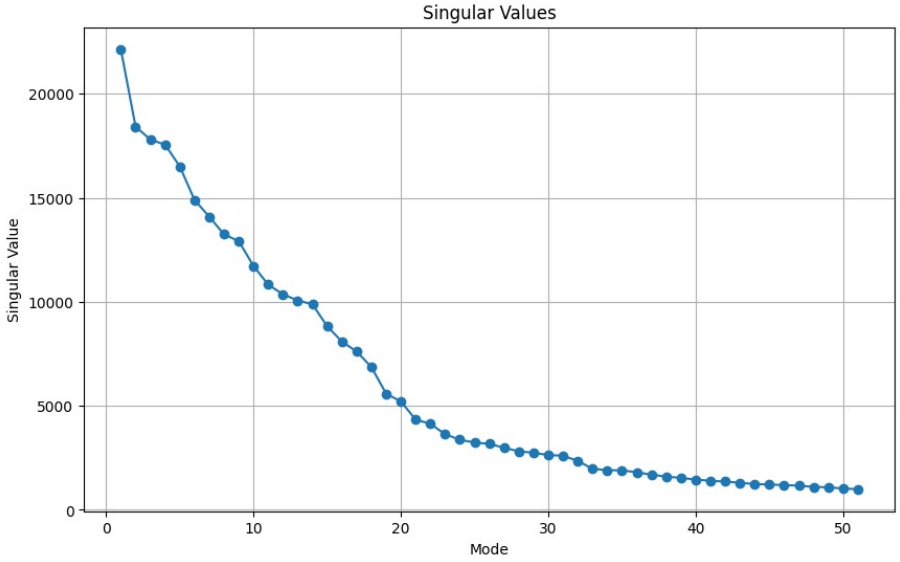


Figure 2: Singular Value v/s Mode

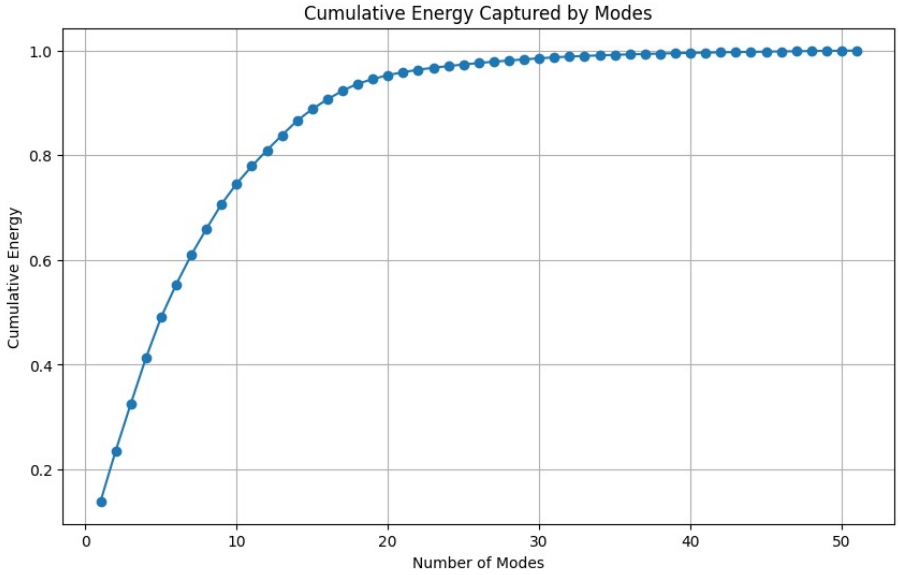


Figure 3: Cumulative Energy v/s Mode

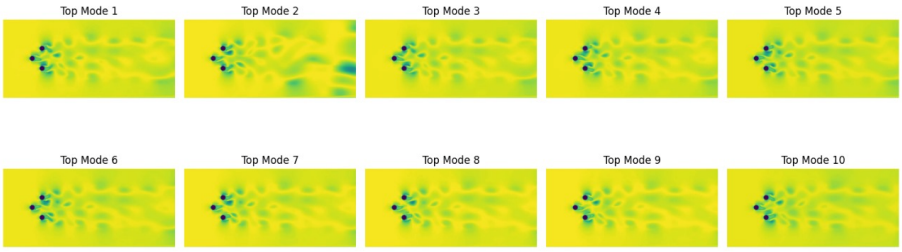


Figure 4: Cumulative Energy v/s Mode

4 Noise!

4.1 Adding Noise

We have chosen Gaussian And Spackle Noise for our image data and applied them on our data with different magnitudes.And after that we have applied POD on these images with different magnitude of noise and got the following results.

