TECHNICAL UNIVERSITY OF MUNICH

School of Engineering and Design

Master's Thesis

High-Fidelity Supersonic Flow Field Reconstruction using Conditional Denoising Diffusion Probabilistic Models

---

Author:	Krish Bharadwaj
Matriculation No.:	03763861
Study Program:	M.Sc Aerospace
Chair:	Chair of Aerodynamics and Fluid Mechanics
Supervisor:	Rim Abaidi, M.Sc.
Submission Date:	10 May 2025

Acknowledgement

I sincerely thank my supervisor, Ms. Rim Abaidi, for her immense support and patience throughout this project. Coming into this thesis, I was completely new to the field of generative models and machine learning for fluid dynamics. Rim not only helped me navigate the technical complexities but also provided guidance that shaped the direction of this work. I genuinely feel that I've come out of this thesis with a deeper understanding of generative modelling and hands-on experience in working with large codebases, GPU training workflows, and debugging under pressure, skills I am especially grateful to Rim for helping me develop.

I would also like to thank the Chair of Aerodynamics and Fluid Mechanics at TUM for providing access to the necessary computational infrastructure, including GPU servers, without which this work would not have been possible. Special thanks to Fabian for his assistance in keeping everything running smoothly behind the scenes.

Lastly, I am grateful to my friends and family for supporting me throughout this journey and for patiently tolerating months of rants, late-night frustrations, and the occasional existential conversations!

Abstract

This thesis investigates the application of Conditional Denoising Diffusion Probabilistic Models for reconstructing high-fidelity supersonic flow fields from Schlieren images in atomisation nozzles. While high-fidelity CFD simulations are essential for capturing complex flow phenomena, their high computational cost motivates the development of efficient, data-driven surrogates. This work develops a conditional DDPM framework to reconstruct velocity fields from Schlieren inputs and physical boundary condition labels.

Using a dataset of 5,043 CFD simulations, the model is evaluated under various architectural and training configurations. Cosine-based noise scheduling, FiLM conditioning on physical labels, and a multi-injection strategy for Schlieren conditioning across network depths significantly improve reconstruction quality. Additionally, a hybrid loss function combining Mean Squared Error and Structural Similarity Index Measure with an 80–20 weighting provides a balanced trade-off between pixel-wise accuracy and structural fidelity.

The best-performing DDPM, combining cosine scheduling, FiLM conditioning, multi-resolution Schlieren injection, and hybrid loss, achieves a test SSIM of 0.961 and MSE of 0.004. This DDPM reconstructs detailed flow structures, including shocks and gradients, and demonstrates generalisation across various inlet conditions. Spatial error analysis reveals minor underprediction in flow magnitudes and subtle misalignments at shock interfaces. In this data-limited regime, a supervised U-Net benchmark trained on the same dataset achieves superior metrics (SSIM 0.994, MSE 0.0002), highlighting its advantage for deterministic reconstruction.

The DDPM framework can be readily extended to additional variables such as Mach number and density without architectural changes and produces similar metrics on the test set. The findings position conditional diffusion models as promising generative surrogates for high-fidelity CFD reconstruction, especially in scenarios requiring probabilistic sampling, spatial conditioning, or generalisation across diverse flow regimes.

Statement of Academic Integrity

I,

Last name: Bharadwaj

First name: Krish

Matriculation Number: 03763861

Hereby confirm that the attached thesis, 'High-Fidelity Supersonic Flow Field Reconstruction using Conditional Denoising Diffusion Probabilistic Models' was written independently by me without the use of any sources or aids beyond those cited, and all passages and ideas taken from other sources are indicated in the text and given the corresponding citation. I confirm to respect the "Code of Conduct for Safeguarding Good Academic Practice and Procedures in Cases of Academic Misconduct at Technische Universität München, 2015", as can be read on the website of the Equal Opportunity Office of TUM. Tools provided by the chair and its staff, such as models or programs, are also listed. These tools are the property of the institute or of the individual staff member. I will not use them for any work beyond the attached thesis or make them available to third parties. I agree to the further use of my work and its results (including programs producedand methods used) for research and instructional purposes.

I have not previously submitted this thesis for academic credit.

Munich, 10 May 2025 

Table of Contents

1. Introduction *3*

1.1. Computational Challenges in CFD 3

1.2. Supersonic Atomization Nozzles: Physics and Computational Challenges 5

1.3. Research Motivation 7

2. Literature Review *8*

2.1. Introduction to Machine Learning for CFD Acceleration 8

2.2. Generative Models for Flow Field Reconstruction 10

2.3. Super-Resolution as Image Translation in CFD 15

2.4. Diffusion Models for CFD 17

2.5. Research Gap and Motivation 20

3. Methodology *21*

3.1. Dataset Description 22

3.2. Exploratory Data Analysis 24

3.3. Data Preprocessing 25

3.4. Denoising Diffusion Probabilistic Model 29

3.5. U-Net Architecture Overview 38

3.6. Loss Functions 41

3.7. Evaluation Setup 43

3.8. Training Setup 44

4. Results and Discussion *45*

4.1. Noise Scheduler Comparison 45

4.2. Learned Variance and KL Divergence Loss 47

4.3. Non-Uniform Time Sampling 49

4.4. Dataset Modification: Cropping and Augmentation 51

4.5. Injection Strategies: Enhanced Conditional Guidance 52

4.6. Hybrid Loss Function: 54

4.7. Spatial Error Analysis 59

4.8. Variable Transferability: Predicting Density and Mach Number 60

4.9. Supervised U-Net Benchmark 64

5. Conclusion 69
6. References 71

List of Figures

Figure 1: Turbulence models in CFD (left). Simulated Flow resolution of different models (right)[7] 4
Figure 2: Laval Nozzle Geometry (left) and Flow characteristics in a Laval nozzle under varying back pressure conditions (right) [10] 6
Figure 3: Architecture of a GAN[19] 10
Figure 4: TempoGAN Architecture [20] 11
Figure 5: Comparison of ML techniques for fluid super-resolution[21] 12
Figure 6: VAE Architecture 13
Figure 7: CNN based Super Resolution[29] 15
Figure 8: DDPM Workflow[33] 18
Figure 9: Schematic of the gas atomisation plant[37] 22
Figure 10: Near-field focus region of the simulation domain[38] 23
Figure 11: Preprocessed training sample (Nitrogen). Left: Velocity magnitude field. Right: Corresponding Schlieren image. Conditions: P = 71 bar, T = 309.8 K 28
Figure 12: Preprocessed training sample (Argon). Left: Velocity magnitude field. Right: Corresponding Schlieren image. Conditions: P = 26.5.2 bar, T = 397.7 K 29
Figure 13: Visualising Progressive Noising of the Velocity Magnitude field 30
Figure 14: Evolution of signal and noise weighting terms across 1000 timesteps for Linear, Cosine and Sigmoid Schedules 32
Figure 15: Comparison of Signal retention term across different schedulers 33
Figure 16: Comparison of noised velocity fields at timestep 300 from Standard DDPM and Edge-Aware DDPM 35
Figure 17: Relatively strong alignment between generated and ground truth velocity fields. 46
Figure 18: The model broadly captures the flow structure, but slight deviations are visible. 46
Figure 19: Generated output highlights a failure case of the baseline model. 47
Figure 20: Comparison of a common test sample generated by the baseline model (bottom) and the learned variance model (top). 49
Figure 21: timestep sampling distribution and average loss per timestep at epoch 200. Both curves highlight that early timesteps contribute disproportionately to the total training loss. 50
Figure 22: Cropped dataset generated output vs ground truth velocity 51
Figure 23: Representative examples from the multi-injection model. 54
Figure 24: Examples of velocity fields generated by the best-performing DDPM model. 56
Figure 25: Test set SSIM plotted against temperature for the best-performing DDPM model. 57
Figure 26: Test set SSIM plotted against temperature for the best-performing DDPM model. 58
Figure 27: Example of a low SSIM score prediction. 59
Figure 28: Spatial Error Analysis on select samples. Bright spots indicate higher pixel wise error. 60
Figure 29: DDPM-generated Mach number fields compared to ground truth. 62
Figure 30: DDPM-generated density fields compared to ground truth. 63
Figure 31:Supervised U-Net Velocity Magnitude predictions. 67
Figure 32: PCA projection of feature embeddings extracted from ground truth, DDPM-generated, and supervised U-Net output. 68

List of Tables

Table 1: Data Analysis of Flow Field Variables 24
Table 2: Quantitative comparison of noise schedules on the test set 45
Table 3: Performance comparison between learned variance using the hybrid: MSE + KL Divergence loss and the baseline model 47
Table 4: Quantitative comparison of timestep sampling strategies 49
Table 5: Performance of models trained on modified datasets. 51
Table 6: Performance comparison of conditioning injection strategies 53
Table 7: Performance of different hybrid loss weightings using multi-injection architecture 55
Table 8: Supervised U-Net trained for 100 epochs with SSIM/MSE hybrid loss sweep 64

1. Introduction

High-fidelity Computational Fluid Dynamics (CFD) simulations are invaluable in engineering design and scientific research. Over the years, they have served as the backbone of complex engineering design applications and natural phenomenon predictions. In the aerospace industry, for instance, CFD analyses flow fields around aircraft under various operating conditions, optimising performance and efficiency [1]. Another important example comes from the renewable energy industry. High-fidelity CFD simulations of the flow fields around wind turbines are essential for optimising energy capture and turbine performance [2]. Additionally, CFD plays a critical role in weather forecasting, aiding in disaster management and long-term climate studies [3].

1.1. Computational Challenges in CFD

However, these highly accurate simulations demand significant computational resources and prolonged processing times, creating a need for more efficient methods. DNS resolves all turbulence scales but demands extremely fine grids and time steps. The computational cost of DNS grows proportionally with the cube of the Reynolds number. For instance, a tenfold increase in Reynolds number can require roughly a thousandfold increase in computational cost [4]. Consequently, DNS of high Reynolds-number flows are practically impossible with current computing power.

Even LES**,** which resolves large scales and models only the smallest eddies, remains expensive for many applications[5]. The NASA CFD Vision 2030 study indicates that LES would need radical algorithmic advances to remain relevant for future applications[6]. By 2030, fully resolving wall-bounded turbulence via wall-resolved LES for a complete aircraft is projected to remain infeasible on expected HPC systems. While hybrid RANS-LES approaches can improve tractability, these methods introduce modelling errors and remain under active development.

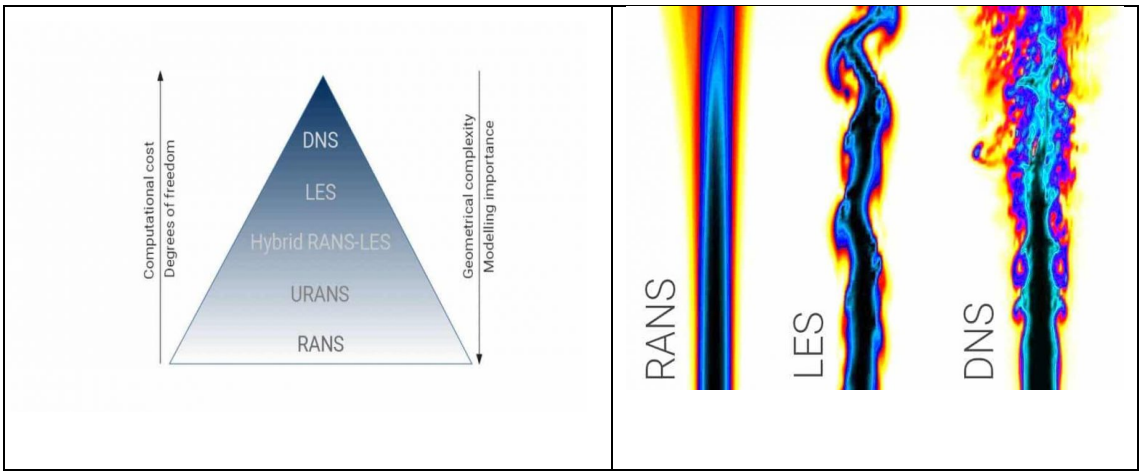


Figure 1: Turbulence models in CFD (left). Simulated Flow resolution of different models (right)[7]

The scaling of traditional CFD solvers on modern high-performance computers poses challenges as well. Legacy CFD codes struggle to efficiently utilise today's massively parallel architectures, limiting their ability to exploit computational resources. Consequently, these solvers cannot take full advantage of large supercomputers, reducing their practical utility. Additionally, bottlenecks in pre- and post-processing further hinder end-to-end CFD workflows.

Industry often relies on RANS simulations, which are far less computationally intense but less predictive for complex flows. The high computational cost of unsteady, high-fidelity simulations like LES or DNS makes their routine use impractical for iterative design processes where numerous configurations must be evaluated.

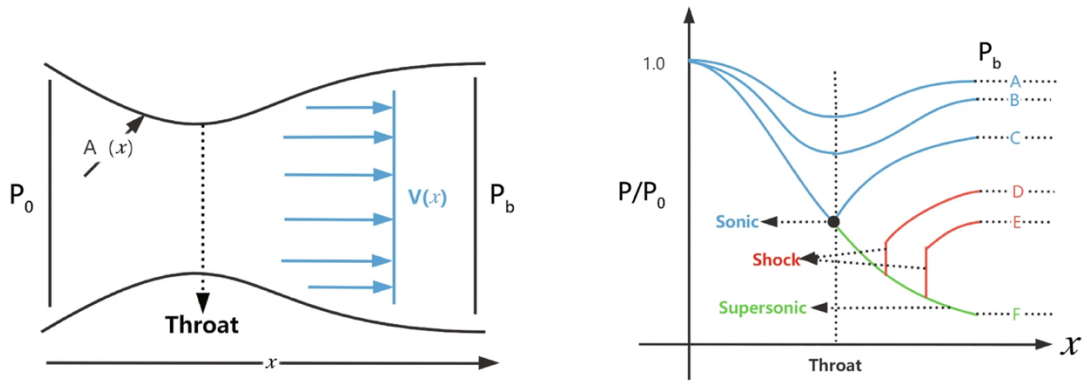
These computational barriers serve as motivation for exploring ML techniques to accelerate CFD. ML could potentially achieve near DNS accuracy at a fraction of the cost by learning important aspects of the modelling process from data. For example, it was demonstrated that an ML-augmented solver could match the accuracy of a standard solver with 8–10x finer resolution in each dimension, leading to speed-ups of roughly 40–80x in turbulent flow simulations [4]. The promise of such substantial speed-ups drives the shift towards data-driven approaches in the field of CFD.

Therefore, ML and deep generative models have emerged as powerful tools in fluid mechanics, offering data-driven alternatives for accelerating DNS and refining turbulence models, among other potential applications[8]. Recent advancements in generative data-driven modelling have opened possibilities and have shown the potential for reconstructing high-fidelity flow fields from sparse data [9].

1.2. Supersonic Atomization Nozzles: Physics and Computational Challenges

A particularly demanding application of CFD involves modelling supersonic flow in atomization nozzles, which are widely used in metallurgy and propulsion systems for applications such as cooling and fuel injection. These nozzles operate under extreme flow conditions, leading to complex phenomena, including compressibility effects, shock formations, and turbulent interactions.

Laval nozzles, also known as converging-diverging nozzles, generate supersonic flow through controlled expansion. A subsonic flow undergoes isentropic expansion as it accelerates through the converging section, reaching sonic conditions (Mach 1) at the throat. Beyond the throat, in the diverging section, the flow continues to expand, increasing in velocity while decreasing in static pressure and temperature, achieving supersonic speeds. However, real-world conditions introduce several complexities, forming distinct shock structures and expansion waves depending on the operating conditions.



(a) (b)

Figure 2: Laval Nozzle Geometry (left) and Flow characteristics in a Laval nozzle under varying back pressure conditions (right) [10]

Figure 2 illustrates the geometry of a Laval nozzle (left) and the influence of back pressure (P_b) on the flow characteristics (right). The back pressure determines whether the expansion is ideal or whether shock waves and expansion fans are developed. When the back pressure matches the design back pressure, the flow expands ideally without experiencing shocks (Curve F). However, deviations from the design back pressure lead to various shock structures [11]:

- Normal Shocks**:** A normal shock can form within the diverging section if the back pressure is sufficiently high. This thin discontinuity causes the supersonic flow to suddenly decelerate to subsonic speeds, resulting in an abrupt rise in static pressure, density, and temperature.
- Oblique Shocks**:** When the supersonic flow encounters a compression corner (such as an overexpanded nozzle experiencing higher-than-design back pressure), it generates an oblique shock wave. These shocks cause a reduction in Mach number and an increase in static pressure and temperature, but more gradually and directionally than normal shocks.
- Mach Discs: In highly under-expanded nozzle flows, where the exit pressure is significantly higher than the ambient pressure, a characteristic shock structure forms downstream of the nozzle exit. This includes a normal shock (Mach disc) perpendicular to the flow, surrounded by reflected oblique shocks forming a barrel-

like structure. The Mach disc causes an abrupt decrease in Mach number and is a critical feature in jet dynamics and mixing processes

- Expansion Fans: When the back pressure is lower than the design pressure, the flow expands beyond the nozzle exit, forming Prandtl-Meyer expansion fans. These occur when supersonic flow turns away from itself at an expansion corner, resulting in smooth, continuous acceleration.

Resolving these intricate flow physics necessitates high-fidelity numerical resolution of the Navier-Stokes equations. Conventional methods, such as DNS or LES, present significant computational challenges due to the fine spatial resolution required to capture shock waves, eddies, and compressibility effects. Even with modern supercomputers, these simulations remain prohibitively expensive for routine design and analysis tasks.

1.3. Research Motivation

The computational bottlenecks associated with high-fidelity CFD simulations of supersonic nozzle flows have motivated this research into data-driven alternatives. Denoising Diffusion Probabilistic Models (DDPMs) have emerged as a promising approach for generating high-quality images across various domains. Their ability to model complex probability distributions makes them particularly well-suited for flow field generation and reconstruction.

Therefore, the central objective of this thesis is to explore and develop Conditional DDPM frameworks specifically tailored to reconstruct high-fidelity supersonic flow fields in atomization nozzles. The research focuses on developing models capable of reconstructing detailed velocity fields conditioned on experimental boundary conditions and Schlieren imaging data.

2. Literature Review

2.1. Introduction to Machine Learning for CFD Acceleration

CFD has become essential in engineering disciplines because it provides detailed insights into complex flow phenomena. However, as explained in Section 1, the high computational cost associated with CFD simulations, particularly high-fidelity methods such as DNS and LES, remains a barrier to widespread and routine application. These simulations require extremely fine grids and small-time steps to accurately resolve turbulence, making them not ideal for iterative design processes, optimisation and real-time predictions.

Traditional approaches to reduce computational cost often involve reduced-order models (ROMs) or regression-based approximations. Techniques such as Proper Orthogonal Decomposition (POD) have historically been used to reduce the dimensionality of flow problems by capturing dominant modes of the flow field [12]. These methods are generally intrusive, requiring explicit integration within the numerical solvers, and often require significant manual tuning to maintain stability and accuracy[13].

Machine Learning offers a non-intrusive and flexible alternative for accelerating CFD simulations. ML-based surrogate models leverage data-driven approaches to approximate complex mappings from low-fidelity or coarse-grained simulations to high-resolution predictions. Once trained, these surrogates deliver accurate results at a fraction of the computational cost of traditional solvers, enabling near real-time analysis and rapid iterative design workflows [14].

Among these, deep neural networks have proven especially powerful. They can learn the complex, nonlinear patterns that exist in turbulent flow fields. In fact, they often perform better than classical techniques like polynomial regression or Gaussian process models. One of their biggest strengths is their ability to take coarse simulation data and learn to

produce high-fidelity flow predictions, making them a practical and scalable way to speed up CFD simulations.

A notable strength of ML-based approaches is their ability to handle high-dimensional data directly. Traditional surrogate modelling techniques are typically limited to predicting global quantities such as lift, drag, or pressure drop. In contrast, modern ML methods, particularly generative models, can reconstruct entire high-dimensional flow fields. Generative models, which learn the probability distributions of complex datasets, enable the generation of multiple physically plausible realisations of flow fields, thus providing richer insights into uncertainties and variabilities inherent in fluid flows.

Recent work demonstrated that CNNs could accurately reconstruct high-fidelity flow fields from significantly coarser simulations, achieving computational speed-ups of 2–10x compared to direct CFD simulations[15]. Their work illustrates a promising strategy wherein ML is combined with CFD solvers to dramatically reduce computational overhead without significant loss of accuracy. Furthermore, researchers have shown that ML-augmented CFD solvers could approximate the accuracy of high-resolution traditional simulations at significantly reduced resolutions, leading to speed-ups of approximately 40–80x [16].