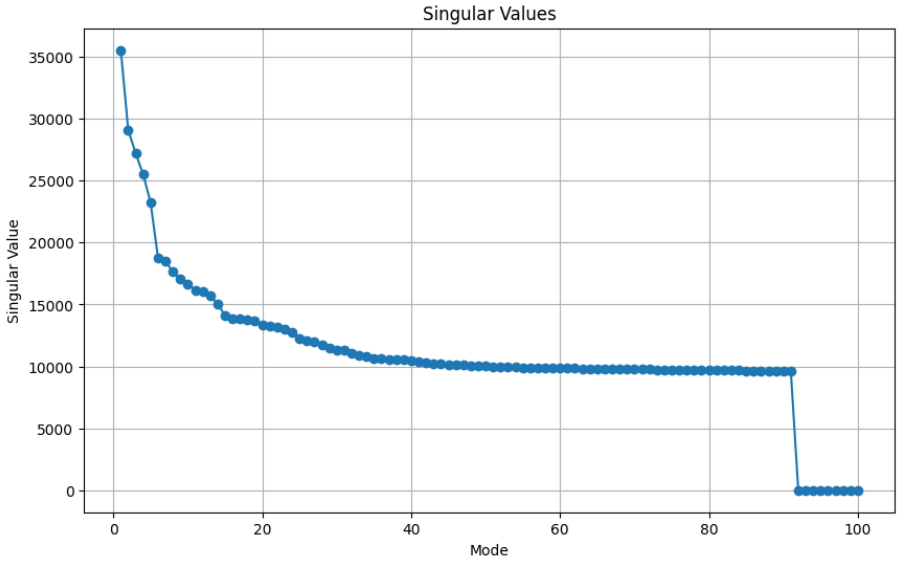


Figure 5: Singular Value v/s Mode: 20% Gaussian Noise

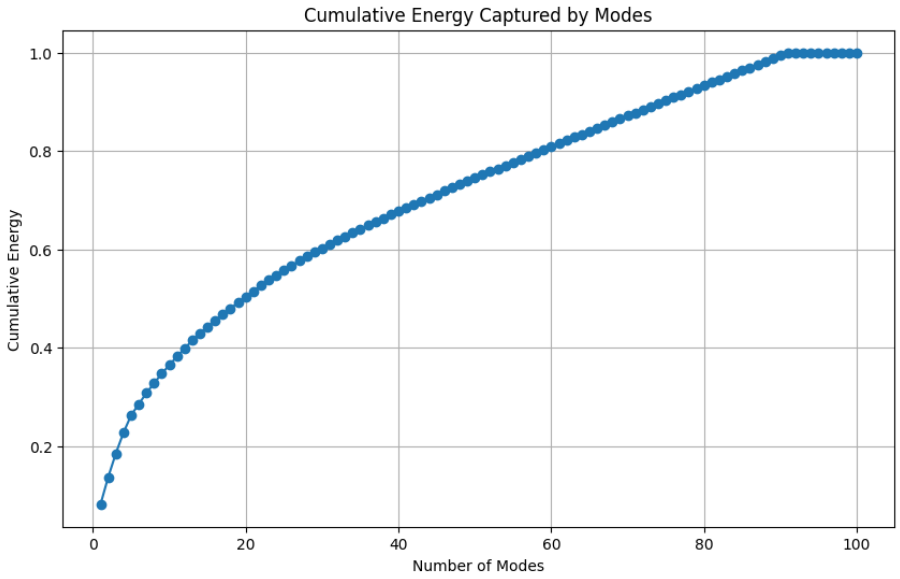


Figure 6: Cumulative Energy v/s Mode: 20% Gaussian Noise

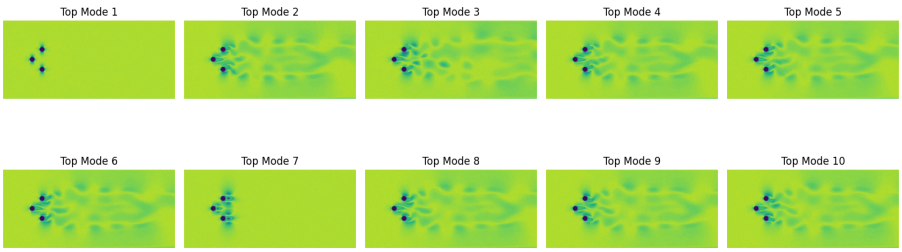


Figure 7: Top 10 modes

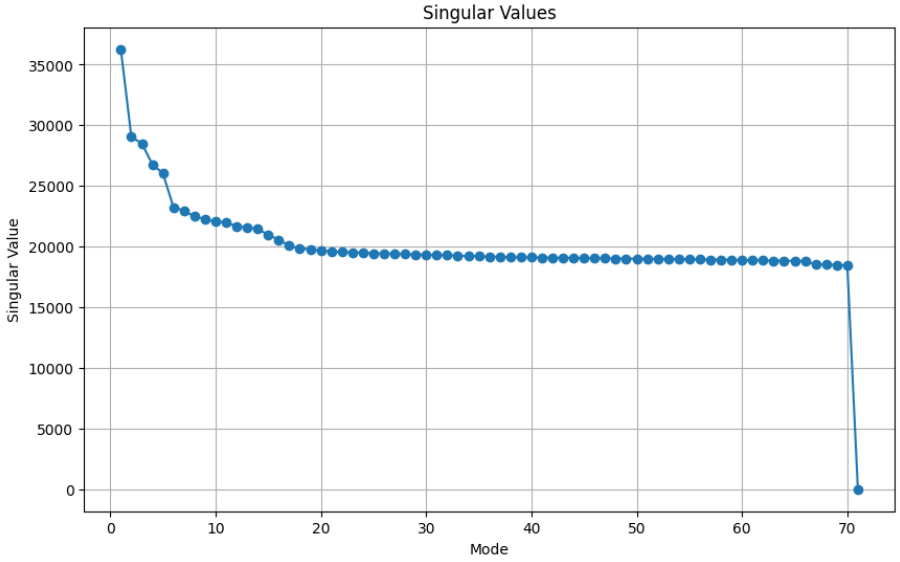


Figure 8: Singular Value v/s Mode: 40% Gaussian Noise

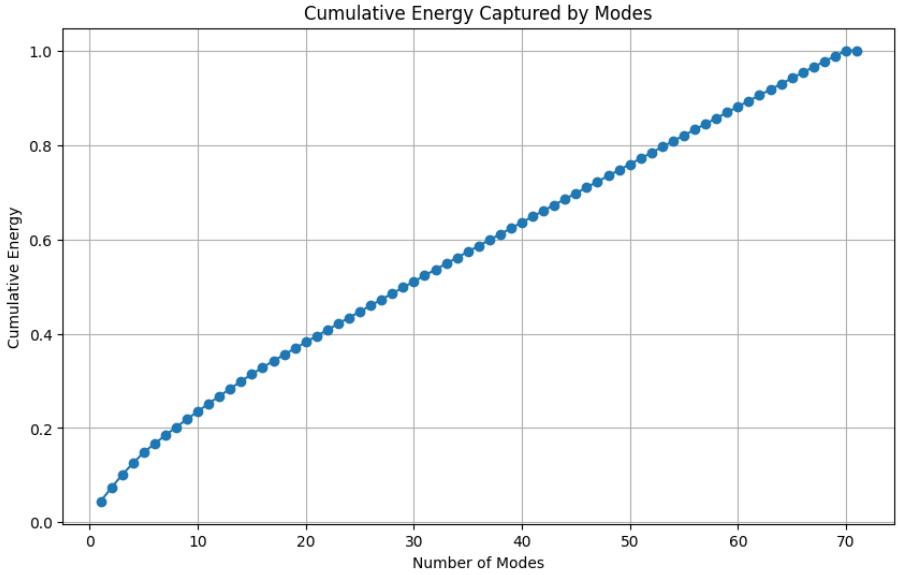