While supervised learning methods such as CNNs have demonstrated significant efficiency gains, they typically produce a single deterministic output for a given input. However, real-world turbulent flows are inherently stochastic, with multiple physically plausible realisations possible even under identical boundary conditions. This intrinsic variability highlights a limitation of purely deterministic surrogates[17]. Generative models have emerged to address this, aiming to model the full probability distribution of possible flow fields rather than a single fixed prediction.

This shift toward data-driven surrogate modelling emphasises generative approaches as particularly advantageous. Unlike deterministic surrogates, generative models can capture the inherent variability and uncertainty in turbulent flows, thereby providing a more comprehensive representation of complex flow physics. Among these generative

approaches, techniques such as Generative Adversarial Networks (GANs), Variational Autoencoders (VAEs), and, more recently, Diffusion Models have gained prominence due to their ability to synthesize realistic and physically consistent flow fields.

The following sections review these generative modelling approaches in detail, highlighting their strengths and limitations and setting the stage for exploring Diffusion Models specifically tailored for flow-field reconstruction tasks in CFD.

2.2. Generative Models for Flow Field Reconstruction

2.2.1. GANs for CFD: Strengths and Limitations

GANs are a class of deep generative models that have gained popularity for generating realistic images, and more recently, for reconstructing complex physical fields in fluid dynamics. A GAN consists of two neural networks trained in opposition: a generator that tries to produce synthetic data and a discriminator that learns to distinguish between real and generated samples[18]. As training progresses, the generator becomes increasingly skilled at producing outputs that resemble real data, ultimately resulting in highly realistic reconstructions. Figure 3 illustrates the architecture of a typical GAN.

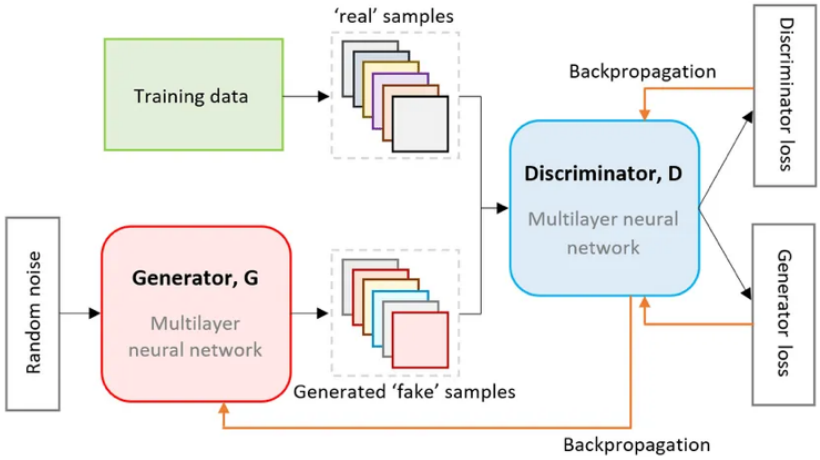


Figure 3: Architecture of a GAN[19]

In the context of CFD, GANs have been explored for tasks like super-resolution, flow field inpainting, and reconstruction from partial observations. A notable example of GAN-based

super-resolution is TempoGAN, which enhances low-resolution 3-D fluid simulations[20]. As illustrated in Figure 4, this approach incorporates two discriminators: a spatial discriminator, ensuring local structure preservation, and a temporal discriminator, enforcing consistency across time. By conditioning on physical flow quantities, such as vorticity fields, the generator reconstructs realistic small-scale vortices and turbulence structures while maintaining temporal coherence.

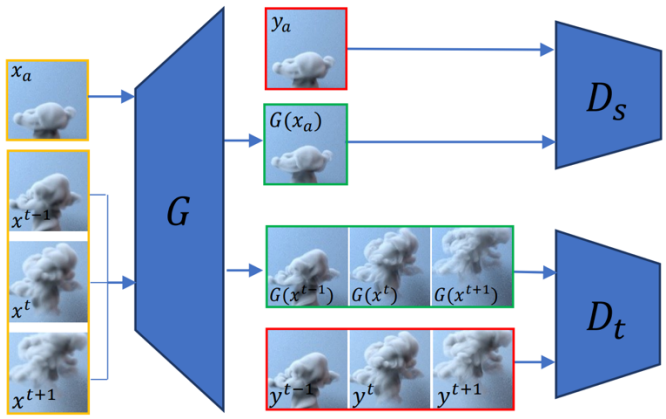


Figure 4: TempoGAN Architecture [20]

Another impactful use of GANs in CFD is super-resolution, which aims to reconstruct high-resolution flow fields from coarse or under-resolved inputs. Conventional CNN-based models often rely on MSE loss, which produces overly smoothed outputs and fails to preserve sharp features such as vortices and shock waves. GANs overcome this limitation by introducing an adversarial loss, encouraging the generation of sharper, more physically realistic flow fields that better reflect the underlying distribution of turbulent structures.

These strengths are supported by comparative studies. For example, one study evaluating velocity component reconstruction in turbulent flows found that GANs outperformed traditional CNNs, particularly in cases where the velocity components were weakly correlated (Figure 5: bottom row) [21]. This highlights the value of GANs in capturing the complex and multi-scale nature of turbulent flows. Figure 5 captures the findings of this comparison.

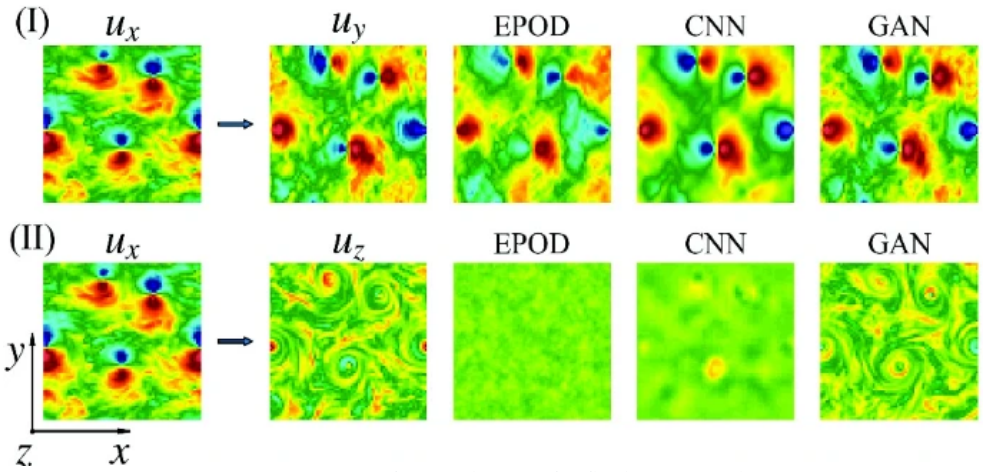


Figure 5: Comparison of ML techniques for fluid super-resolution[21]

GANs have also shown promise in scenarios where data is incomplete, such as flow inpainting. In such cases, GANs learn spatial correlations from the training data and can infer plausible reconstructions in regions with missing or corrupted information. This capability is especially useful in experimental settings or real-time sensing environments, where measurements may be sparse or obstructed [22].

Despite these advantages, GANs come with several limitations. The adversarial training process is notoriously unstable**,** often requiring careful tuning of hyperparameters and network architectures. One common issue is mode collapse [23], where the generator produces only a limited variety of outputs, failing to represent the full diversity of real-world flow data. Furthermore, GAN-generated outputs may violate physical laws, such as mass or momentum conservation, unless these constraints are explicitly enforced through physics-informed loss functions or architectural modifications.

These drawbacks have motivated exploring alternative generative models that offer better training stability, stronger uncertainty modelling, and improved physical consistency.

2.2.2. VAEs for CFD: Strengths and Limitations

VAEs are another class of generative model used in CFD, particularly for dimensionality reduction, flow field reconstruction, and conditional generation. As shown in Figure 6, VAEs consist of an encoder that compresses high-dimensional data into a lower-dimensional

latent space and a decoder that reconstructs the original data from this representation. Unlike standard autoencoders, VAEs impose a probability distribution on the latent space, enabling the generation of new samples by sampling latent vectors and passing them through the decoder.

VAEs have shown promise in applications where learning a compressed, meaningful representation of flow fields is beneficial. For instance, researchers have trained VAEs on snapshots of turbulent velocity fields to interpolate between different flow regimes or generate new realisations for design studies[24]. The ability to condition VAEs on specific input parameters, such as boundary conditions or geometry, has enabled Conditional VAEs to generate customised flow predictions for various physical setups. This makes them useful tools for tasks like turbulent inflow synthesis, reconstruction from sparse sensor measurements, and data-driven optimisation.

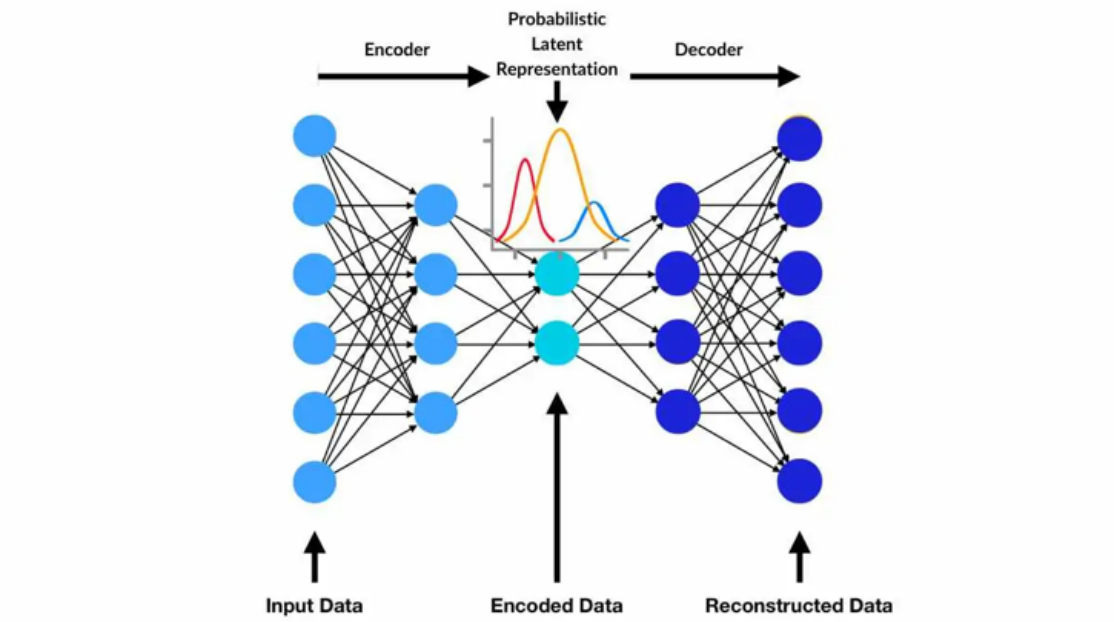


Figure 6: VAE Architecture

One strength of VAEs lies in their probabilistic nature. This allows them to model uncertainty in the data. This is especially useful in turbulence modelling, where flows can exhibit a wide range of possible behaviours. Additionally, the learned latent space can be interpreted and

manipulated for downstream tasks, making VAEs suitable for design space exploration or sensitivity analysis in engineering applications.

VAEs also come with notable limitations, especially when it comes to reconstructing sharp and high-fidelity flow features. The reconstruction process in VAEs typically minimises a pixel-wise loss function (such as MSE), which tends to average over possible solutions and leads to blurry outputs. This smoothness is especially problematic in fluid dynamics, where preserving small-scale vortical structures, shock waves, and discontinuities is critical for accuracy.

Another challenge to note is that the stochastic sampling used during generation introduces variability, which can reduce consistency in predictions. For example, two different reconstructions from the same input may differ slightly due to sampling noise. This is an issue when repeatable outputs are needed. Attempts to reduce this variation through output averaging only further blur the predicted fields. The dimensionality of the latent space also significantly affects model performance. If chosen too low, the model fails to capture complexity, and if chosen too high, training becomes unstable or overfitted.

Recent works have attempted to mitigate these issues by introducing physics-informed constraints, such as enforcing mass conservation or incorporating physical loss terms during training[25]. While these adjustments improve realism, they don't fully overcome the inherent trade-off in VAEs between capturing variability and preserving fine detail.

To summarise, VAEs offer a flexible and interpretable framework for modelling fluid flows, especially when uncertainty quantification or parametric control is required. However, their tendency to produce smooth reconstructions and their limited ability to recover fine-scale features limit their use in high-fidelity flow generation. These limitations have spurred interest in more robust generative approaches like Diffusion Models, which combine training stability with strong detail preservation and are increasingly seen as a promising solution for high-resolution flow field reconstruction tasks.

2.3. Super-Resolution as Image Translation in CFD

Super-resolution (SR) is a widely used technique in CFD that focuses on enhancing low-resolution flow fields into high-resolution counterparts. In practical terms, SR aims to reconstruct fine-scale structures, such as vortices, shock waves, and shear layers that are often lost in under-resolved simulations like coarse LES or RANS. Super-resolution models serve as surrogate upscaling tools, enabling high-fidelity insights from relatively inexpensive simulations. In doing so, they bridge the gap between computational efficiency and physical accuracy [15].

Although SR is traditionally treated as a regression problem, it can be more effectively understood as a form of image-to-image translation. The objective is to translate a low-detail or incomplete input into a refined, high-fidelity output. This is conceptually like other translation tasks, such as converting grayscale images to colour or restoring blurred images to a sharper version. In this study, the super-resolution task involves translating Schlieren images into high-resolution velocity fields, framing it as a conditional generation problem

Early work on SR in fluids leveraged CNNs trained with pixel-wise loss functions such as MSE. Models like SR-CNN [26] and Deep Convolutional SCN[27] demonstrated the ability to upscale flow fields by learning mappings between coarse and fine resolutions. However, MSE-based training tends to produce blurry reconstructions, particularly in regions with high gradients or discontinuities, as it penalises deviations uniformly across the image and favours conservative averages over sharp features [28].

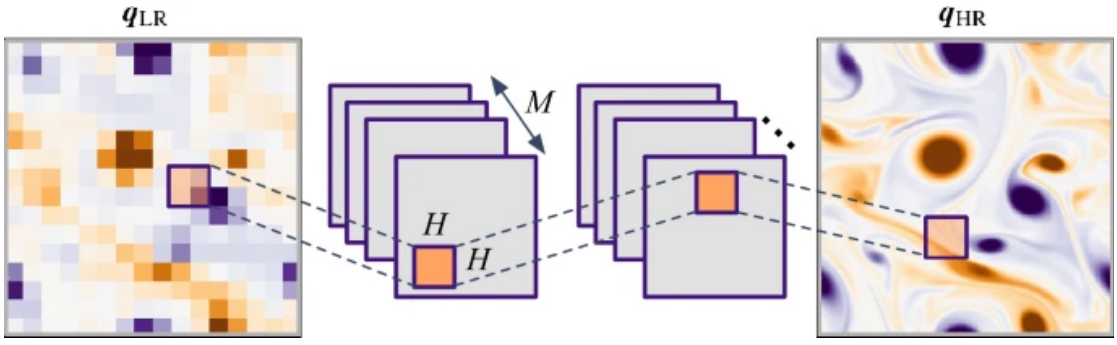


Figure 7: CNN based Super Resolution[29]

Moreover, CNN-based models are often agnostic to underlying physics, which can result in outputs that violate conservation laws. Researchers have proposed physics-informed SR models to address this, incorporating terms from the Navier–Stokes equations into the loss function to enforce mass and momentum conservation [30]. While these constraints improve realism, they are limited by their reliance on handcrafted loss formulations and can introduce optimisation difficulties.

To improve reconstruction sharpness and overcome the limitations of CNN-based models, generative models such as GANs and VAEs have been explored for super-resolution in CFD. GANs have demonstrated success in recovering small-scale features by using adversarial training to generate sharper and more realistic outputs [20]. However, as mentioned earlier, they are prone to mode collapse, training instability, and lack explicit enforcement of physical laws. VAEs, by contrast, provide a probabilistic framework that supports uncertainty quantification and latent-space control, but their reliance on pixel-wise loss leads to blurry reconstructions, especially in regions with high gradients and discontinuities. Additionally, the stochastic nature of VAE sampling introduces variability that can hinder consistent prediction quality.

Another key challenge in SR models is generalisation across different geometries, flow conditions, or Reynolds numbers. A model trained on one configuration, for instance, cylinder flow, may perform poorly on an airfoil configuration. This is due to the high nonlinearity and contextual dependence of fluid systems. Some works have proposed transfer learning and multi-fidelity frameworks to address this issue. For instance, SURFNet [31] demonstrated improved cross-geometry performance by first training on low-resolution data and then fine-tuning on high-resolution targets, but at the cost of model complexity and increased data demands.

Understanding super-resolution as an image-to-image translation task is especially relevant for the present work, where the objective is to reconstruct velocity fields conditioned on Schlieren images. This framing highlights the importance of models capable of conditional

generation, high-resolution fidelity, generalisation, and uncertainty handling, a combination of requirements that existing CNNs, GANs, and VAEs struggle to fully satisfy.

These limitations lead us to DDPMs, which combine probabilistic modelling with strong detail preservation and stable training. As discussed in the following section, DDPMs provide a promising alternative for SR tasks in CFD, particularly where preserving fine-scale physics is essential.

2.4. Diffusion Models for CFD

Diffusion models have recently emerged as a powerful class of generative models capable of producing high-fidelity, diverse, and stable outputs across various applications. These include image synthesis, inpainting, and super-resolution. In contrast to GANs and VAEs, which generate outputs in a single forward pass, diffusion models construct data samples gradually through a multi-step denoising process that begins with pure noise.

The DDPM framework consists of two primary stages. First, a forward diffusion process, where Gaussian noise is progressively added to the data over a series of time steps until the original structure is destroyed. This is followed by a reverse process, where a neural network is trained to iteratively denoise this corrupted input and recover the original sample[32]. This iterative refinement allows the model to recover complex spatial features with greater stability and precision than one-shot generation techniques. A schematic of the workflow is given in Figure 8.

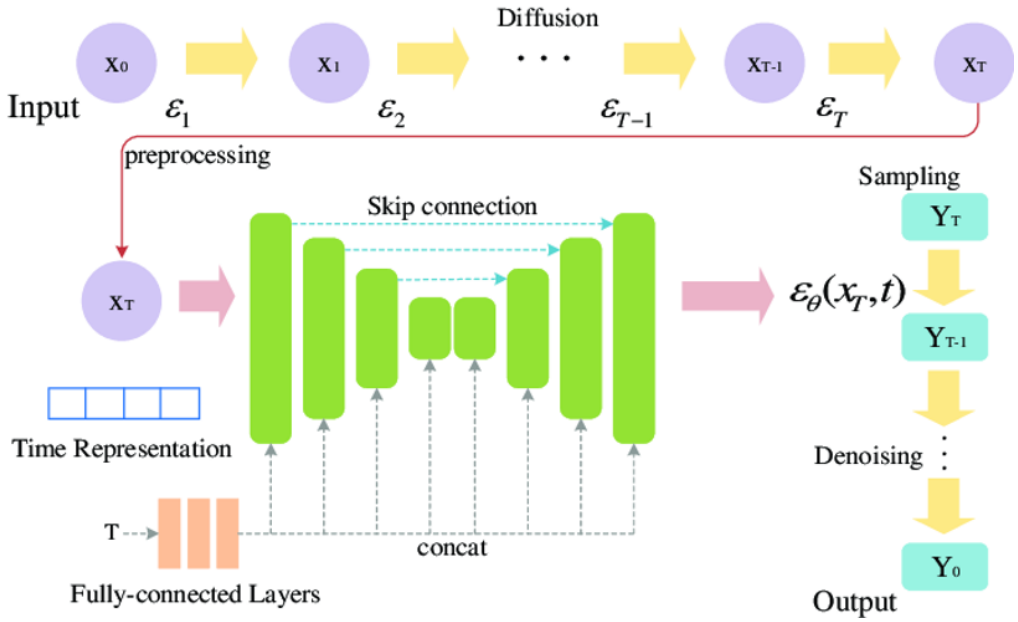


Figure 8: DDPM Workflow[33]

One of the key advantages of diffusion models is their training stability. Unlike GANs, which require a delicate adversarial setup, diffusion models follow a more straightforward training process. They use a simple regression-based loss function, making them more stable and reliable during training. This stability is particularly valuable when working with complex, high-dimensional fluid flow data, as it ensures that the model learns effectively without the risk of divergence or instability. Another key advantage is that diffusion models do not suffer from mode collapse. While GANs often suffer from mode collapse, diffusion models learn to reconstruct the entire data distribution[34].