Figure 9: Cumulative Energy v/s Mode: 40% Gaussian Noise

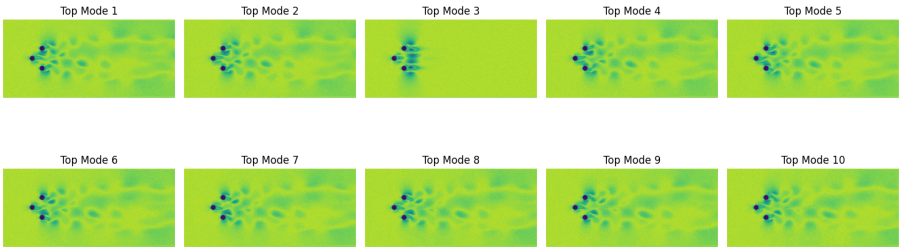


Figure 10: Top 10 modes

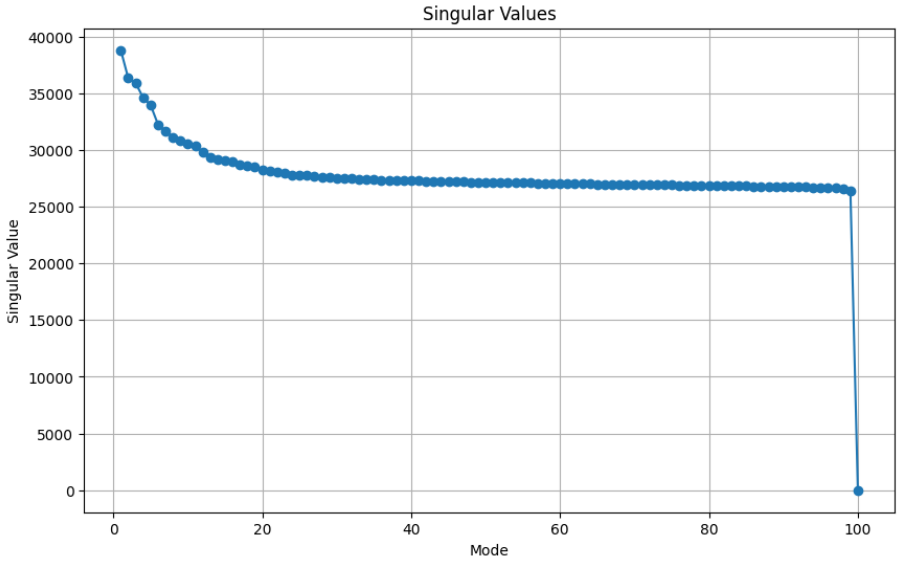


Figure 11: Singular Value v/s Mode: 60% Gaussian Noise

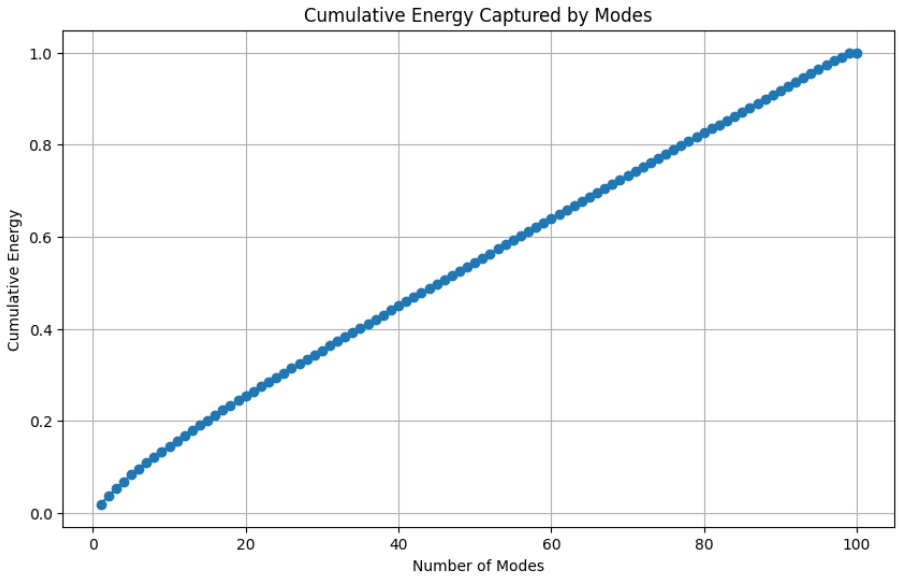


Figure 12: Cumulative Energy v/s Mode: 60% Gaussian Noise

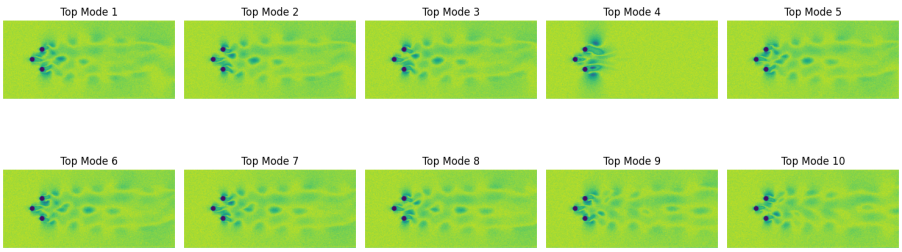


Figure 13: Top 10 modes