Moreover, diffusion models inherently support probabilistic sampling, making them well-suited for modelling the uncertainty and variability inherent in turbulent flow fields. Unlike VAEs, which often smooth out critical structures, diffusion models generate samples that retain fine-scale details such as shock waves and vortices due to their iterative denoising nature. In comparative benchmarks on natural image tasks, diffusion models have even been shown to surpass GANs in perceptual quality while maintaining diversity and stability [35].

These properties make DDPMs a compelling choice for generative modelling in CFD, where training robustness, fine detail reconstruction, and physical interpretability are crucial.

2.4.1. Conditional DDPMs for Image Translation

An extension of the DDPM framework is the Conditional DDPM, which enables data generation based on additional input information. This capability is particularly relevant in image-to-image translation tasks, which aim to generate a target image. In a CFD context, this translates to reconstructing flow field variables from Schlieren images, for example, which visualise density gradients but lack direct information about other flow variables. Conditional DDPMs condition the reverse diffusion process on these indirect observations, resulting in high-quality reconstructions.

Unlike GANs, conditional DDPMs do not require a separate discriminator, and unlike VAEs, they are not constrained by a latent bottleneck. Instead, the conditioning signal is integrated directly into the reverse diffusion process often via concatenation, cross-attention, or Feature-wise Linear Modulation (FiLM). This enables context-aware, spatially detailed generation. The iterative denoising nature of DDPMs also helps preserve fine-scale structure and makes them inherently stable during training.

Recent applications demonstrate the strong potential of conditional diffusion models across scientific domains. In fluid mechanics, a study introduced a physics-informed conditional DDPM for reconstructing high-fidelity CFD fields from low-fidelity inputs[9]. Their model achieved accurate and physically consistent reconstructions even on out-of-distribution inputs without retraining by incorporating PDE residuals as conditioning signals. This demonstrated the model's robustness and generalizability in challenging CFD scenarios.

A similar strategy has been successfully applied in medical imaging, where conditional DDPMs are used to enhance the quality of diagnostic scans[36]. For instance, they have been employed to translate CBCT scans, which are a type of 3-D X-ray imaging commonly

used in dental procedures and radiation therapy, into synthetic CT images that offer higher resolution and improved anatomical accuracy. These models are trained to generate clean, detailed images from artefact-prone CBCT inputs, which are often limited in diagnostic quality.

These studies demonstrate the unique strengths of conditional DDPMs for tasks involving indirect supervision, uncertainty, and multi-scale spatial structure.

2.5. Research Gap and Motivation

Machine learning has advanced surrogate modelling in CFD, with CNNs, VAEs, and GANs being widely applied to tasks like super-resolution and flow reconstruction. However, these models face persistent challenges. CNNs produce overly smooth results, GANs are prone to instability and mode collapse, and VAEs tend to blur fine-scale structures. Additionally, most existing models struggle to generalise beyond specific flow configurations.

Diffusion models offer a promising alternative. They provide stable training, probabilistic sampling, and detailed reconstructions. While they have recently been explored in fluid applications, their use in supersonic flows, particularly in atomisation nozzles, remains largely unexplored. These regimes involve complex phenomena such as shocks and compressibility effects, which demand models capable of capturing fine structures while maintaining physical consistency.

This thesis addresses this gap by applying a Conditional DDPM to reconstruct high-resolution velocity fields from Schlieren images in supersonic nozzle flows. The goal is to evaluate whether conditional diffusion models can recover physically realistic flow fields from indirect visual input, offering a stable, generalisable alternative to traditional ML-based reconstruction methods in compressible flow regimes.

3. Methodology

This thesis follows an exploratory, experiment-driven approach to create conditional diffusion models to reconstruct high-resolution velocity fields from Schlieren images in supersonic flow regimes. Instead of starting with a fixed architecture, the methodology focuses on iteratively enhancing a baseline DDPM and systematically refining its components to improve performance.

The overall pipeline consists of three core stages:

1. Data Preprocessing: High-fidelity CFD simulations are cleaned, normalised, and scaled to match the input requirements of the diffusion model. Special care is taken to preserve physical gradients and edge features, which are critical in supersonic flow prediction.
2. Model Development: The baseline conditional DDPM is constructed using a U-Net backbone for noise prediction, with Schlieren images and physical conditioning information integrated through various mechanisms. The diffusion process is experimented upon through different noise schedules, timestep sampling strategies, and reverse sampling variants. Multiple U-Net architectural configurations are explored, including multi-resolution conditioning and attention-based enhancements.
3. Evaluation and Comparison: Model outputs are evaluated quantitatively via MSE and SSIM, and qualitatively (visual inspection of flow features). The diffusion model is benchmarked against a direct Schlieren-to-velocity supervised U-Net. Different loss functions are investigated, including pure MSE, a hybrid KL-divergence-based formulation aligned with learned variance, and a weighted MSE and SSIM loss.

The methodology is structured to enable iterative experimentation across components. The sections that follow describe each stage of the process in detail.

3.1. Dataset Description

The dataset used in this study is derived from high-fidelity CFD simulations of a gas atomisation plant, where a supersonic gas jet is used to atomise molten metal into fine droplets. The physical process was modelled within ANSYS Fluent. The simulations solve the RANS equations with the Shear Stress Transport (SST) k−ω turbulence model. The domain represents the nozzle region of an atomisation chamber, where high-speed gas enters through an annular slit and interacts with a central molten jet. A schematic of the experimental setup is shown in Figure 9.

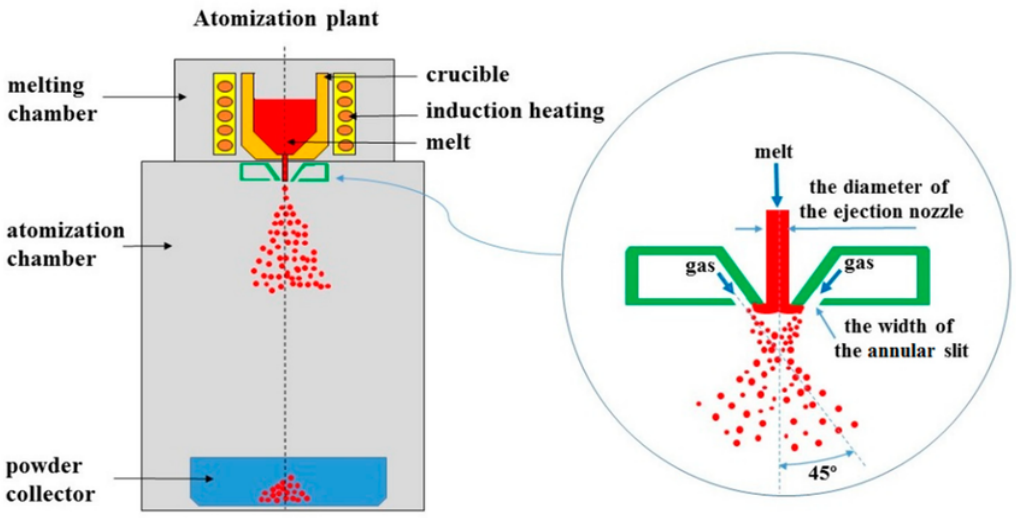


Figure 9: Schematic of the gas atomisation plant[37]

To reduce computational complexity, axisymmetry is assumed about the vertical centerline, allowing for a 2D flow representation. This simplification retains key features such as shock waves, expansion fans, and shear layers, which are essential in characterising compressible supersonic flows.

The dataset includes only the near-field region located immediately downstream of the nozzle (Figure 10). This area contains the most intense interactions between the supersonic gas and molten jet and features complex phenomena such as shock diamonds, compressibility effects, and primary breakup.

A total of 5,043 simulation samples were generated by systematically varying the inlet pressure**,** temperature, and working fluid. The temperature values were uniformly sampled between 274 K and 673 K, while pressures ranged from 5 bar to 75 bar, ensuring a diverse range of operating conditions. The working fluid was set to either argon or nitrogen, encoded as a categorical input. This variation enables the dataset to capture a wide spectrum of supersonic flow behaviours and material responses. This setup enables broad physical diversity, covering a wide range of supersonic flow conditions and material responses. [38]

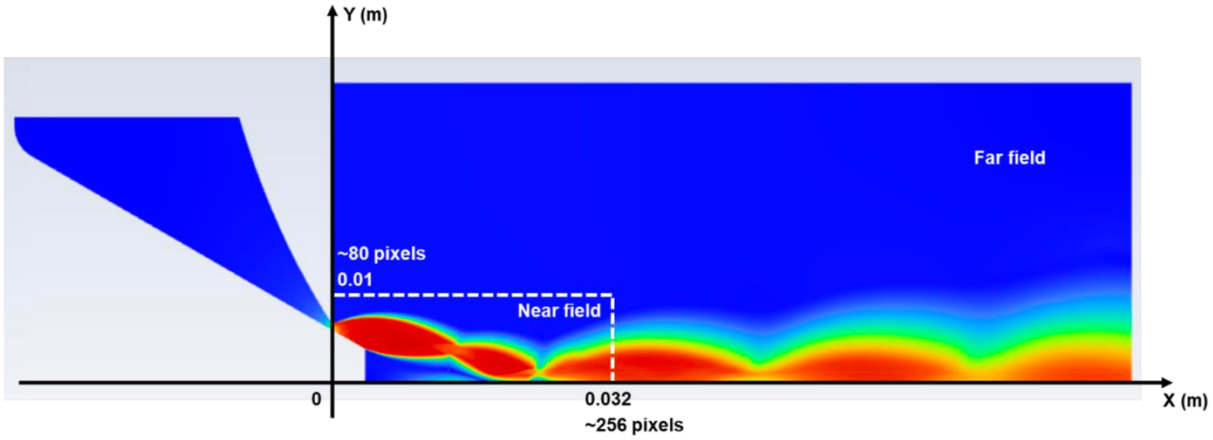


Figure 10: Near-field focus region of the simulation domain*[38]*

Each simulation output is stored in NPZ format, containing eight flow field variables. The spatial resolution for each variable is 80 × 256 pixels, corresponding to the near-field simulation window. The complete dataset occupies approximately 6.19 GB.

3.2. Exploratory Data Analysis

The dataset exhibits significant variability due to the wide range of simulation conditions. Each NPZ file contains the following physical variables:

1. Density
2. Velocity Components (u, v)**:** Velocity components in the x and y directions, respectively.
3. Velocity Magnitude: The scalar magnitude of the fluid velocity.
4. Velocity Angle: The angle of the velocity vector relative to the x-axis.
5. Mach Number: Ratio of the local fluid velocity to the local speed of sound
6. Schlieren Intensity: Visual representation of density gradients in the fluid flow, post-processed using an Abel transformation.
7. Density Gradient Field: Measures spatial changes in fluid density.

Due to the diverse simulation parameters, the dataset covers a broad range of flow behaviours. To gain a comprehensive understanding, we perform an initial EDA, focusing on the statistical properties of the variables. To quantify the variability within the dataset, we first compute the fundamental statistical properties of each variable, including the mean, median, standard deviation, minimum, and maximum values. The summary of these statistics is presented in Table 1.

Table 1: Data Analysis of Flow Field Variables

Variable	Mean	Median	Std	Min	Max	Skewness
Velocity(u)	231.69	205.08	176.60	0.0	544.00	0.12
Velocity(v)	-0.30	-0.10	1.82	-11.90	12.76	1.41
Density	1.43	1.34	0.52	0.0	3.39	-0.28
Velocity(mag)	231.70	205.14	176.60	0.0	544.02	0.12
Mach	0.78	0.62	0.64	0.0	2.24	0.32
Velocity(angle)	-0.22	-0.03	0.83	-3.14	3.14	0.18
Schlieren	-63.81	-278.98	14849.04	-71937.87	83241.65	0.59
Density(gradient)	-0.78	0.09	224.34	-4535.17	2422.63	-0.98

The statistics reveal that the distributions of the variables in the dataset are significantly different from one another. Both velocity components and magnitude show low skewness, indicating balanced data around their respective mean. On the contrary, the schlieren and density gradient variables exhibit high skewness, suggesting they contain more extreme values. Shock structures and corresponding rapid density changes in the flow field drive these. The Mach number also exhibits a slight positive skew.

These differences underscore the variability within the dataset, making it important to carefully consider the characteristics of each variable before proceeding with further analysis and preprocessing. The next section outlines the preprocessing steps applied to normalise the dataset, address outliers, and prepare the variables for use in the diffusion pipeline.

3.3. Data Preprocessing

3.3.1. Outlier Handling

To ensure numerical stability and prevent anomalous spikes from degrading model performance, outlier detection was applied on a per-image basis. A Z-score thresholding approach was used, where a pixel (value X) was flagged as a potential outlier if:

$$Z = \frac{X - \mu_{image}}{\sigma_{image}} > \tau \tag{1}$$

Here, μ_{image} and σ_{image} represent the mean and standard deviation computed across all pixel values within a single simulation image, and τ is a user-defined threshold. However, since large gradients are physically expected near shocks and shear layers in supersonic flows, a high Z-score alone was not sufficient to mark a pixel as an outlier.

Therefore, the preprocessing pipeline considers the gradient around each pixel of these potential outliers. If the gradient difference between a potential outlier and its neighbouring pixels is small, the pixel is deemed part of the flow field rather than an anomaly. This 2-step

process ensures that sharp flow features seen in supersonic flows are not inadvertently treated as outliers.

Since our dataset is relatively small by conventional deep learning standards, we deliberately chose not to remove outliers entirely. Instead, they are handled by replacing them with the local mean of their valid neighbour pixels. Furthermore, before this step, the dataset was already cleaned of any numerically inconsistent values.

Outlier handling was selectively applied to all variables, excluding Schlieren data. Schlieren images primarily capture sharp discontinuities such as shock waves, and even minor adjustments could blur these critical features. Therefore, Schlieren data was deliberately left untouched to preserve its physical accuracy and sharpness

3.3.2. Data Normalisation and Scaling

Once outliers were addressed, all flow variables were processed using a two-step transformation:

3.3.2.1. Step 1: Normalisation

Each image was normalised using the mean and standard deviation across the dataset to account for variability across samples according to:

$$X_{norm} = \frac{X - \mu_{dataset}}{\sigma_{dataset}} \tag{2}$$

Where:

- X = pixel values
- $\mu_{dataset}$ = mean of the variable across the entire dataset
- $\sigma_{dataset}$ = standard deviation of the variable across the entire dataset

Normalisation ensures that the variables are centred around zero and are comparable in scale. Centring the data helps improve training stability.

3.3.2.2. Step 2: Rescaling to [-1,1]

These normalised values were subsequently scaled to the range [-1, 1].

$$X_{scaled} = -1 + 2.\frac{X_{norm} - min(X_{norm})}{max(X_{norm}) - min(X_{norm})} \quad (3)$$

This scaling is essential because DDPMs perform best when the data lies within this range, as it aligns with the noise distribution added during the diffusion process. Aligning the data range with the noise helps the model learn the denoising process more effectively.

Special Cases: Pressure, Temperature, and Material Encoding:

Three auxiliary inputs are included with every sample to describe the experimental setup: temperature**,** pressure, and working fluid type. These values serve as conditioning signals that help the model account for the underlying flow conditions. The working fluid: Argon or Nitrogen, was treated as a categorical variable and numerically encoded as 0 and 1, respectively. The temperature and pressure inputs, treated as continuous variables, were scaled to the range [-1,1] to maintain consistency with the rest of the normalised dataset. These three values were combined into a single-label tensor and passed to the model as external conditioning inputs.

3.3.3. Data Resampling

To enhance spatial continuity, bicubic resampling is applied to the cleaned dataset. Bicubic interpolation smoothens transitions between pixel values and helps reduce minor irregularities without significantly altering essential flow structures.

3.3.4. Data Loading

The final preprocessed tensors were loaded into PyTorch Dataloaders for batched training and evaluation. A few physically meaningful data augmentation strategies were briefly tested to explore the effect of dataset diversity on training. These included horizontal flipping and 180-degree rotation of flow field images, both preserving the underlying physics in a symmetric nozzle geometry. Additionally, a lower special resolution variant of the dataset

was created by cropping images to a size of 80×128 instead of the full 80×256 to assess the performance trade-offs under limited spatial information. These variations were used selectively in certain experimental runs and were not part of the default training configuration unless otherwise stated. Finally, setting the random seed ensures that reproducible test sets are formed for fair comparison between different models. Representative samples from the training dataset are shown in Figure 11 and Figure 12 to illustrate the preprocessed velocity magnitude and Schlieren pairs.

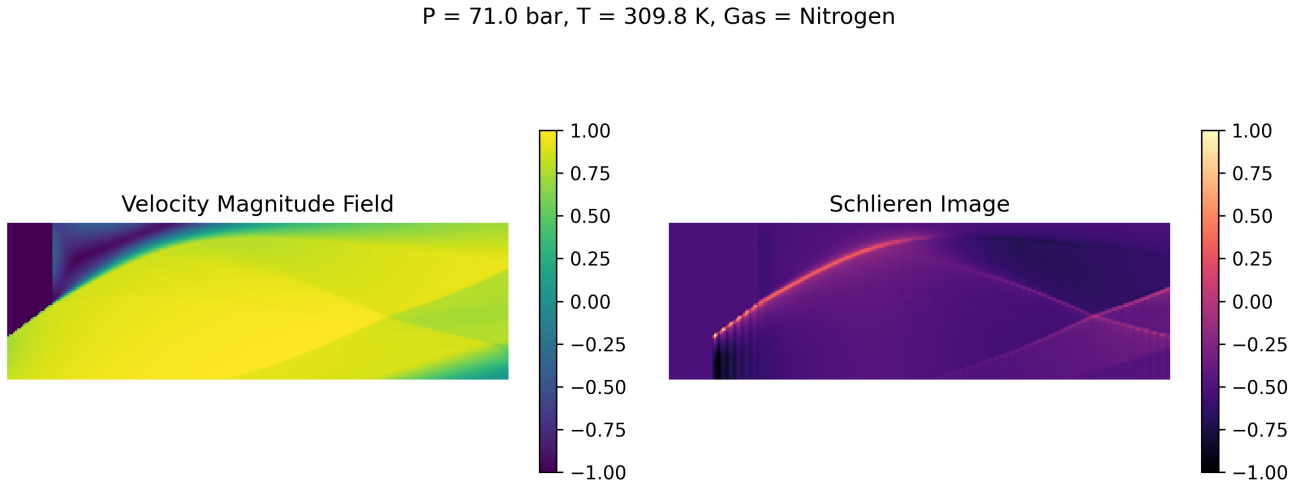


Figure 11: Preprocessed training sample (Nitrogen). Left: Velocity magnitude field. Right: Corresponding Schlieren image. Conditions: P = 71 bar, T = 309.8 K

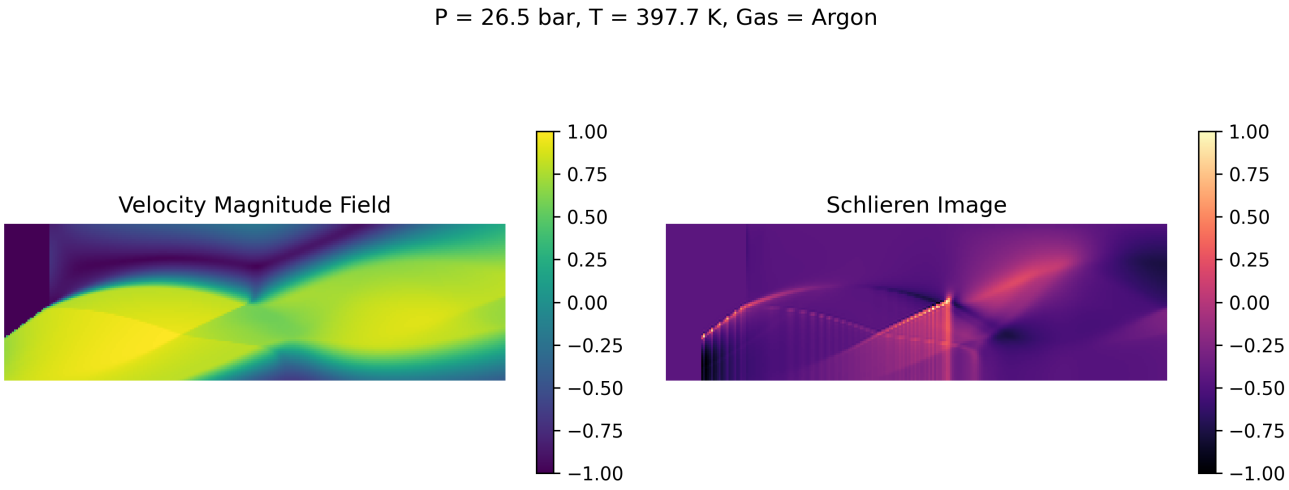


Figure 12: Preprocessed training sample (Argon). Left: Velocity magnitude field. Right: Corresponding Schlieren image. Conditions: P = 26.5.2 bar, T = 397.7 K

3.4. Denoising Diffusion Probabilistic Model

This section outlines the specific implementation of DDPMs used in this work. Data is progressively noised via a forward diffusion process and then reconstructed using a learned denoising network in the reverse direction. While the theoretical foundations of DDPMs were discussed in Section 2, we focus on the mathematical formulation and scheduling choices here.