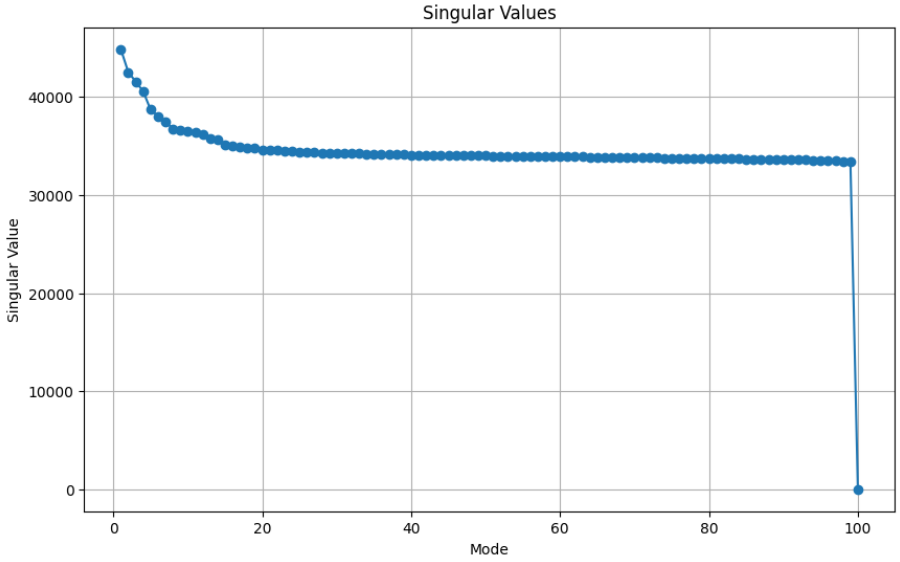


Figure 14: Singular Value v/s Mode: 80% Gaussian Noise

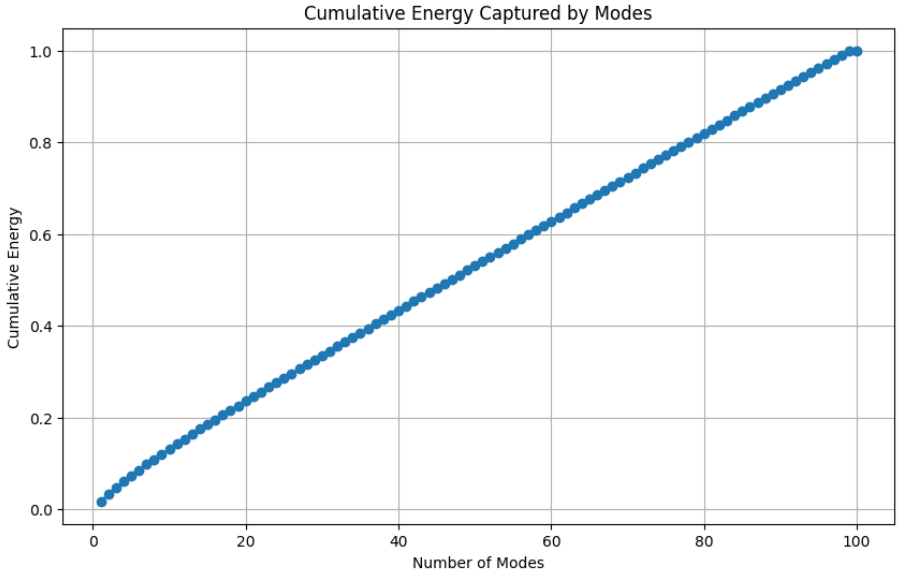


Figure 15: Cumulative Energy v/s Mode: 80% Gaussian Noise

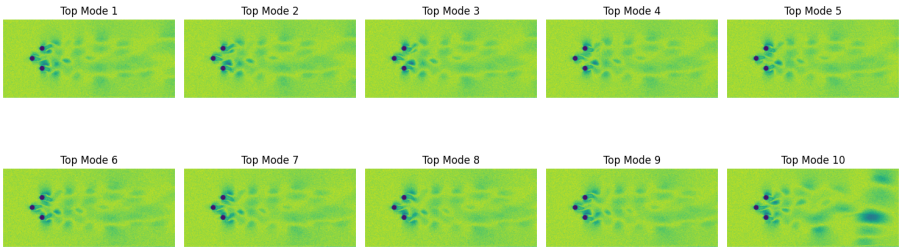


Figure 16: Top 10 modes

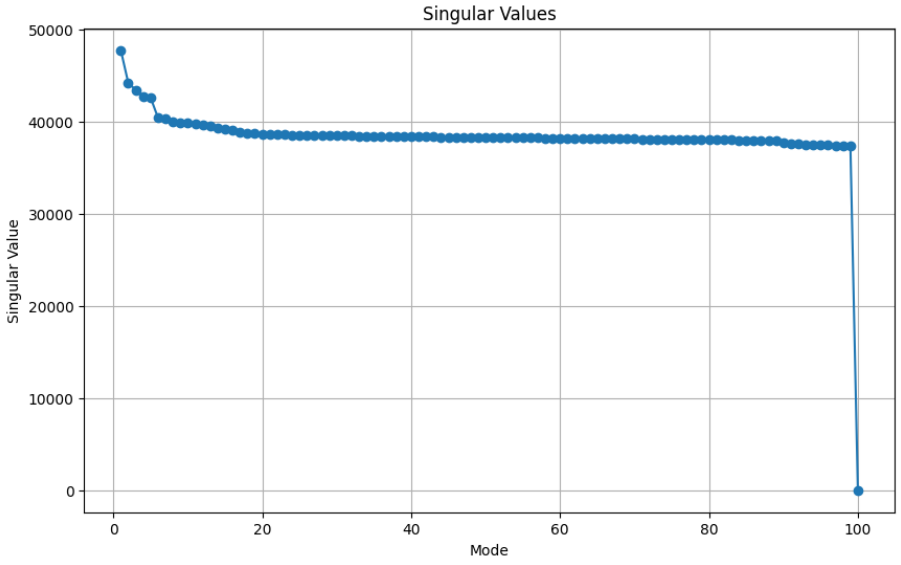


Figure 17: Singular Value v/s Mode: 20% Speckle Noise

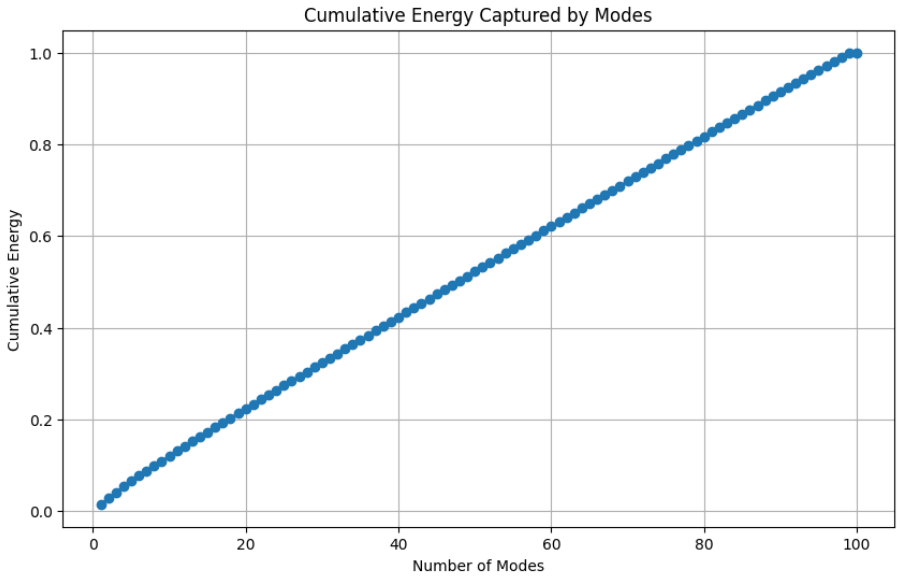