3.4.1. Forward Diffusion Process

The forward diffusion process defines how noise is gradually added to a clean data sample over a sequence of timesteps. In this work, each ground truth sample x_0 is noised over T steps by adding Gaussian noise in a controlled manner. At each timestep t, a small amount of noise is added, producing a noisy sample x_t. This is done recursively via:

$$q(x_t|x_{t-1}) = N\left(x_t; \sqrt{1-\beta_t}.x_{t-1}, \beta_t I\right) \tag{4}$$

Here, β_t is defined by a noise schedule. This parameter controls how much of the previous signal is preserved. A smaller β_t value retains more structure from x_{t-1} while a larger value injects more Gaussian noise. Over many steps, this process gradually destroys the original data. In the limit where t approaches very high values, the sample x_t becomes pure Gaussian noise, effectively removing all traces of the original sample.

To simplify training, this process is reparametrized to express the noised image x_t directly in terms of the original image x_0 and the noise term ε.

$$x_t = \sqrt{\overline{\alpha_t}}.x_0 + \sqrt{1-\overline{\alpha_t}}.\varepsilon \tag{5}$$

Where α_t = 1 - β_t, and $\overline{\alpha_t}$ = $\prod_{s=1}^{t} \alpha_s$ is the cumulative product. Therefore, using this reparameterisation, we can efficiently sample noisy versions of clean data at arbitrary timesteps during training without repeatedly going through the transition function.

Figure 13 shows the forward diffusion process of a velocity field sample from the dataset. The figure demonstrates the gradual transition from the original flow field to fully noisy data, highlighting the intermediate stages at selected time steps. This captures the forward diffusion process, illustrating how noise is incrementally introduced during the forward process.

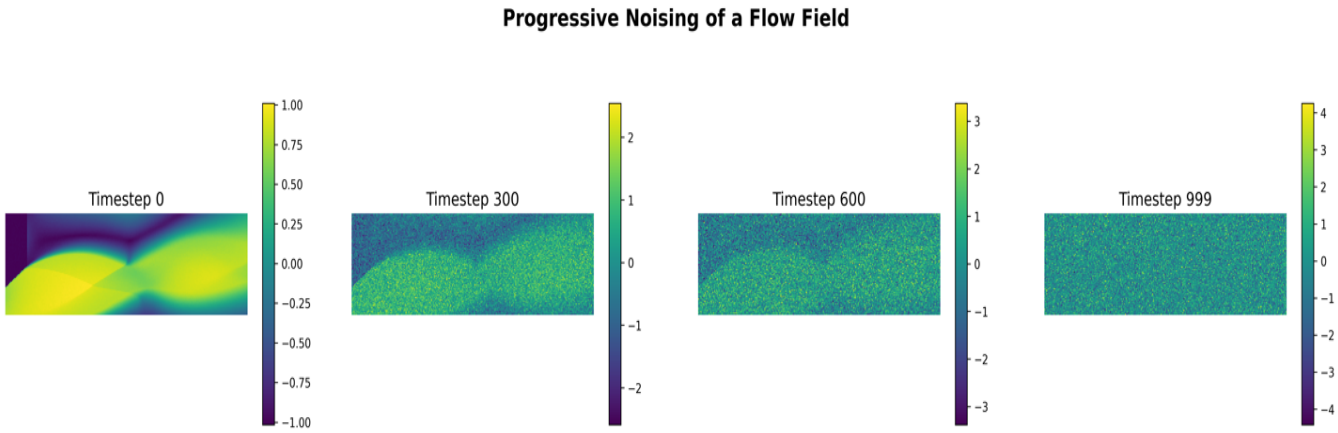


Figure 13: Visualising Progressive Noising of the Velocity Magnitude field

3.4.1.1. Noise Scheduling

The choice of noise scheduler $\{\beta_t\}_{t=1}^{T}$ plays a crucial role in the performance of the diffusion model. This determines the rate at which noise is added to the input data over time and, thus, how much of the original sample is destroyed at each step. If the schedule is too gentle, the model may not fully corrupt the data. This will cause poor performance during sampling since the model is never asked to denoise truly noisy inputs during training. On the other hand, an overly aggressive noise schedule may destroy the image too quickly, making it difficult for the model to learn meaningful denoising steps.

Ideally, the scheduler should gradually reduce the signal-to-noise ratio over time, so the model sees a smooth transition from structured inputs to pure Gaussian noise. Importantly, the final timestep should correspond to a sample drawn from an isotropic Gaussian distribution. Listed below are some widely used noise schedulers used in DDPMs [39,40]:

- Linear Schedule: This is the simplest formulation, where noise linearly increases over time. The variance parameter is interpolated between two values.

$$\beta_t = \beta_{start} + (\beta_{end} - \beta_{start}).\left(\frac{t - 1}{T - 1}\right) \tag{6}$$

  Typical values: β_{start} = 10^{-4}, β_{end} = 0.02

- Sigmoid Schedule: The sigmoid schedule introduces a gradual change in noise, starting and ending slowly with a sharper increase in the middle timesteps. It can be expressed as:

$$\beta_t = \sigma\left(\frac{t - \frac{T}{2}}{k}\right).(\beta_{end} - \beta_{start}) + \beta_{start} \tag{7}$$

  where the parameter k controls the steepness, and σ represents the sigmoid function.

- Cosine Schedule: This schedule defines a smooth decay in the signal-to-noise ratio using a cosine function. The formulation is as follows:

$$\overline{\alpha_t} = cos^2\left(\frac{\frac{t}{T} + s}{1+s}\frac{\pi}{2}\right) \tag{8}$$

S is set as 0.008 to prevent vanishing noise at initial timesteps.

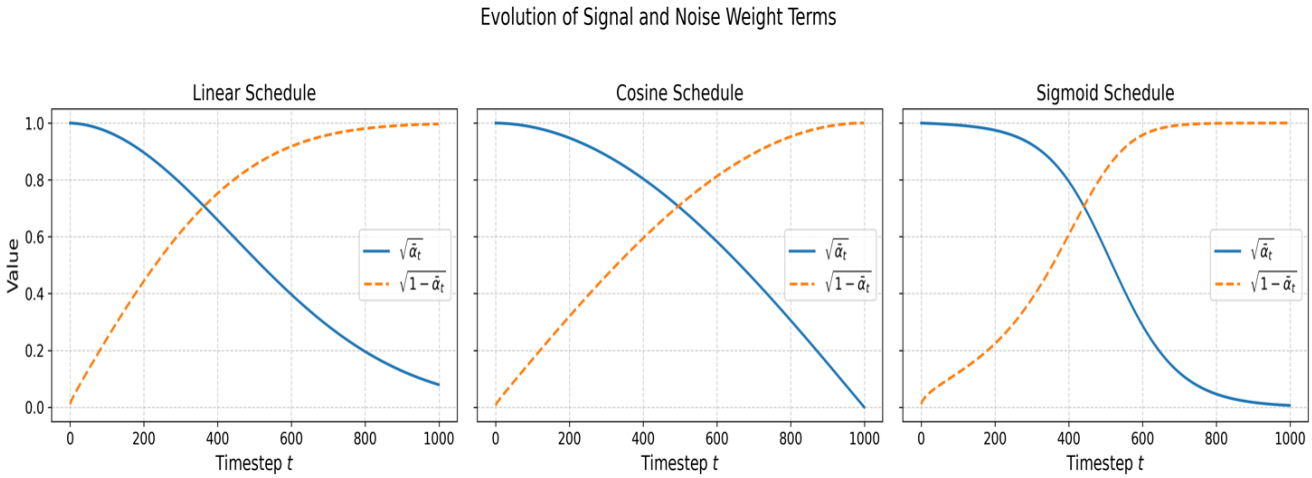


Figure 14: Evolution of signal and noise weighting terms across 1000 timesteps for Linear, Cosine and Sigmoid Schedules

We visualise how the signal and noise weight terms evolve over time to further illustrate the effect of different noise schedules. Figure 14 shows the behavior of $\sqrt{\overline{\alpha_t}}$ and $\sqrt{1 - \overline{\alpha_t}}$, which controls how much of the original signal and Gaussian noise, respectively, are present at each timestep. In all cases, the signal term decreases, and the noise term increases as the diffusion progresses. However, the shape of this transition varies significantly across schedules. The cosine schedule decays the signal more gradually, especially in the early stages, which allows the model to preserve structure for longer. The sigmoid schedule introduces a sharper transition in the middle of the process, while the linear schedule decays the signal at a constant rate.

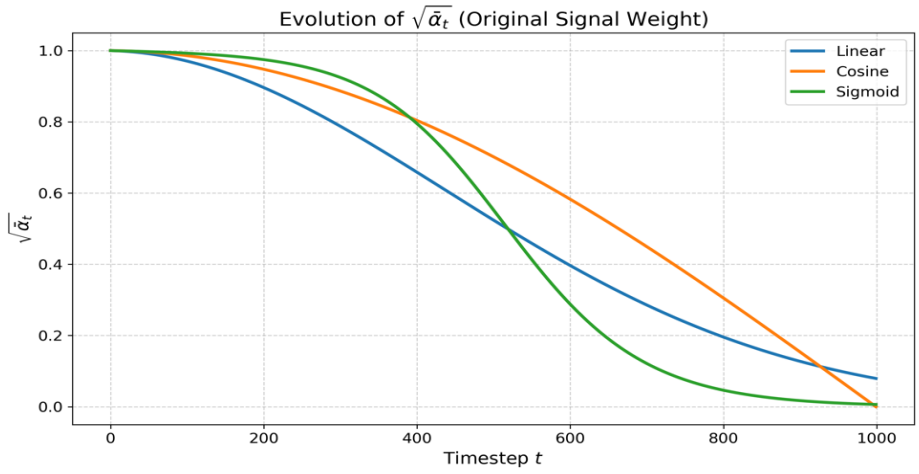


Figure 15: Comparison of Signal retention term across different schedulers

Figure 15 isolates the signal weight term across all three schedules, making their differences more apparent. The cosine schedule maintains higher signal strength for a larger portion of the diffusion trajectory than the linear and sigmoid schedules.

While cosine scheduling has demonstrated strong empirical performance in image synthesis tasks, its effectiveness in domain-specific applications such as fluid flow reconstruction remains relatively underexplored. To address this, we systematically evaluate multiple noise schedules in the context of our conditional DDPM framework to determine the most suitable configuration for preserving physically meaningful features.

3.4.1.2. Advanced Noising Options: Non-Uniform Sampling and Edge-Aware Noise

In addition to experimenting with different noise schedules, a few newer techniques proposed in the diffusion model community are also explored. These methods aim to improve the generative process, but they have been challenging to implement effectively, as noted by numerous researchers. The objective is to examine whether these techniques might benefit the model in this specific task.

One such method is non-uniform timestep sampling [41]. In standard DDPMs, noise is added uniformly across timesteps. However, not all timesteps contribute equally to learning. The core idea behind this method is to prioritise the more informative timesteps,

where the model struggles most with denoising. This is achieved by assigning a sampling probability to each timestep based on its loss gradient. This allows the model to focus on the regions of the diffusion process where learning is most effective. Typically, the model is made to train on a few warm-up epochs with uniform sampling to accumulate sufficient statistics. While this approach has shown promise in natural image domains, its utility in physics-based problems like fluid flow reconstruction remains underexplored. This work evaluates its performance in conditional DDPMs for supersonic nozzle flow.

Another strategy explored in this work is edge-aware noise scheduling, based on a recent study[42]. Unlike standard isotropic noising, this method introduces spatial variation in noise depending on the local image structure. Gradient magnitudes are computed for each image using the Sobel operator, and pixels with strong gradients (edges) receive less noise during early diffusion steps. This method has shown promising results in image generation tasks, demonstrating improved FID scores and better structure preservation. As diffusion progresses, a time-dependent transition function gradually shifts the noising process from anisotropic to standard isotropic noising.

This two-stage design aims to preserve sharp flow structures such as shocks and discontinuities early in the forward process while still allowing complete corruption by the final timestep. The method is implemented into our conditional DDPM framework and evaluated alongside other noise schedules described earlier.

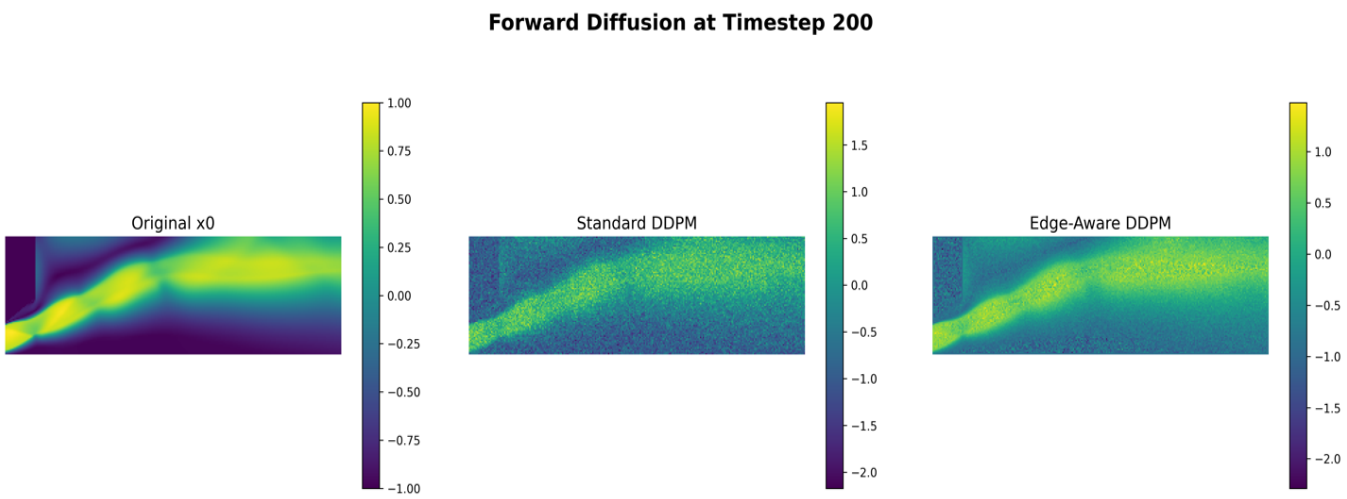


Figure 16: Comparison of noised velocity fields at timestep 300 from Standard DDPM and Edge-Aware DDPM

Figure 16 illustrates the noised velocity field generated by the Standard DDPM and the Edge-Aware DDPM at an intermediate timestep (t=200). The Edge-Aware DDPM preserves sharper gradients and finer flow structures compared to the Standard DDPM, demonstrating its effectiveness in maintaining edge information during the forward diffusion process

3.4.2. Reverse Diffusion Process

The reverse process in a DDPM aims to recover the original sample x_0 starting from pure noise x_t. This is achieved by iteratively denoising the sample over a series of timesteps t=T to t=0 using a learned Markov process. At each step, the model estimates a Gaussian distribution that defines the transition to the previous state x_{t-1}:

$$p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2 . I) \quad (10)$$

To perform this transition, we need:

- The mean $\mu_\theta(x_t, t)$ predicted by the model
- The variance σ_t^2 which is derived from the noise schedule.

The reverse sampling algorithm is briefly described as follows:

Step 1: Predicting Noise $\epsilon_\theta(x_t, t)$ and Clean Image: $x_{0,pred}$

The model first estimates the noise $\epsilon_\theta(x_t, t)$ using the neural network and subsequently predicts the clean image as:

$$x_{0,pred} = \frac{1}{\sqrt{\overline{\alpha_t}}}\left[\ x_t - \sqrt{1-\overline{\alpha_t}}\,.\,\epsilon_\theta(x_t, t)\ \right] \tag{11}$$

In practice, the predicted $x_{0,pred}$ is clipped to the range [−1,1] as the input data has been preprocessed to lie within this range. Clipping ensures that the denoised image remains within valid bounds to maintain the fidelity of the generated samples.

Step 2: Computing the Mean: $\mu_\theta(x_t, t)$

Once the predicted clean image $x_{0,pred}$ is computed, the estimated mean for the reverse Gaussian can be calculated as:

$$\mu_\theta(x_t, t) = \frac{\sqrt{\overline{\alpha_{t-1}}}\,\beta_t}{1-\overline{\alpha_t}} x_{0,pred} + \frac{\sqrt{\overline{\alpha_t}}(1-\overline{\alpha_{t-1}})}{1-\overline{\alpha_t}} x_t \tag{12}$$

Equivalently, after substituting the definition of $x_{0,pred}$, the mean can also be expressed directly in terms of the predicted noise as:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left[x_t - \frac{\beta_t}{\sqrt{1-\overline{\alpha_t}}}\,\epsilon_\theta(x_t, t)\right] \tag{13}$$

Step 3: Sampling from the Reverse Distribution

With the known mean and variance of the reverse distribution, the next image in the reverse process is sampled according to the following reparameterisation:

$$x_{t-1} = \mu_\theta(x_t, t) + \sigma_t.\,z.\,; z \sim N(0, I) \tag{14}$$

The noise term z introduces stochasticity into the generation process, allowing the model to sample from a learned distribution rather than outputting averaged predictions. At the final timestep this noise term is omitted, and the mean prediction is returned as the final output.

3.4.2.1. Fixed Variance

In the original DDPM formulation[32], the variance used in the reverse process is derived as:

$$\sigma_t^2 = \frac{\beta_t(1 - \overline{\alpha_{t-1}})}{1 - \overline{\alpha_t}} \tag{15}$$

However, it has been shown that using the simplified form $\sigma_t^2 = \beta_t$ yields comparable sample quality in practice[41]. This thesis adopts the same approach and uses the fixed variance β_t throughout the reverse sampling process.

3.4.2.2. Learned Variance

In addition to the fixed variance approach, the reverse process can be extended to use learned variances [41]. In this formulation, the model is trained to predict a scalar interpolation factor v, which is used to compute a log variance for each timestep via interpolation between two known bounds:

- The lower bound: log $\widetilde{\beta_t}$ derived from the true posterior variance
- The upper bound: log β_t, log of the forward schedule

The resulting expression for the learned log-variance is:

$$log\sigma_t^2 = v.\log \beta_t + (1 - v).\log \widetilde{\beta_t} \tag{16}$$

This approach allows the model to adaptively choose how much uncertainty to assign at each timestep by blending between these two bounds. The learned variance is then used in the reverse sampling process. Implementing this requires modifying the U-Net to output the noise as well as the interpolation weight. The change also necessitates a hybrid loss, described in Section 3.6.2, that balances reconstruction accuracy with probabilistic modelling.

3.5. U-Net Architecture Overview

U-Net is a widely used encoder-decoder architecture initially developed for biomedical image segmentation[43], and it has since gained popularity in generative models due to its ability to retain spatial information. The U-Net used in DDPMs serves as the core denoising network within the reverse diffusion process. At each diffusion timestep, it receives a noisy velocity field (along with the timestep information) as input and predicts the added noise.
The architecture is broadly based on the U-Net design presented in the original DDPM paper[32], which has a symmetric encoder-decoder structure with skip connections between corresponding resolution levels. These skip connections help preserve spatial features that are often lost during downsampling.

The input to the U-Net consists of a single-channel, noised velocity magnitude image. The Schlieren image is processed and concatenated with the input tensor. This combined input is passed through an initial convolutional layer that projects it into a 128-dimensional feature space. From there, the encoder processes the data through a series of four downsampling stages, reducing the spatial resolution while increasing the feature dimensionality. A bottleneck layer sits at the deepest point of the architecture, followed by a symmetric decoder path that upsamples the features back to the original resolution. Each decoder stage also receives skip connections from its corresponding encoder layer, allowing the model to recover spatial details lost during downsampling.

The output of the final decoder layer is passed through a convolutional head to produce the predicted noise tensor, which matches the shape of the original input. The diffusion model then uses this prediction during reverse sampling.