Figure 18: Cumulative Energy v/s Mode: 20% Speckle Noise

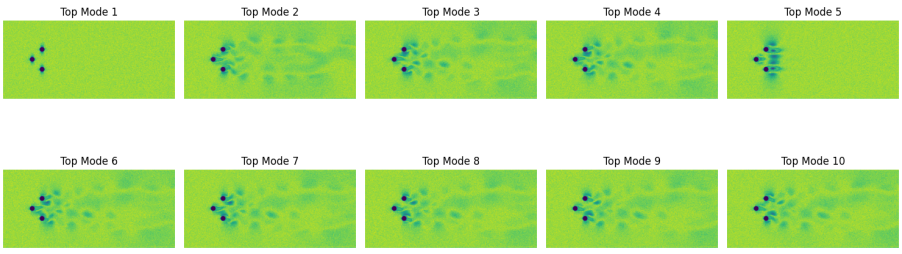


Figure 19: Top 10 modes

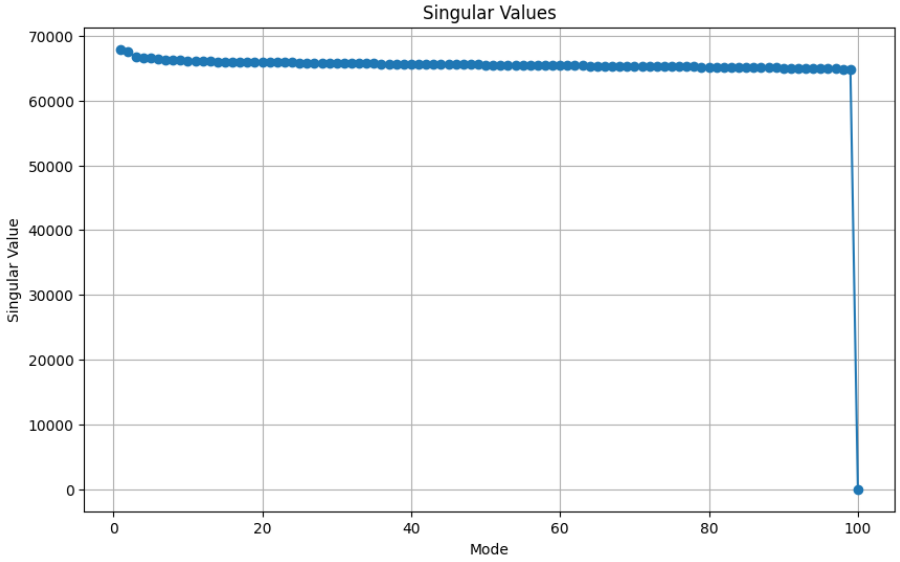


Figure 20: Singular Value v/s Mode: 40% Speckle Noise

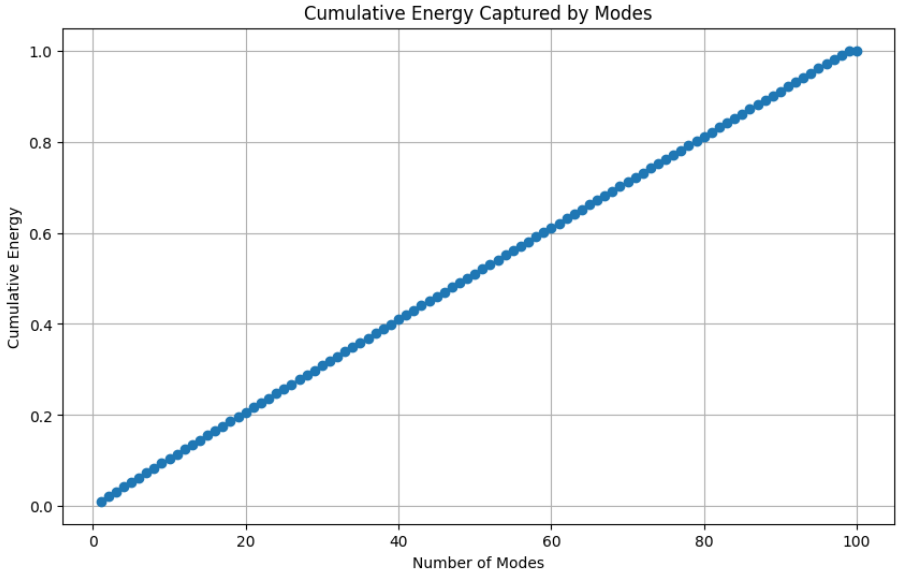


Figure 21: Cumulative Energy v/s Mode: 40% Speckle Noise

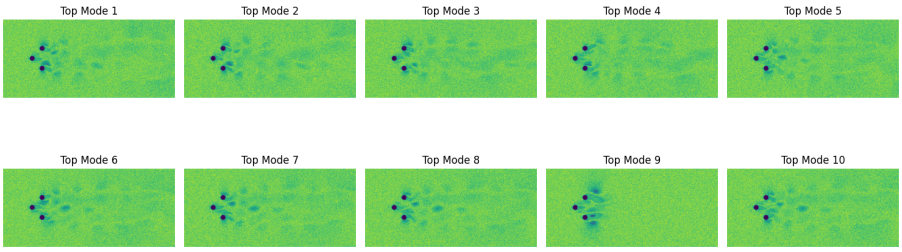


Figure 22: Top 10 modes

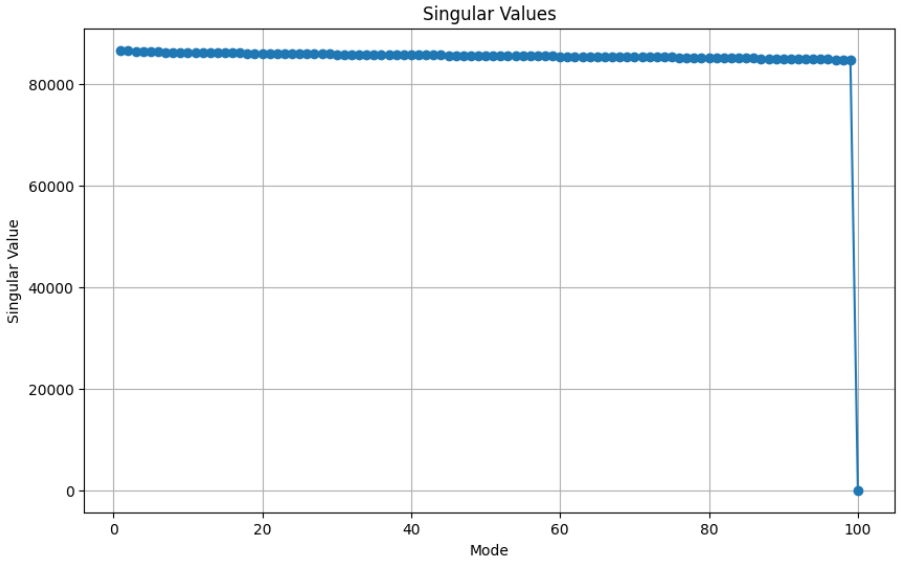


Figure 23: Singular Value v/s Mode: 60% Speckle Noise

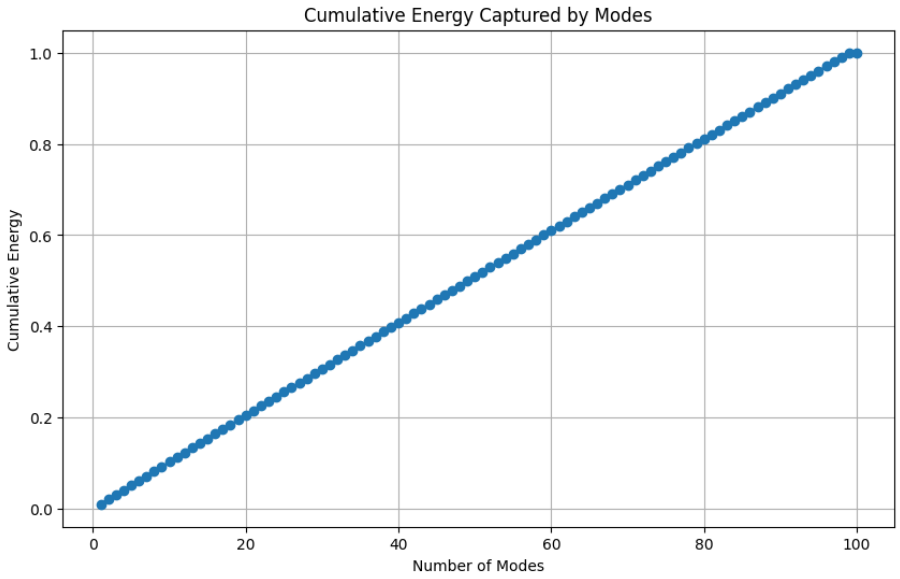


Figure 24: Cumulative Energy v/s Mode: 60% Speckle Noise

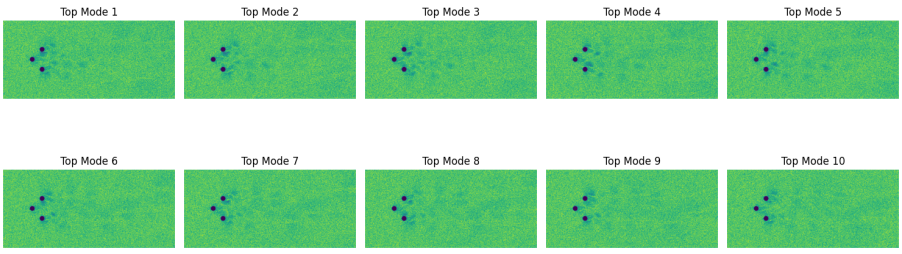


Figure 25: Top 10 modes

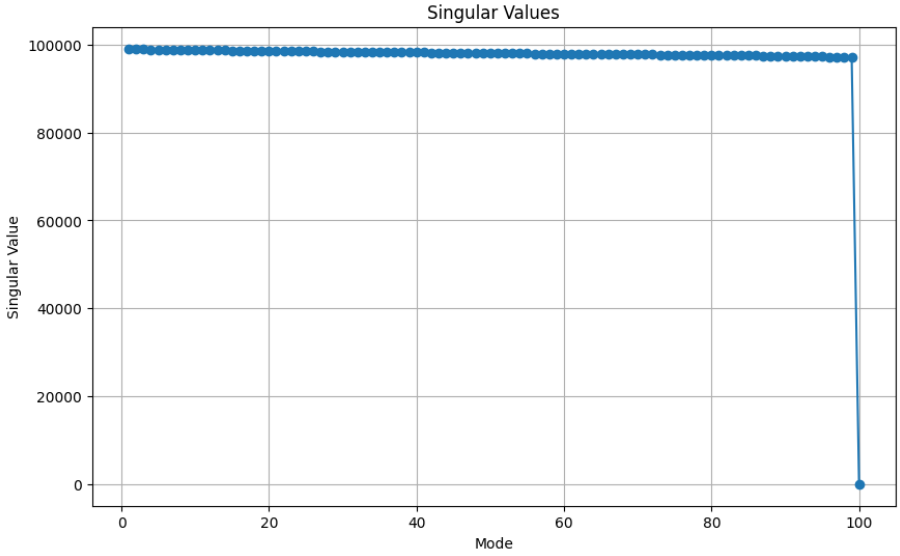


Figure 26: Singular Value v/s Mode: 80% Speckle Noise

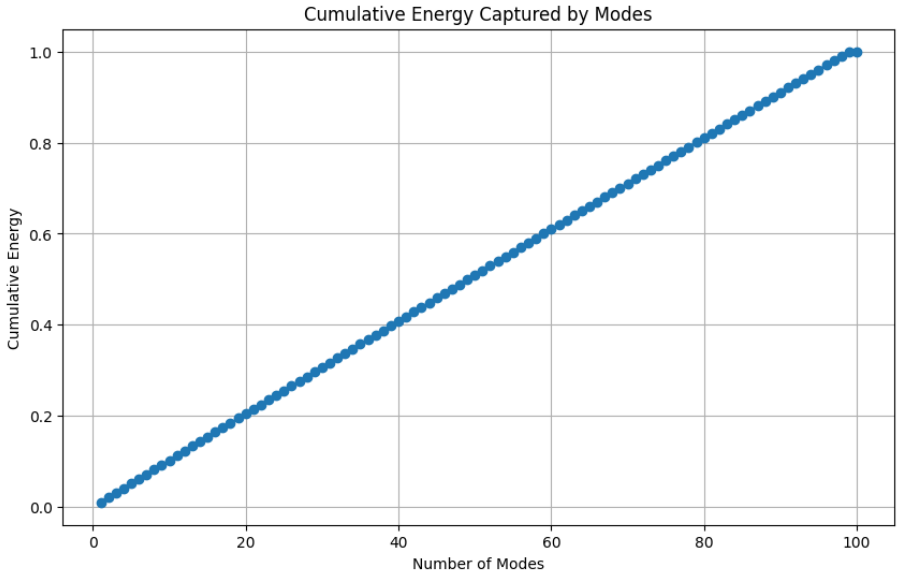


Figure 27: Cumulative Energy v/s Mode: 80% Speckle Noise

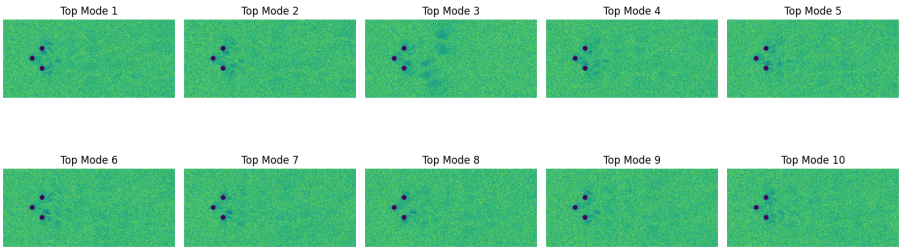


Figure 28: Top 10 modes

4.2 Effect on POD Modes

- How does the energy transcend through modes?

we have added the gaussian and speckle noise. As we add the noise in the original images, the energy of the modes or singular values increases and the energy mode values for our truth lable is less than the energy mode values of gaussian and speckle noise.