3.5.1. Enhancements and Shared Architectural Choices

While the overall structure remains consistent with the baseline U-Net [32], several enhancements were introduced to improve training stability and flow structure reconstruction:

- Residual Blocks: All downsampling and upsampling blocks are built using residual ResNet-style modules, which improve gradient flow and feature reuse.
- Self-Attention Layers: Multi-head attention blocks are included at the mid-resolution level (16×64) in both the encoder and decoder, as well as in the bottleneck layer. These help the model capture long-range dependencies, which are critical for reconstructing global structures like shock reflections and expansion fans.
- Group Normalisation: All convolutional layers use GroupNorm (32 groups), which performs reliably across different batch sizes and improves training stability.
- SiLU Activations: The model uses SiLU activation functions, which provide smoother gradients and improve convergence compared to ReLU.
- Xavier Initialisation: All convolution and linear layers are initialised using Xavier uniform initialisation to maintain balanced signal propagation at the start of training.
- Circular Padding: All convolutional operations use circular padding, which preserves periodicity and avoids edge artefacts in the horizontal direction, a common property in CFD simulation domains.

These design choices are shared across all U-Net variants explored in this work. Architectural modifications specific to certain experiments, such as learned variance prediction or multi-resolution conditioning, are described separately in the following sections.

3.5.2. Conditioning Mechanisms

3.5.2.1. Timestep and Label Conditioning

The diffusion timestep is embedded using a sinusoidal positional encoding, similar to those used in transformer architectures[44]. This results in a fixed-length vector that captures the temporal stage of the denoising process.

The simulation-specific label tensor is passed through a small multi-layer perceptron (MLP) to project it into the same embedding space as the timestep vector.

The two embeddings are then concatenated and passed through another MLP to produce a modulation vector. This vector is used to generate a pair of learnable coefficients: a scale factor γ and a shift factor β. These values are applied to the intermediate feature maps in each residual block to adaptively modify how the model processes information at that layer. This operation is known as Feature-wise Linear Modulation (FiLM)[45] and takes the form:

$$h` = \gamma . \mathrm{h} + \beta \tag{17}$$

Here, h is the intermediate feature map, and h` is the modulated output. Intuitively, this allows the model to selectively amplify or suppress certain features depending on the current timestep and physical simulation parameters. FiLM has been shown to be a lightweight yet effective way to inject conditioning into deep networks

3.5.2.2. Spatial Conditioning

The U-Net must incorporate spatial conditioning through Schlieren images. Two architectural strategies were explored for injecting Schlieren-based spatial conditioning:

1. Single Injection Variant: In the simpler version, the Schlieren map is passed through a small convolutional encoder consisting of a 1 x 1 convolution, a GELU activation, and a 3 x 3 convolution with circular padding. This produces a feature map of the same spatial resolution and dimensionality as the input. The resulting tensor is concatenated with the noisy velocity field along the channel dimension before entering the network's initial convolution. This form of early fusion introduces spatial context into the model without modifying its deeper structure.

2. Multi Injection Variant: Instead of using the Schlieren conditioning map just once at the input, it is injected at multiple points throughout the encoder. After the initial embedding step, the Schlieren feature map is progressively downsampled using average pooling to match the resolution of each downsampling stage. At each level, a small 1×1 convolution adjusts the number of channels to match the main feature map, and the two are then added together. By reinforcing the conditioning signal

throughout the encoder, the architecture is designed to support more effective integration of spatial context, which may aid in preserving structural features across scales

3.6. Loss Functions

The training objective of the diffusion model is to learn a mapping from a noised image to the original noise that was added. Different loss functions were explored to improve reconstruction accuracy and perceptual quality, especially in high-gradient flow regions.

3.6.1. Mean Squared Error

The simplest training objective is based on the original DDPM formulation, where the model is trained to predict the added Gaussian noise at each timestep. This results in a standard mean-squared error loss:

$$\mathcal{L}_{MSE} = \mathrm{E}_t[\ \|\varepsilon - \varepsilon_\theta(x_t, t)\|^2] \tag{18}$$

This loss can be derived from the variational lower bound on the data likelihood, where minimising the KL divergence between the true and predicted reverse distributions is shown to reduce to an MSE loss under the assumption of fixed variance. This loss provides a stable and interpretable objective that directly optimises for denoising accuracy, but it does not explicitly account for perceptual similarity or structural alignment.

3.6.2. Hybrid Loss: MSE + KL Divergence for Learned Variance

As described in Section 3.4.2.2, the learned variance formulation extends the model to predict an interpolation factor v ∈ [0,1] which is used to compute the log-variance during reverse sampling. To support this, the U-Net is modified to output two channels: one for the noise prediction and one for v, which is passed through a sigmoid activation and spatially averaged to produce a scalar per sample.

Training this model requires a hybrid loss function that jointly optimises both components of the reverse distribution. The objective consists of:

- An MSE term for noise prediction
- A KL divergence term for the variance prediction

The resulting hybrid loss is:

$$\mathcal{L}_{hybrid} = |\epsilon - \varepsilon_\theta(x_t, t)|^2 + \lambda \cdot D_{\mathrm{KL}}\big(q(\, x_{t-1} \mid x_t, x_0\,) \,|\, p_\theta(\, x_{t-1} \mid x_t\,)\big) \tag{19}$$

Here, λ is a weighting factor to control the influence of the KL term. This formulation allows the model to learn both the denoising trajectory and the associated uncertainty at each timestep.

3.6.3. Hybrid Loss: MSE + SSIM Loss

MSE treats all deviations uniformly and may not prioritise preserving high-frequency or structural details. A combined loss function incorporating the Structural Similarity Index Measure (SSIM) was explored to better align the training objective with perceptual and structural fidelity.

The SSIM metric evaluates the similarity between two images based on luminance, contrast, and structural information. Unlike MSE, which penalises individual pixel errors, SSIM captures the overall spatial arrangement of patterns, making it more sensitive to the types of features present in fluid flow fields. The combined objective includes two components:

- MSE between the predicted noise and the ground truth noise
- SSIM between the ground truth velocity image and the predicted clean image based on the noise prediction of the U-Net (Equation 11)

The combined loss function is then defined as:

$$\mathcal{L}_{combined} = \alpha . L_{MSE} + (1 - \alpha)(1 - SSIM) \tag{20}$$

where $\alpha \in [0,1]$ is a tunable weighting parameter. Experimental results for different values of α are presented in the Results section.

With the training objectives defined, we now describe the setup used to evaluate each model and compare its outputs across different configurations

3.7. Evaluation Setup

Model performance was evaluated using a combination of quantitative metrics and qualitative visual inspection. The primary quantitative metrics used were the MSE and the SSIM, both of which are standard in image reconstruction and generative modelling tasks. In addition to these metrics, visual inspection of flow fields was performed to assess qualitative fidelity.

All models were evaluated on a fixed test set comprising 10% of the dataset, with the remaining 90% used for training. Having consistent test sets was ensured for accurate comparison of various models

3.7.1. Comparative Models and Experimental Setup

To benchmark the performance of the conditional DDPM, the same baseline U-Net architecture was trained in a supervised manner. This model is trained using a direct image-to-image regression framework, where the goal is to directly predict the ground truth velocity field from a given Schlieren image and its corresponding label tensor without any diffusion process. These inputs are processed and injected into the U-Net using the same conditioning mechanisms (FiLM). The output is a single-channel velocity magnitude image, trained directly using MSE (or hybrid) loss. This model is referred to as the Supervised U-Net in later sections, and its performance is evaluated alongside the diffusion-based variants using the same test set and evaluation metrics

3.8. Training Setup

All models were trained using PyTorch 2.5.1 on a workstation with an NVIDIA RTX A6000 GPU. Training was conducted for 200 epochs with a batch size of 32, using a fixed random seed to ensure reproducibility across runs.

The Adam optimizer was used throughout. A Cosine Annealing learning rate scheduler was applied, starting with an initial learning rate of 2^{-5} and decaying to a fixed minimum rate at 10^{-9}.

Gradient clipping with a maximum norm at 1 was applied to ensure numerical stability during training. This helped avoid exploding gradients, particularly in the early training stages. For all diffusion-based models, the number of timesteps in the forward and reverse processes was fixed at 1000, following the original DDPM formulation. This resolution provides a fine-grained sampling trajectory, balancing reconstruction fidelity with computational cost.

4. Results and Discussion

This section presents the performance evaluation of the conditional DDPM models trained for reconstructing high-resolution velocity fields from Schlieren images in supersonic nozzle flows. The results are structured around key experimental variables mentioned in the methodology section.

Both quantitative metrics, MSE and SSIM, are reported across all model variants. The baseline model consists of a cosine-scheduled DDPM trained using MSE loss and single-injection conditioning. All enhancements are benchmarked relative to this configuration unless otherwise stated.

4.1. Noise Scheduler Comparison

DDPMs are sensitive to the choice of noise schedule, as it governs how the structure is progressively degraded during the forward diffusion process. Four noise schedules were evaluated: Linear, Sigmoid, Edge-Aware, and Cosine. All models in this comparison were trained under identical conditions using MSE loss and single-point Schlieren injection, ensuring a fair assessment of the scheduler's influence. The results are summarised in the table below:

Table 2: Quantitative comparison of noise schedules on the test set

Noise Schedule	MSE	SSIM
Linear	0.092	0.751
Sigmoid	0.105	0.729
Edge Aware	0.20	0.733
Cosine	**0.087**	**0.757**

The cosine scheduler achieved the highest overall SSIM and lowest MSE. Its smooth signal-to-noise decay enabled the model to retain structured information over more timesteps,

leading to better reconstructions. The sigmoid schedule exhibited the weakest results, likely due to excessive signal destruction in the mid-range timesteps that impaired effective reverse denoising. The linear schedule showed results closer to the cosine scheduler. The edge-aware noise scheduler, despite its success in unconditional and stroke-to-image generation tasks, [42] underperformed on both MSE and SSIM in our dataset. This suggests that while this anisotropic, edge-aware noising may benefit artistic or loosely structured domains, its effectiveness does not directly translate to our highly structured, physics-driven supersonic flow field reconstruction application.

Based on these findings, the cosine scheduler is adopted as the baseline for all subsequent experiments and model improvements.

Some representative outputs from the baseline model, comparing the predicted velocity magnitude field against the corresponding ground truth sample, are illustrated below.

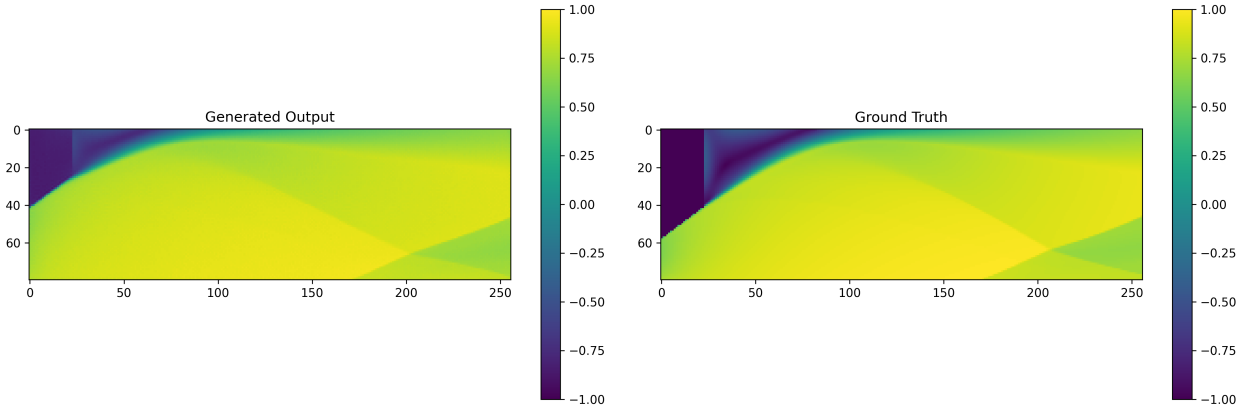


Figure 17: Relatively strong alignment between generated and ground truth velocity fields.

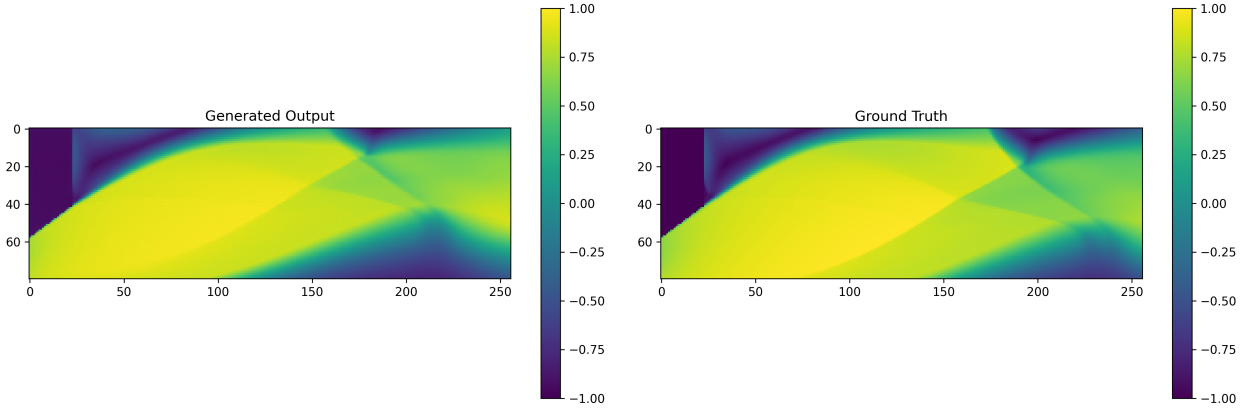


Figure 18: The model broadly captures the flow structure, but slight deviations are visible.

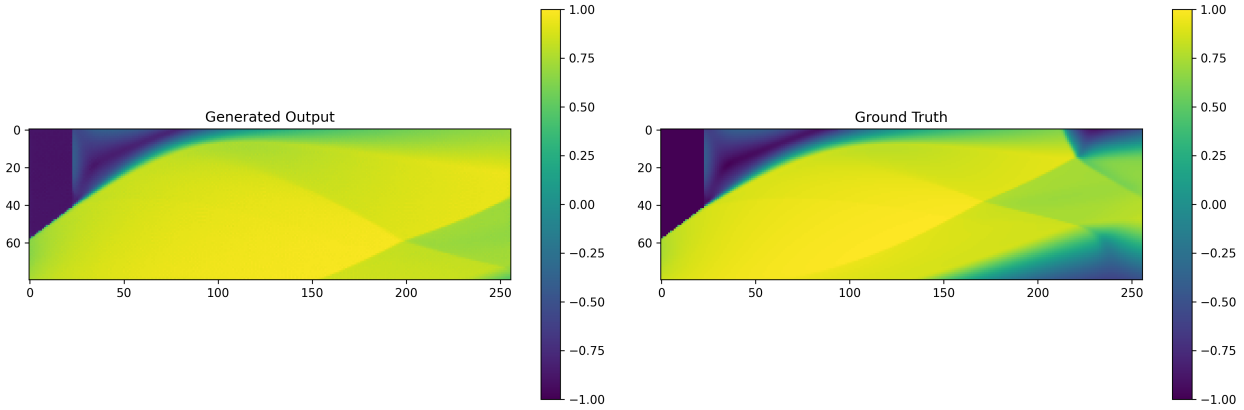


Figure 19: Generated output highlights a failure case of the baseline model.

Three representative samples comparing the predicted and ground truth velocity magnitude fields are shown above. The first example (Figure 17) illustrates a high-fidelity reconstruction, with the generated flow accurately capturing both the global structure and finer shock features. In the second example (Figure 18), while the general flow pattern is recovered, some deviation in shock curvature and intensity can be observed. The third example (Figure 19) highlights a failure case of the model, where the predicted output completely mispredicts the velocity field. These examples demonstrate that the baseline DDPM can learn the dominant trends of compressible flow but still struggles with precise shock localisation in more complex cases.

4.2. Learned Variance and KL Divergence Loss

We evaluated the learned variance formulation using the hybrid loss objective, motivated by its success in natural image domains such as CIFAR-10 and LSUN, where modelling uncertainty has been shown to improve generative quality.

Two values of the KL loss weight (λ) were tested; 0.001 and 0.01.

Table 3: Performance comparison between learned variance using the hybrid: MSE + KL Divergence loss and the baseline model

λ	MSE	SSIM
0.001	0.158	0.653
0.01	0.128	0.689

Baseline	0.087	0.757

Despite its success in other domains, the learned variance model struggled to converge effectively and produced lower-quality reconstructions. This may be attributed to a few factors. Firstly, the hybrid loss introduces a more complex optimisation landscape, which is difficult to balance without precise λ scaling. Also, with a relatively small dataset, the added learning task of estimating variance likely introduces noise without sufficient generalisation benefit.

These findings suggest that a fixed variance model remains a more robust and stable choice for our specific task. Nevertheless, testing this formulation was valuable in understanding the applicability limits of advanced generative modelling strategies in this domain.

Figure 20 shows the generated samples from the learned variance model and the baseline model on the same ground truth to further illustrate this point. Learned variance output lacks sharpness and underestimates gradient strength, particularly around shock regions.

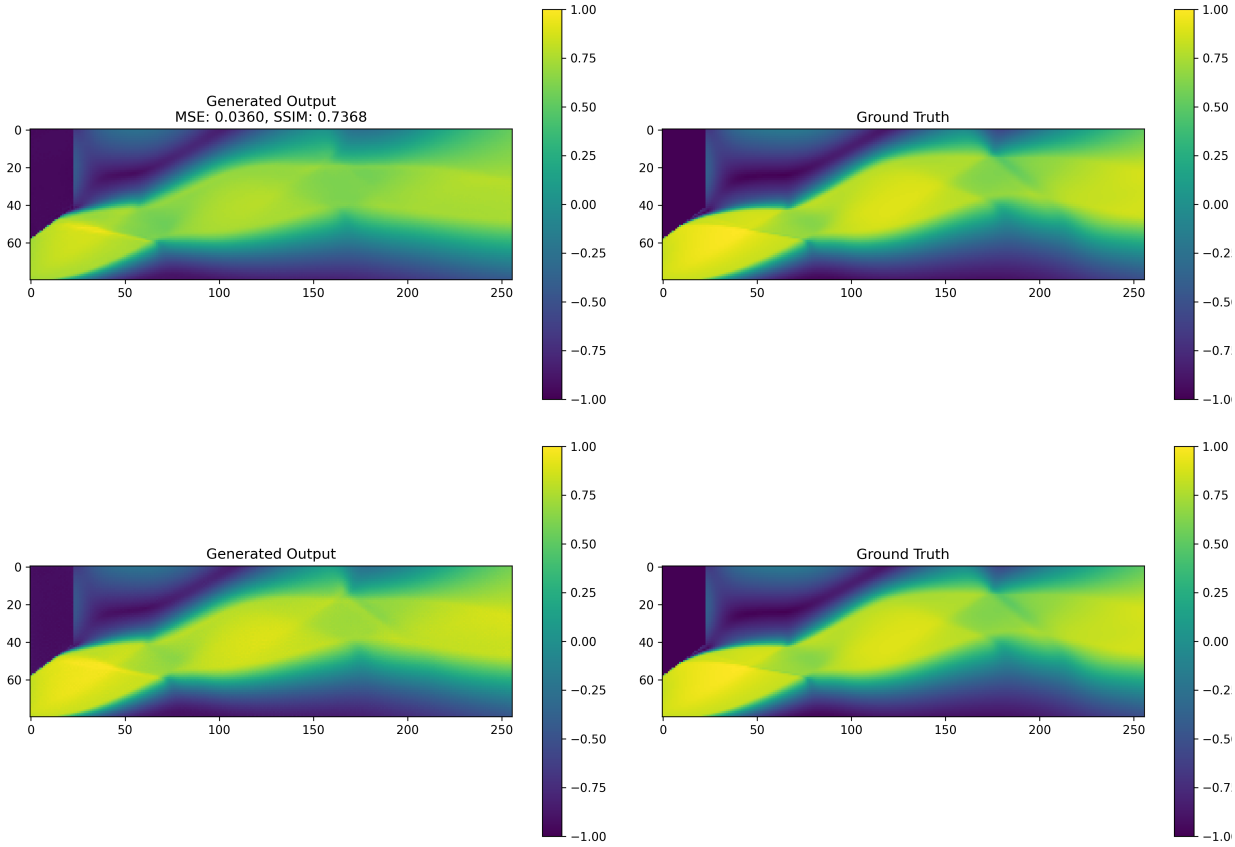


Figure 20: Comparison of a common test sample generated by the baseline model (bottom) and the learned variance model (top).

4.3. Non-Uniform Time Sampling

To further refine model performance**,** non-uniform timestep sampling, where the training timesteps are sampled based on their contribution to the loss rather than uniformly. Table 4 documents the metrics across both strategies. The improvement is modest, but this confirms that non-uniform sampling is a useful optimisation technique after key hyperparameters have been tuned

Table 4: Quantitative comparison of timestep sampling strategies

Sampling Strategy	**MSE**	**SSIM**
Uniform (Baseline)	0.087	0.757
Non-Uniform	0.078	0.765

Interestingly, the per-timestep loss curve (Figure 21) revealed that the highest losses occurred at early timesteps when the input is only slightly noised. This can be explained as follows: the model faces a high signal-to-noise ratio at early timesteps and must predict very subtle noise in mostly intact images. Small prediction errors in this regime contribute significantly to the loss. In contrast, late timesteps involve heavily noised inputs where the model's job is to capture a broader structure, which is comparatively easier. By shifting sampling probability toward these difficult early steps, the model benefits from increased exposure to the most critical regions of the denoising process.

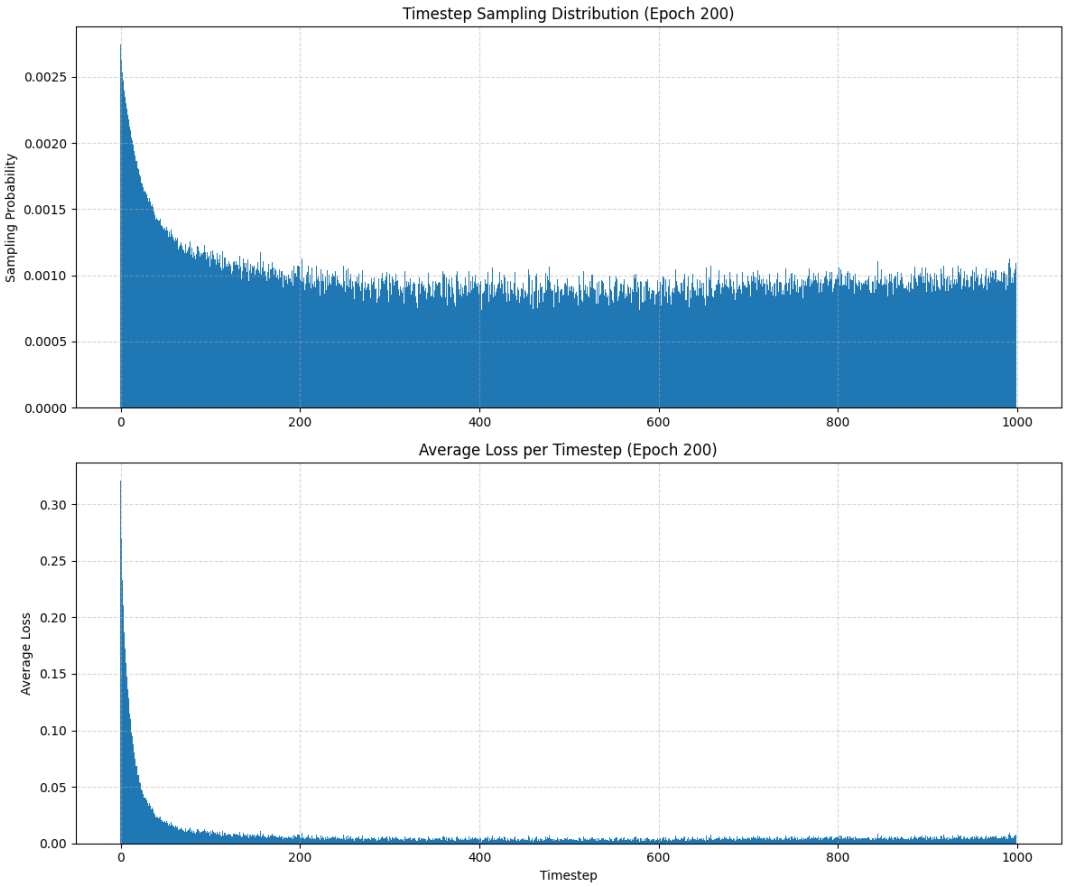


Figure 21: timestep sampling distribution and average loss per timestep at epoch 200. Both curves highlight that early timesteps contribute disproportionately to the total training loss.

4.4. Dataset Modification: Cropping and Augmentation

To investigate the impact of dataset manipulation on model performance, we experimented with two strategies: cropping the spatial domain to reduce input resolution and applying physics-consistent data augmentation techniques such as horizontal flipping and 180° rotation. In the cropping variant, only the left half of each image was retained, reducing the spatial coverage of each sample. These transformations preserve the symmetry of the flow and are valid for centerline-symmetric nozzle configurations.

Table 5: Performance of models trained on modified datasets.

Dataset Variant	MSE	SSIM
Baseline (full resolution)	0.087	0.757
Half Resolution (cropped)	0.055	0.849
Augmented (p=0.2)	0.35	0.519

The cropped dataset improved performance significantly. However, this improvement likely stems not from enhanced generalisation but from a reduction in diversity. Cropped regions tend to contain more uniform, repetitive flow structures across samples. This simplification likely made the learning task easier. Figure 22 further illustrates this point. The selected test sample exhibits greater structural variation than most other cropped samples. In this case, the model continues to show signs of underprediction, particularly in regions surrounding shocks. This suggests that even with reduced resolution, generalisation remains limited for more complex flow scenarios.

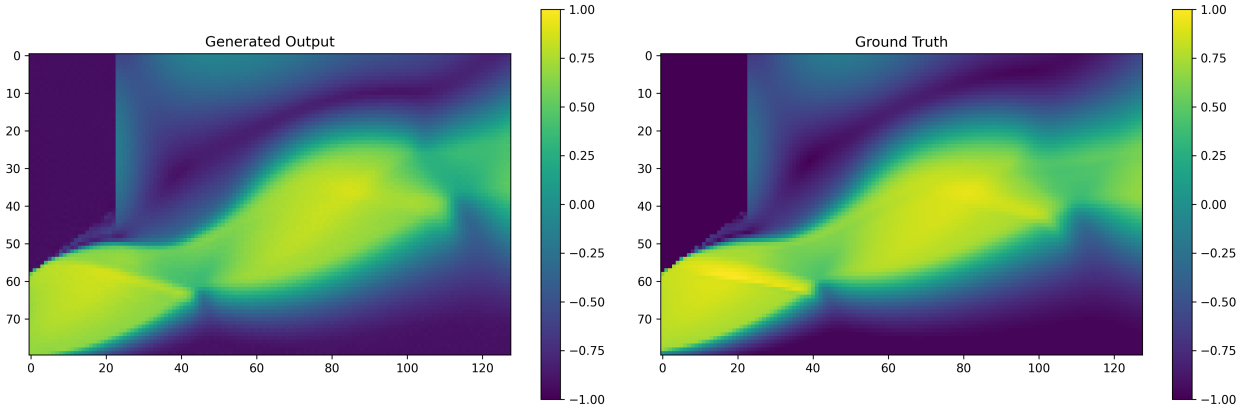


Figure 22: Cropped dataset generated output vs ground truth velocity

Secondly, despite using physically consistent augmentations such as horizontal flips and 180° rotations, the model's performance deteriorated significantly. This suggests that even when the transformations preserve the governing physics, they may disrupt spatial cues or symmetry expectations learned by the model during training. While augmentation is widely used in machine learning to increase dataset diversity, our results suggest that its application in physics-based generative modelling must be handled cautiously, especially when conditioning is involved.

4.5. Injection Strategies: Enhanced Conditional Guidance

Since the model relies heavily on Schlieren images (and boundary condition labels) to guide the generation of velocity fields, improving how this information is injected was a logical next step. We also trained a DDPM using density as the conditioning variable instead of Schlieren to probe this further. Given the close physical relationship between density and velocity, this setup was expected to yield highly accurate predictions. As expected, it achieved a test MSE as low as 0.01. However, even in this favourable scenario, the model still exhibited signs of underprediction and smoothing in high-gradient regions.

This observation reinforced a key insight: the bottleneck wasn't just the conditioning data but rather how the U-Net processes and integrates that information. Thus, we improved the conditioning injection strategy within the U-Net to address this structural limitation. This section explores this.

As described in Section 3.5.2.2, the baseline model injected spatial conditioning information only once at the input layer by concatenating it with the noisy velocity image. While simple, this approach limits the model's ability to leverage conditional signals in deeper layers of the U-Net.

Consequently, a multi-injection strategy was tested, where the same conditioning information is added at multiple resolutions throughout the downsampling layers of the U-Net. This ensures that spatial guidance remains available globally and locally and can influence feature maps at different network depths. The result was a substantial boost in

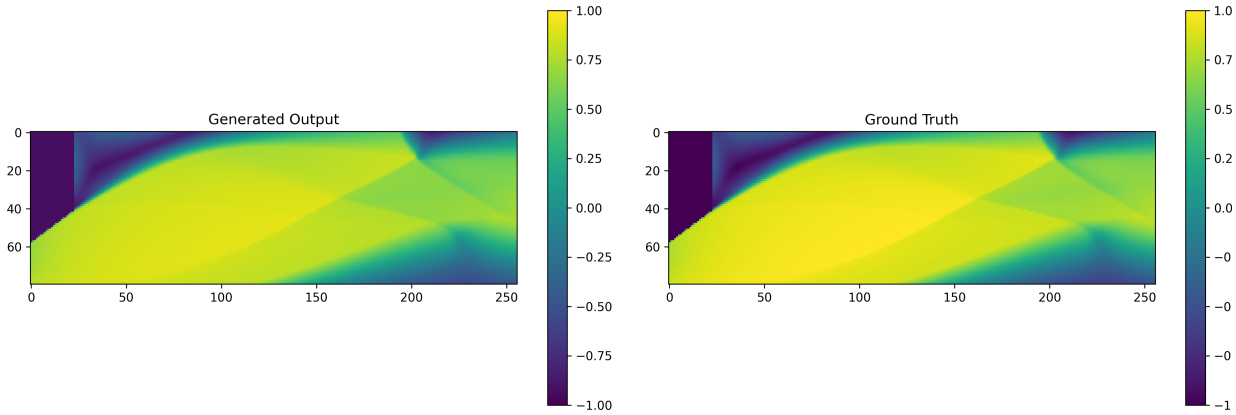
performance, far exceeding earlier improvements from scheduler tweaks or sampling strategies:

Table 6: Performance comparison of conditioning injection strategies

Injection Strategy	MSE	SSIM
Single Injection (baseline)	0.087	0.757
Multi-Injection	0.01	0.940

Figure 23 shows representative samples from the test set to visually assess the improvement brought by multi-injection. These examples highlight how the model can now recover sharper shocks. Compared to baseline predictions, the improvements in structure preservation are clearly visible. In these cases, the baseline model had previously struggled to capture the correct shock shapes or gradient intensities, underscoring the effectiveness of enhanced conditional guidance in complex flow scenarios.

The multi-injection design became the foundation for the final architecture used in subsequent experiments, including those with hybrid loss.



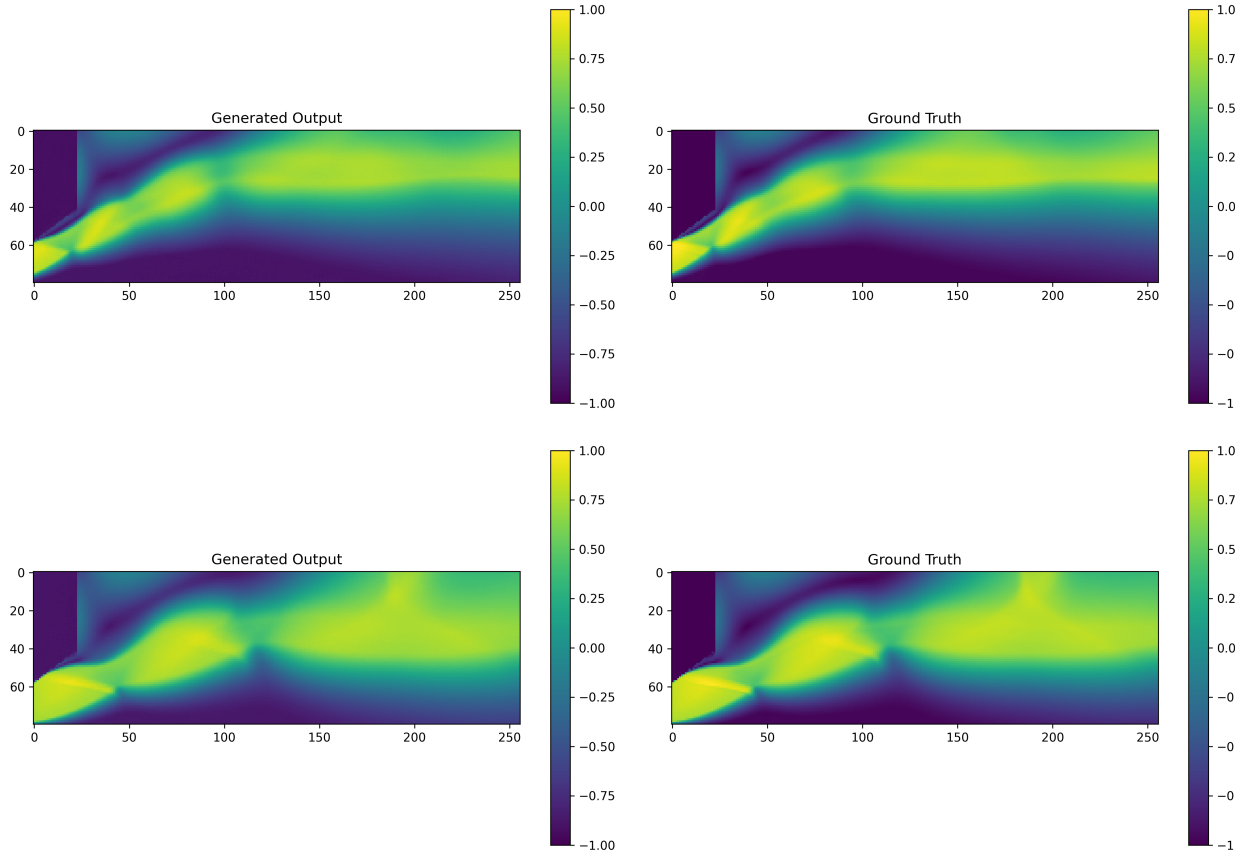


Figure 23: Representative examples from the multi-injection model.

4.6. Hybrid Loss Function:

As outlined in the methodology, a hybrid loss function combining MSE and SSIM was introduced. This approach aimed to further mitigate the smoothing observed when using pure MSE.

A hyperparameter sweep was conducted by varying the MSE weighting factor λ to identify a balance between pixel-wise accuracy and structural fidelity. All models were trained for 100 epochs with the multi-injection architecture. Table 7 summarises the performance across different values of λ.

Table 7: Performance of different hybrid loss weightings using multi-injection architecture

MSE weight λ	Loss Type	MSE	SSIM
1	Pure MSE	0.04	0.820
0.8	MSE Heavy	0.0060	0.923
0.5	Balanced	0.005	0.890
0.2	SSIM Heavy	0.0056	0.879

These results reveal several key insights. Firstly, incorporating SSIM into the loss function improves perceptual quality and leads to a lower MSE than training with pure MSE alone. This suggests that adding structural guidance helps the model converge more effectively, even by pixel-wise metrics.

Secondly, the effect of SSIM weighting is non-linear. While a 50/50 balance already yields substantial gains, increasing the SSIM emphasis further (λ = 0.2) reduces the SSIM score to 0.879. In contrast, a more conservative SSIM weighting (λ = 0.8) produces the best structural quality with SSIM = 0.923 while maintaining a competitive MSE of 0.006.

This suggests that a weighting of approximately 80% MSE and 20% SSIM strikes the best trade-off, offering a small sacrifice in pixel-wise accuracy for a relatively larger gain in structural fidelity.

Finally, we trained a final model using the multi-injection architecture and this hybrid loss configuration (λ = 0.8) for 200 epochs. The resulting model achieved an MSE of 0.004 and SSIM of 0.961. This marked the highest-performing DDPM-based setup for the thesis.

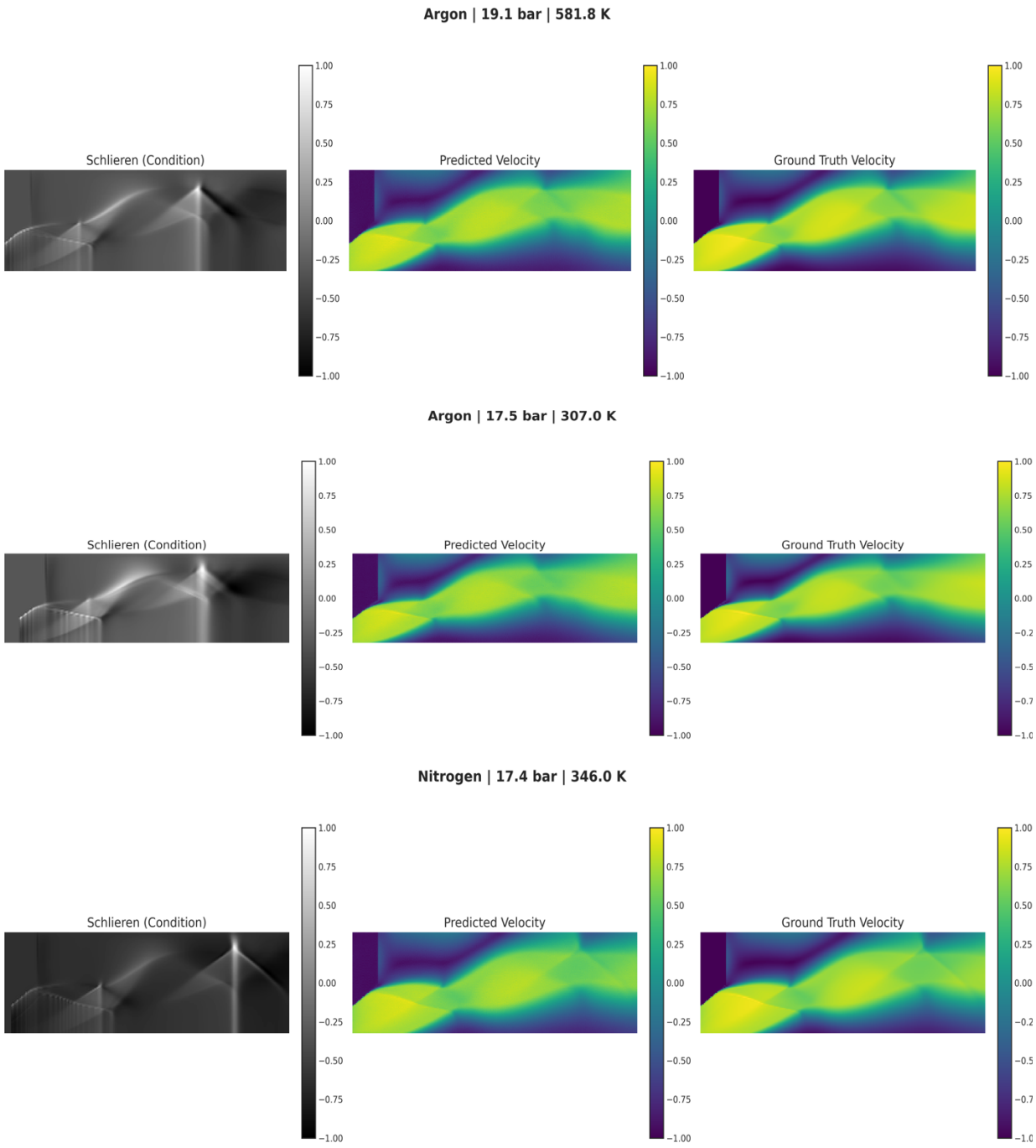


Figure 24: Examples of velocity fields generated by the best-performing DDPM model.

The selected samples plotted in Figure 24 represent some of the best-performing test cases, with SSIM values around 0.97 and MSE below 0.004. The predicted outputs exhibit high structural consistency with the ground truth across all shown cases. The model accurately

captures the number and general position of shock structures, as well as the general morphology of the flow field. This suggests that the DDPM has learned to represent core physical features reliably. Nevertheless, a consistent shortcoming almost across the domain is observed in that the model still slightly underestimates velocity magnitudes. This underestimation is likely a result of the denoising process.