- Can you give any statistical evidence of the energy changes in the modes?

```
Top 10 energy states (modes):
Mode 1: Energy = 22140.7194
Mode 2: Energy = 18423.8024
Mode 3: Energy = 17798.6318
Mode 4: Energy = 17555.6327
Mode 5: Energy = 16508.8169
Mode 6: Energy = 14893.5741
Mode 7: Energy = 14091.2135
Mode 8: Energy = 13240.4198
Mode 9: Energy = 12916.0046
Mode 10: Energy = 11721.0856
```

(a) Mode values for truth lable

```
Top 10 energy states (modes):
Mode 1: Energy = 35540.2249
Mode 2: Energy = 29067.7346
Mode 3: Energy = 27210.2095
Mode 4: Energy = 25501.1729
Mode 5: Energy = 23242.0419
Mode 6: Energy = 18737.0176
Mode 7: Energy = 18516.3579
Mode 8: Energy = 17620.0179
Mode 9: Energy = 17061.628
Mode 10: Energy = 16663.2591
```

(b) Mode value for gauss

```
Top 10 energy states (modes):
Mode 1: Energy = 47710.117
Mode 2: Energy = 44159.85
Mode 3: Energy = 43413.3084
Mode 4: Energy = 42714.434
Mode 5: Energy = 42566.749
Mode 6: Energy = 40375.0691
Mode 7: Energy = 40279.347
Mode 8: Energy = 39986.6387
Mode 9: Energy = 39876.2008
Mode 10: Energy = 39814.192
```

(c) Mode value for speckle

Figure 29: Increasing Modes values for different Noises

- Do different noise types have different type of response in terms of how the energy of the modes changes?

Yes, from $fig29$ we can see that on changing noise type our energy mode values also changes and value for energy is less for gaussian type noise but more for speckle noise. from $fig7$ and $fig19$ we can see how the variation in modes comes.

- Do different noise types have different type of response in terms of how the energy of the modes changes?

We can see from $fig5$, $fig8$, $fig11$ and $fig14$ On increasing the magnitude of the noise the value of energy modes or singular value increases.

- What sort of noise is best suited to be used for POD for artificial analysis?

The way in which noise impacts the energy distribution across modes can vary. While speckle noise may introduce localized disturbances that could more noticeably effect certain modes, Gaussian noise tends to be more equally distributed and may not considerably affect certain modes.Because Gaussian noise has a smooth and uniform distribution, it is generally a good option for artificial analysis.

We can see from $fig30$ the different in energy modes of different Noises

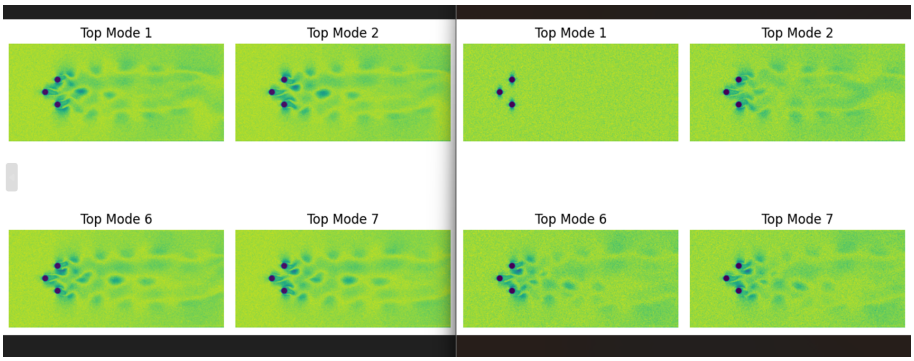


Figure 30: Gaussian And Speckle Energy Modes

5 Super-Resolving

In super resolving method we have used auto-encoders to Denoise images for both Gaussian and speckle noise. An autoen-coder [1] is a type of artificial neural network that aims to learn a representation (encoding) for a set of data. It is the autoencoder that learns noise from training images and then tries to generate a clean image very close to its original input. This autoencoder based proposed model uses convolution layer, convolutional layer and deconvolution layer to defuse the noise. This model mainly consists of input layer, convolution layer, convolutional layers, deconvolution layer and output layer. The several small and linearly connected convolutional denoising autoen-coder (CDA) blocks are used in convolutional layer. CDA learns the degradation from the training images and tries to remove this degradation from input image.

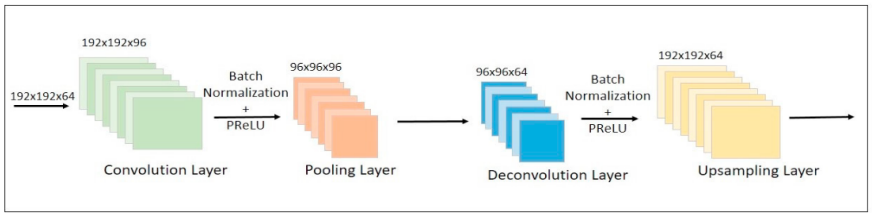
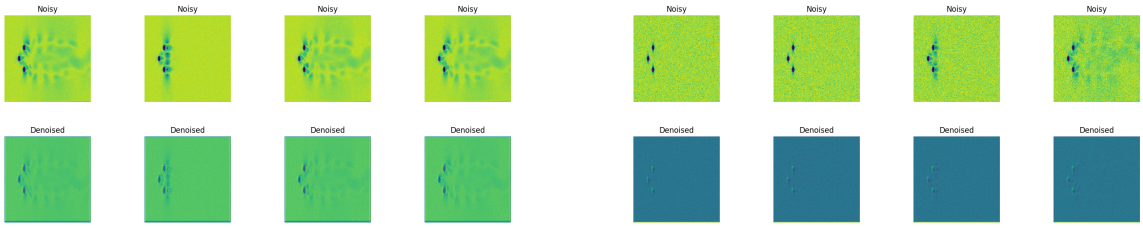


Figure 31: Structure of Convolutional Denoising Autoencoder block



(a) Denoising Gaussian Noise

(b) Denoising Speckle Noise

Figure 32: Denoising Results Using Auto-Encoders

References

[1] Komal Bajaj, Dushyant Kumar Singh, and Mohd Aquib Ansari. "Autoencoders based deep learner for image denoising". In: *Procedia Computer Science* 171 (2020), pp. 1535–1541.

[2] Steven L Brunton, Bernd R Noack, and Petros Koumoutsakos. "Machine learning for fluid mechanics". In: *Annual review of fluid mechanics* 52 (2020), pp. 477–508.

[3] Pushan Sharma et al. "A review of physics-informed machine learning in fluid mechanics". In: *Energies* 16.5 (2023), p. 2343.