Beyond looking at individual samples, assessing how the model performs across different boundary conditions is also helpful. To investigate this, the SSIM scores from the test set were plotted against inlet pressure and temperature. These visualisations provide a broader view of how consistently the model performs across varying flow conditions.

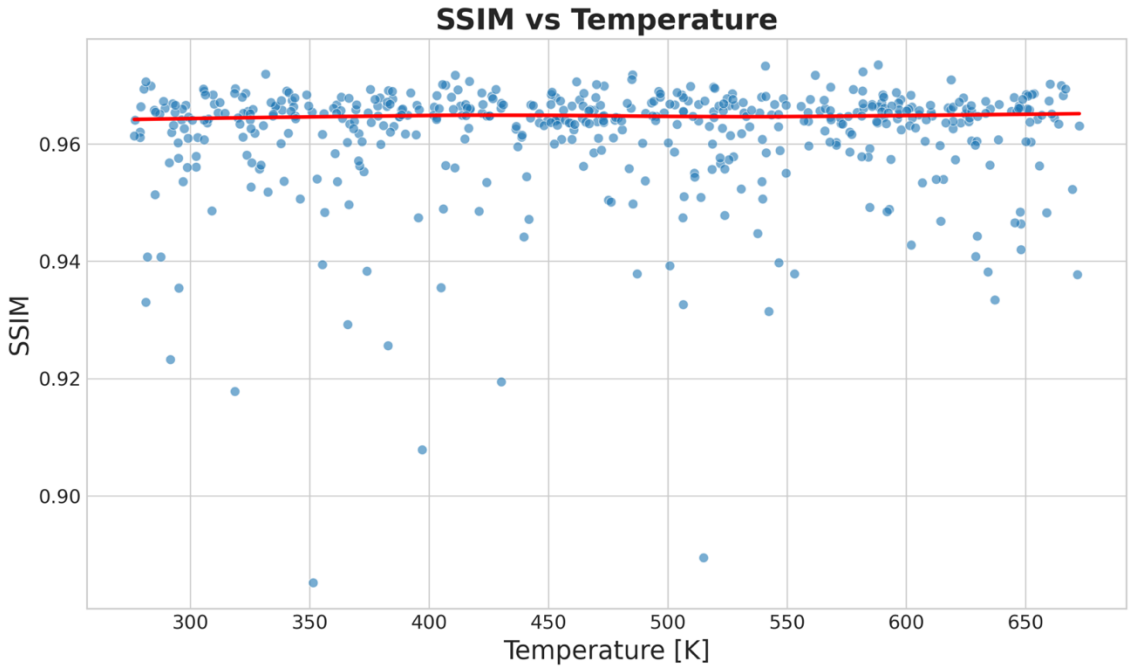


Figure 25: Test set SSIM plotted against temperature for the best-performing DDPM model.

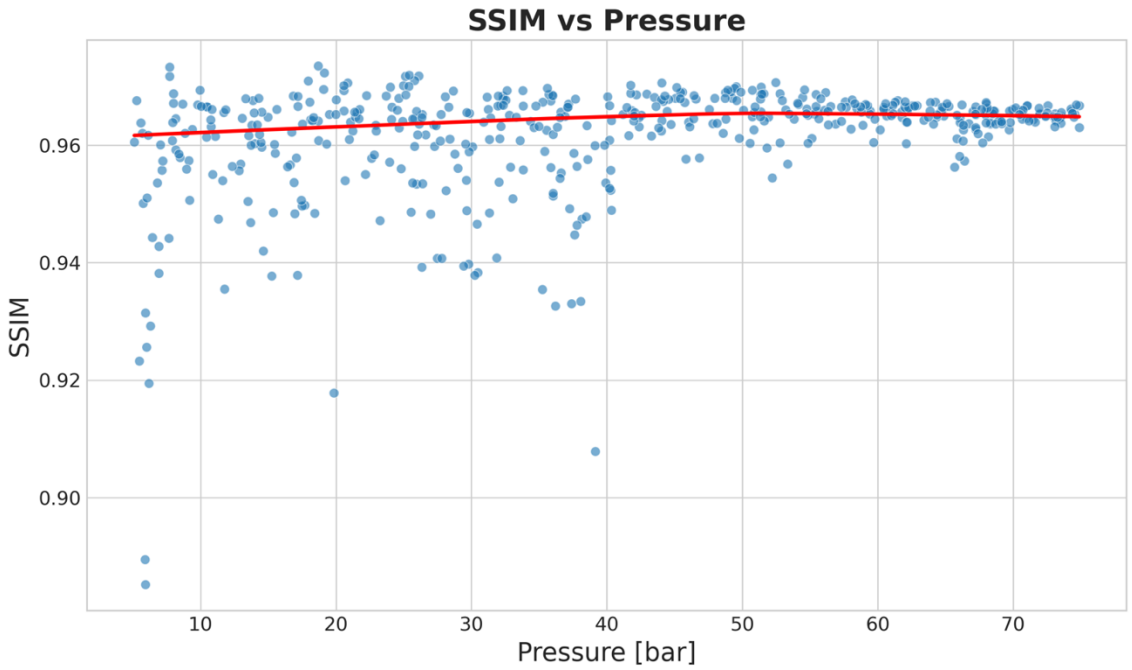


Figure 26: Test set SSIM plotted against temperature for the best-performing DDPM model.

Across the test set, the SSIM values consistently remained between 0.95 and 0.97, indicating stable model performance across a wide range of boundary conditions. The red curve in Figure 25 and Figure 26 represents a locally weighted scatterplot smoothing fit, highlighting overall trends while reducing the impact of local noise. A slight positive trend with pressure was observed, suggesting that higher-pressure cases yield marginally better reconstructions. In contrast, temperature showed minimal influence on model performance, with the SSIM remaining largely flat across the temperature range. A few lower-performing outliers were noted, primarily at lower pressures and temperatures, possibly corresponding to weaker shocks or noisier Schlieren inputs. Overall, the DDPM model generalised well across different conditions without significant degradation.

An example of a case which did poorly on SSIM is further visualised below in Figure 27.

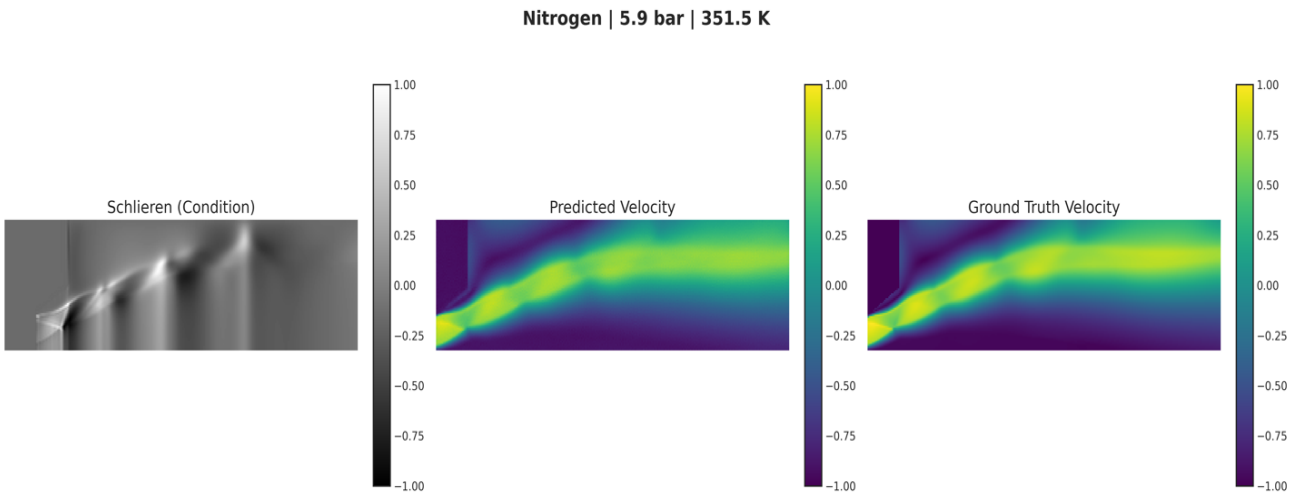


Figure 27: Example of a low SSIM score prediction.

4.7. Spatial Error Analysis

A spatial error analysis on representative samples from the test set was conducted to better understand the model's prediction characteristics. We visualised how prediction errors are distributed across the flow domain by calculating the pixel-wise absolute difference between generated and ground truth velocity fields. Two representative cases are illustrated in Figure 28. The analysis revealed subtle error patterns that might not be immediately apparent through visual inspection of the predictions alone. The most prominent errors appear as thin, bright lines along shock interfaces and flow boundaries, indicating slight misalignments in the predicted shock positions. While hardly detectable to the naked eye when comparing prediction and ground truth directly, these misalignments become evident in the difference maps. They likely stem from a combination of inherent discrepancies between the Schlieren conditioning data and velocity fields, as well as challenges in precisely localising sharp discontinuities. Additionally, we observed a mild but consistent underprediction in the bulk flow regions, visible as a diffuse, low-magnitude error throughout the domain. Despite these patterns, the overall error magnitude remains small (typically below 0.2-0.3 on the normalised scale), explaining the high SSIM scores achieved by the model. These findings offer valuable direction for future refinements, particularly in

enhancing shock position accuracy and addressing the slight systematic underprediction in continuous flow regions.

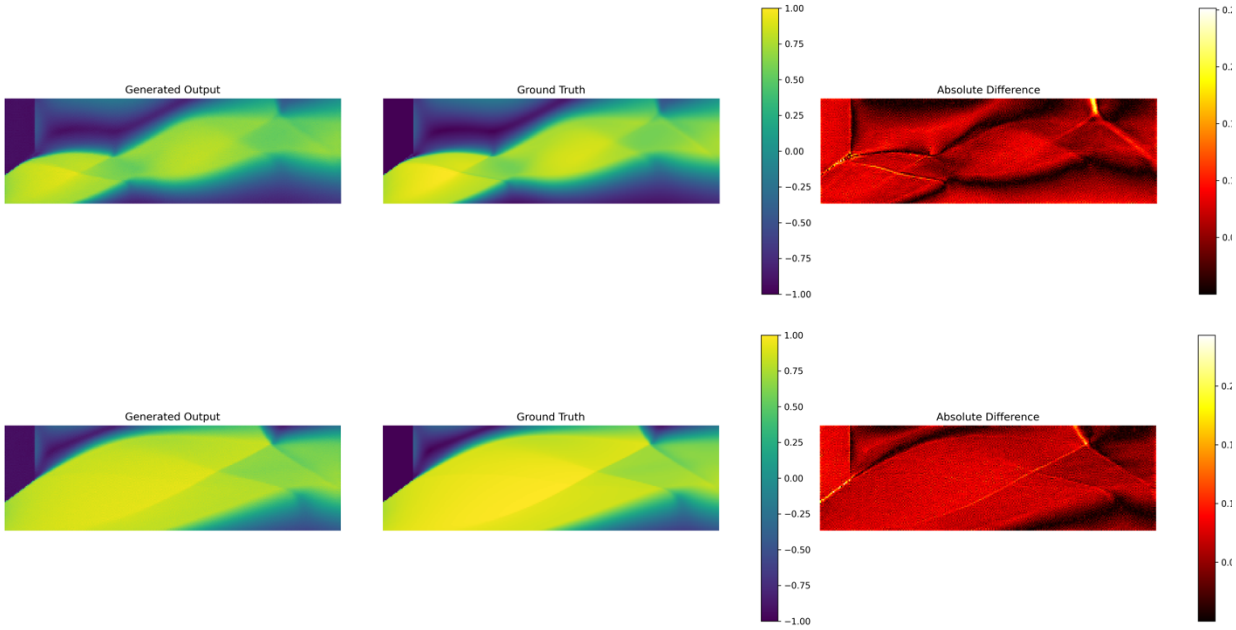


Figure 28: Spatial Error Analysis on select samples. Bright spots indicate higher pixel wise error.

4.8. Variable Transferability: Predicting Density and Mach Number

To evaluate the flexibility of the trained diffusion model, we extended its application beyond velocity prediction. Using the same model architecture, conditioning setup, and optimised hybrid loss, we trained new instances of the DDPM to generate density and Mach number fields. The objective was to assess both qualitative consistency in structure and quantitative performance across these new target variables.

The following figures illustrate that the model performed consistently well across both variables. When trained to predict density and Mach number, the DDPM achieved an average test MSE of approximately 0.003 and SSIM of 0.94 in both cases, comparable to its performance on velocity magnitude. Visual inspection confirmed that key structural features, such as a number of shocks and their locations, were preserved.

These results suggest that the model's learned denoising capacity generalises well to other physically linked variables without requiring architectural changes or reconditioning strategies.

DDPM Generated Mach Number Samples:

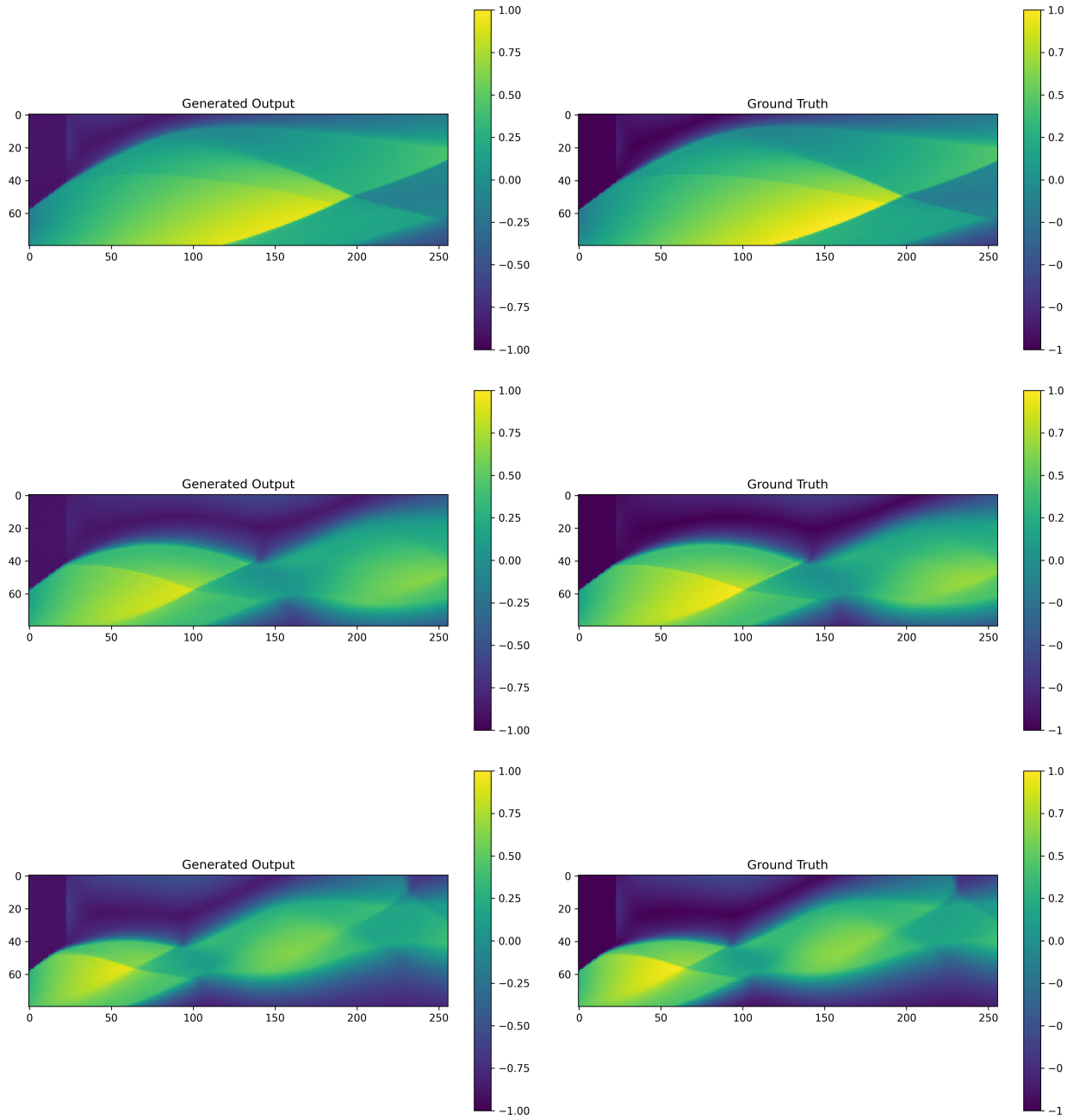


Figure 29: DDPM-generated Mach number fields compared to ground truth.

DDPM Generated Density Samples:

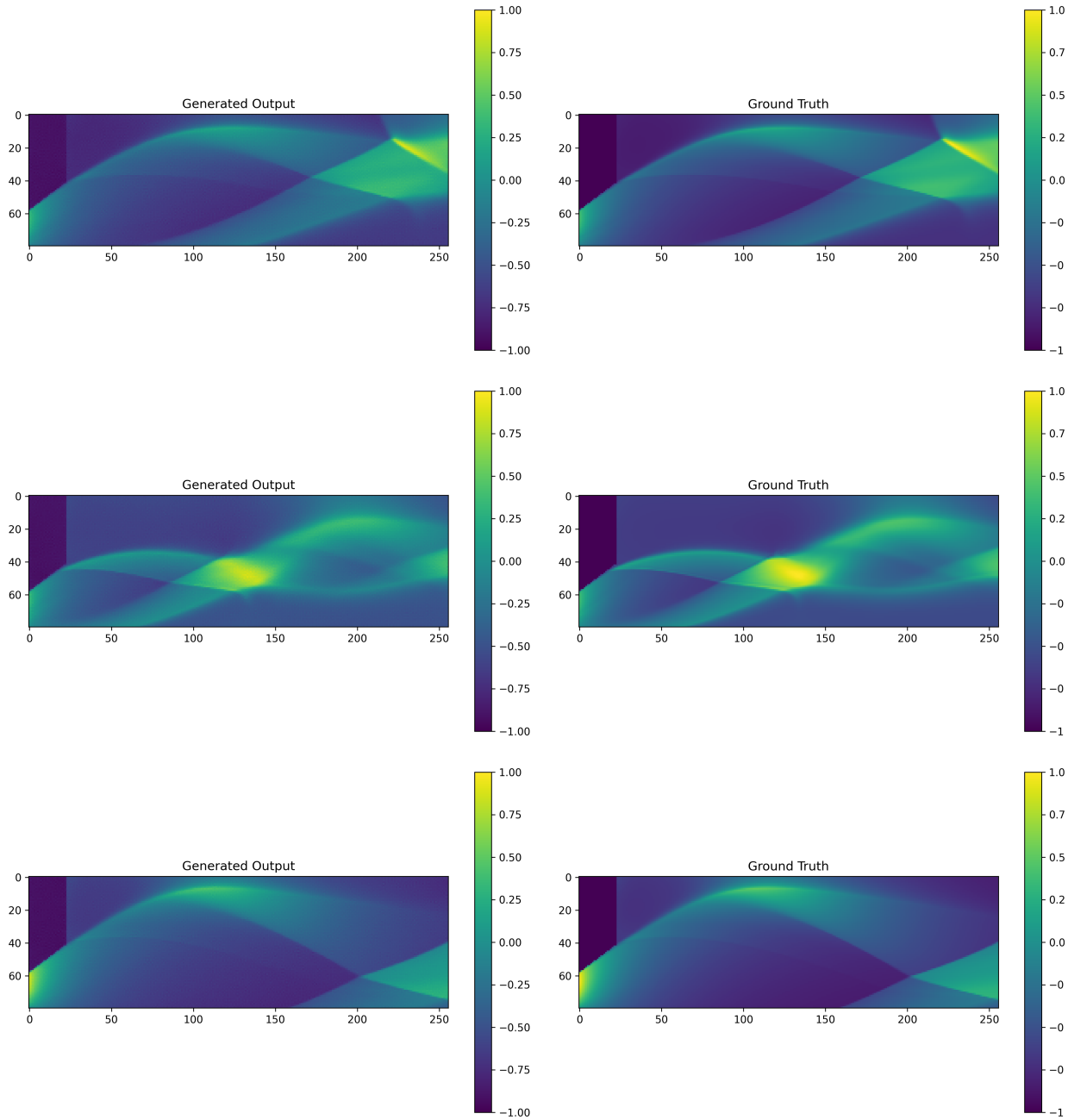


Figure 30: DDPM-generated density fields compared to ground truth.

4.9. Supervised U-Net Benchmark

The U-Net was trained in a supervised manner to directly translate Schlieren images into corresponding velocity magnitude fields to establish a performance benchmark. The architecture remains identical to the U-Net used for noise prediction within the DDPM framework, but all diffusion-specific components were removed. Timestep embeddings were excluded, as they are irrelevant in the supervised setting. Additionally, the Schlieren image, which previously served as a conditioning input, now becomes the primary input to the network. The material tensor was incorporated using the same embedding strategy as in the DDPM setup. The same hybrid loss formulation (MSE + SSIM) was used in a sweep to test the effect of structural loss components. All models were trained for 100 epochs using the same training configurations.

Table 8: Supervised U-Net trained for 100 epochs with SSIM/MSE hybrid loss sweep

MSE Weight	MSE	SSIM
1	0.00030	0.991
0.8	0.00025	0.994
0.5	0.00029	0.993

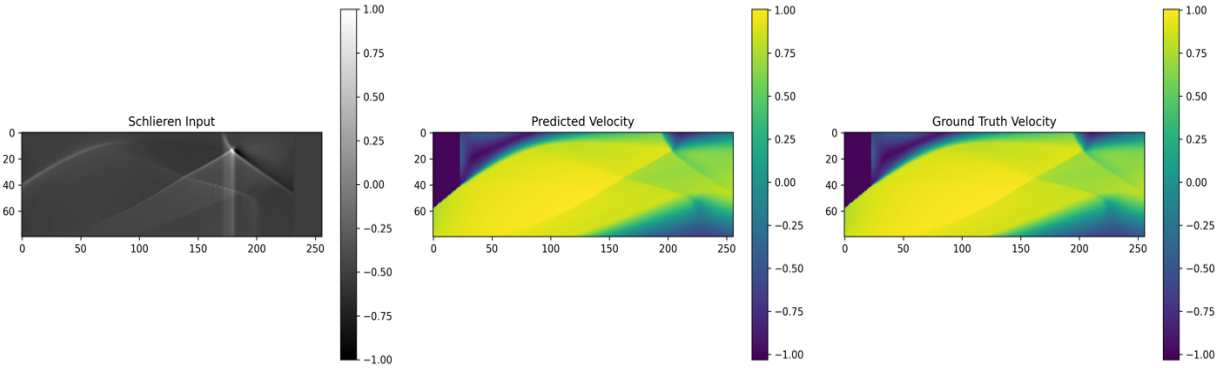
While the effect of hybrid loss tuning is not as pronounced in the supervised U-Net as in the DDPM case, incorporating SSIM still provided marginal improvements. This is likely due to the deterministic nature of supervised learning, where the task is already well-suited to the architecture and input-output mapping. Nevertheless, using a hybrid loss with an MSE weight of 0.8 fully converged with the lowest test MSE (0.00019) and highest SSIM (0.996), outperforming the pure MSE baseline. Interestingly, this also mirrors the optimal λ value observed in the DDPM model. Thus, assigning approximately 80% weight to MSE in the hybrid loss appears to be a consistently effective strategy for this specific task and dataset across both generative and supervised settings.

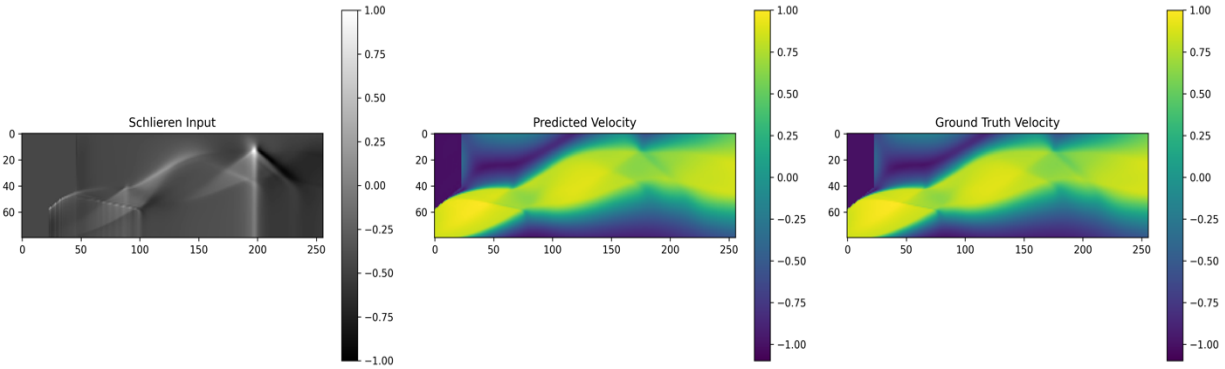
These results show the strength of deterministic supervised learning in small data regimes. With around 5,000 samples, the supervised U-Net performs very well (SSIM 0.994),

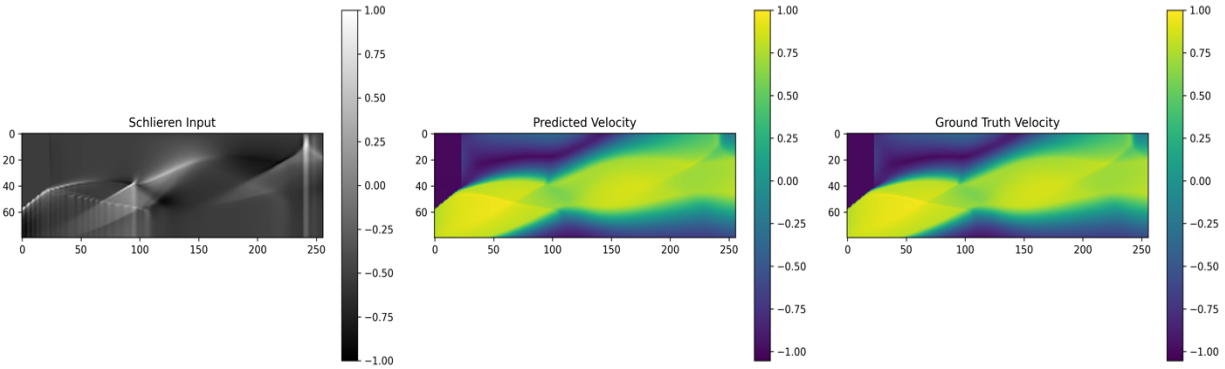
leveraging the direct mapping between Schlieren input and velocity output. In contrast, the DDPM model, despite all variations, performs slightly worse on both metrics. This discrepancy can be attributed to the fundamentally different learning objectives: while supervised models learn a deterministic mapping, DDPMs must capture the entire underlying data distribution and generative process, including inherent uncertainties.

However, it is to be seen if this gap would likely narrow or reverse with substantially larger datasets, where DDPMs can leverage their generative nature better. In this particular context, the supervised model establishes an effective benchmark for the upper bound of achievable performance on the current dataset.

The following examples illustrate predictions made by the supervised U-Net model and further support these observations.


Schlieren Input
Predicted Velocity
Ground Truth Velocity


Schlieren Input
Predicted Velocity
Ground Truth Velocity


Schlieren Input
Predicted Velocity
Ground Truth Velocity

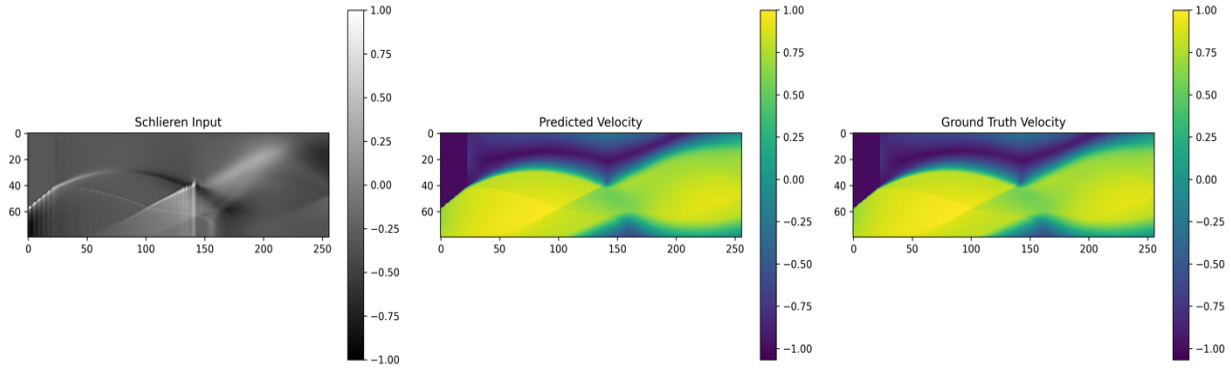


Figure 31:Supervised U-Net Velocity Magnitude predictions.

To further understand how the generated outputs relate to the ground truth, an embedding-based analysis was performed. Feature embeddings were extracted using a pre-trained ResNet18 model (img2vec-pytorch library), resulting in 512-dimensional vectors for each image. Principal Component Analysis (PCA) was then applied to reduce the embeddings to two dimensions for visualization purposes.

Figure 32 shows the resulting PCA projection of the ground truth, DDPM-generated, and supervised U-Net predictions. The supervised U-Net predictions (green) cluster tightly around the ground truth samples (blue), reinforcing the earlier observation of strong structural and visual fidelity. The DDPM outputs (red) in some cases have distinct clusters but still overlap significantly with the ground truth distribution, indicating that the model has learned the underlying data manifold despite the generative uncertainty. This analysis complements the quantitative metrics and confirms that, under the current dataset size, supervised learning achieves the most deterministic mapping, while diffusion-based generation remains a competitive alternative with greater flexibility for scaling to larger, more complex datasets.

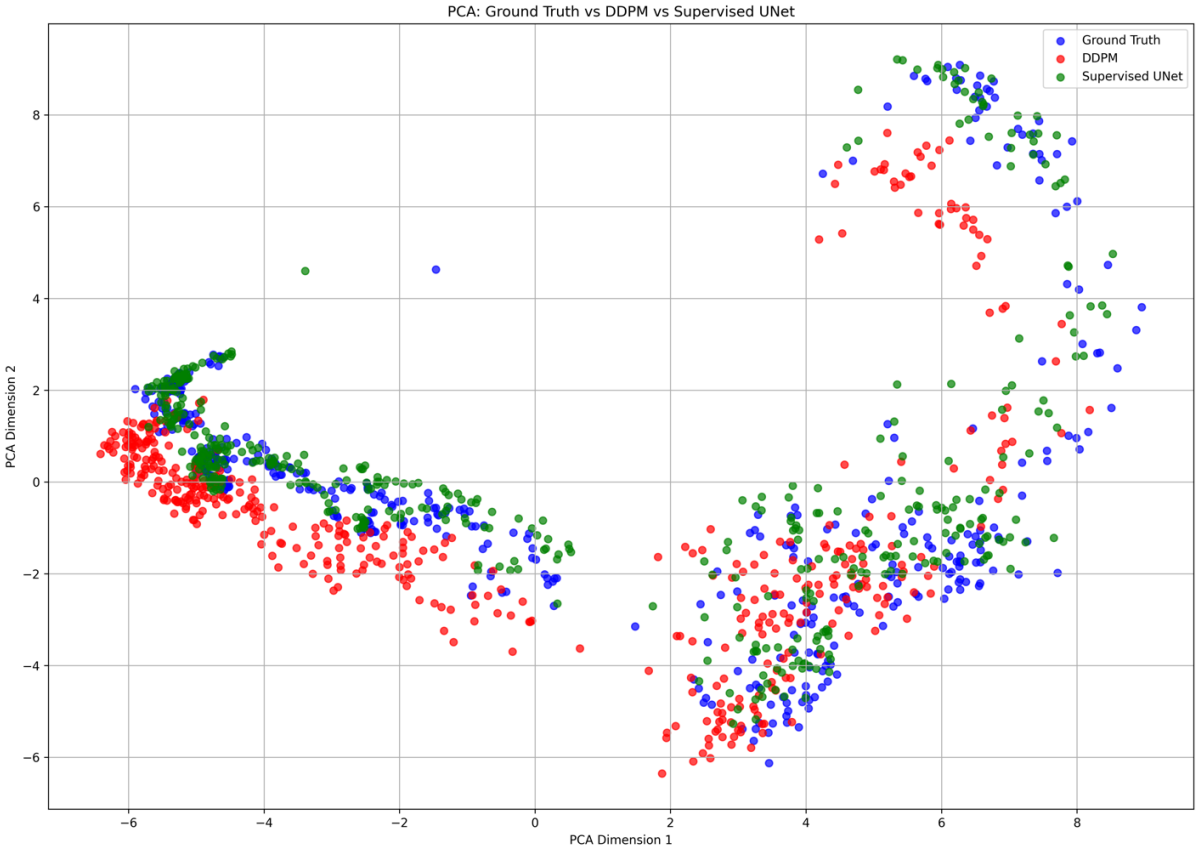


Figure 32: PCA projection of feature embeddings extracted from ground truth, DDPM-generated, and supervised U-Net output.

5. Conclusion

This thesis investigated the application of Conditional DDPMs for reconstructing high-fidelity supersonic flow fields from Schlieren images, addressing the computational bottlenecks in traditional CFD approaches. The research demonstrates that data-driven generative models can offer promising alternatives for flow field reconstruction in complex supersonic regimes.

Through systematic experimentation, we developed a DDPM framework conditioned on Schlieren images and boundary conditions to accurately predict velocity magnitude fields. Our exploration revealed several key findings: (1) cosine noise scheduling significantly outperforms other scheduling methods for supersonic flow applications; (2) multi-injection conditioning dramatically improves reconstruction quality by integrating spatial information across multiple network depths; and (3) a hybrid MSE-SSIM loss function with an 80-20 weighting strikes an optimal balance between pixel-wise accuracy and structural fidelity. The best-performing DDPM configuration achieved a test set SSIM of 0.961 and an MSE of 0.004, demonstrating the viability of diffusion models for this challenging task.

Qualitative analyses, including spatial error mapping, revealed that while the model effectively captures the overall flow structure and shock positions, it exhibits subtle misalignments at shock interfaces and a consistent slight underprediction across the flow field. Nevertheless, the model demonstrated robust generalisation across varying inlet pressures and temperatures, with consistent performance metrics throughout the parameter space. Additionally, the framework showed good flexibility in predicting related variables like density and Mach number without any modifications, suggesting its potential as a versatile tool for multiple flow field quantities.

Comparisons with a supervised U-Net setup were conducted. The supervised approach achieved superior metrics (SSIM of 0.994 and MSE of 0.0002), suggesting that in data-constrained regimes like our 5,000-sample dataset, supervised learning may be preferred

for deterministic applications. At the same time, diffusion models may likely hold greater promise in larger-scale applications with more extensive datasets.

Future work could address the observed limitations, particularly the slight underprediction across flow fields and misalignments at shock interfaces. Exploring physics-informed constraints, larger and more diverse training datasets, and advanced conditioning mechanisms could further enhance performance. Additionally, investigating the scaling behaviour of diffusion models with substantially larger datasets would be valuable in determining the crossover point where their generative advantages might surpass supervised approaches.

6. References

[1] J.D. Anderson, Computational Fluid Dynamics - The Basics With Applications, McGraw Hill Education, 1995.

[2] J.N. Sørensen, W.Z. Shen, Numerical Modeling of Wind Turbine Wakes, Journal of Fluids Engineering 124 (2002) 393–399. https://doi.org/10.1115/1.1471361.

[3] E. Kalnay, Atmospheric Modeling, Data Assimilation and Predictability, Higher Education from Cambridge University Press (2002). https://doi.org/10.1017/CBO9780511802270.

[4] [2102.01010] Machine learning accelerated computational fluid dynamics, (n.d.). https://ar5iv.labs.arxiv.org/html/2102.01010 (accessed March 4, 2025).

[5] H. Choi, P. Moin, Grid-point requirements for large eddy simulation: Chapman's estimates revisited, Physics of Fluids 24 (2012) 011702. https://doi.org/10.1063/1.3676783.

[6] J.P. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, D.J. Mavriplis, CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences, 2014. https://ntrs.nasa.gov/citations/20140003093 (accessed March 3, 2025).

[7] gestione, Turbulence models in CFD - RANS, DES, LES and DNS, IdealSimulations (2019). https://www.idealsimulations.com/resources/turbulence-models-in-cfd/ (accessed March 18, 2025).

[8] R. Vinuesa, S.L. Brunton, Enhancing Computational Fluid Dynamics with Machine Learning, Nat Comput Sci 2 (2022) 358–366. https://doi.org/10.1038/s43588-022-00264-7.

[9] D. Shu, Z. Li, A. Barati Farimani, A physics-informed diffusion model for high-fidelity flow field reconstruction, Journal of Computational Physics 478 (2023) 111972. https://doi.org/10.1016/j.jcp.2023.111972.

[10] H. Liang, Z. Song, C. Zhao, X. Bian, Continuous and discontinuous compressible flows in a converging–diverging channel solved by physics-informed neural networks without exogenous data, Sci Rep 14 (2024) 3822. https://doi.org/10.1038/s41598-024-53680-2.

[11] J.D. Anderson, Fundamentals of Aerodynamics, McGraw-Hill Education, 2017.

[12] P. Singh, N. Gopalaswamy, F. Martınez, ACCELERATING CFD USING MACHINE LEARNING, (n.d.).

[13] K. Taira, S.L. Brunton, S.T.M. Dawson, C.W. Rowley, T. Colonius, B.J. McKeon, O.T. Schmidt, S. Gordeyev, V. Theofilis, L.S. Ukeiley, Modal Analysis of Fluid Flows: An Overview, (2017). https://doi.org/10.48550/arXiv.1702.01453.

[14] N. Thuerey, K. Weissenow, L. Prantl, X. Hu, Deep Learning Methods for Reynolds-Averaged Navier-Stokes Simulations of Airfoil Flows, AIAA Journal 58 (2020) 25–36. https://doi.org/10.2514/1.j058291.

[15] K. Fukami, K. Fukagata, K. Taira, Super-resolution reconstruction of turbulent flows with machine learning, J. Fluid Mech. 870 (2019) 106–120. https://doi.org/10.1017/jfm.2019.238.

[16] D. Kochkov, J.A. Smith, A. Alieva, Q. Wang, M.P. Brenner, S. Hoyer, Machine learning accelerated computational fluid dynamics, Proc. Natl. Acad. Sci. U.S.A. 118 (2021) e2101784118. https://doi.org/10.1073/pnas.2101784118.

[17] J. Pathak, S. Subramanian, P. Harrington, S. Raja, A. Chattopadhyay, M. Mardani, T. Kurth, D. Hall, Z. Li, K. Azizzadenesheli, P. Hassanzadeh, K. Kashinath, A. Anandkumar, FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators, (2022). https://doi.org/10.48550/arXiv.2202.11214.

[18] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative Adversarial Networks, (2014). https://doi.org/10.48550/arXiv.1406.2661.
[19] (PDF) Generative Adversarial Networks for Synthetic Data Generation: A Comparative Study, ResearchGate (n.d.). https://doi.org/10.48550/arXiv.2112.01925.
[20] Y. Xie, E. Franz, M. Chu, N. Thuerey, tempoGAN: A Temporally Coherent, Volumetric GAN for Super-resolution Fluid Flow, ACM Trans. Graph. 36 (2017) 1–14. https://doi.org/10.1145/3072959.3073643.
[21] T. Li, M. Buzzicotti, L. Biferale, F. Bonaccorso, Generative adversarial networks to infer velocity components in rotating turbulent flows, Eur. Phys. J. E 46 (2023) 31. https://doi.org/10.1140/epje/s10189-023-00286-7.
[22] M. Buzzicotti, F. Bonaccorso, P.C. Di Leoni, L. Biferale, Reconstruction of turbulent data with deep generative models for semantic inpainting from TURB-Rot database, Phys. Rev. Fluids 6 (2021) 050503. https://doi.org/10.1103/PhysRevFluids.6.050503.
[23] Y. Kossale, M. Airaj, A. Darouichi, Mode Collapse in Generative Adversarial Networks: An Overview, in: 2022 8th International Conference on Optimization and Applications (ICOA), 2022: pp. 1–6. https://doi.org/10.1109/ICOA55659.2022.9934291.
[24] B. Kim, V.C. Azevedo, N. Thuerey, T. Kim, M. Gross, B. Solenthaler, Deep Fluids: A Generative Network for Parameterized Fluid Simulations, Computer Graphics Forum 38 (2019) 59–70. https://doi.org/10.1111/cgf.13619.
[25] W. Zhong, H. Meidani, PI-VAE: Physics-Informed Variational Auto-Encoder for stochastic differential equations, Computer Methods in Applied Mechanics and Engineering 403 (2023) 115664. https://doi.org/10.1016/j.cma.2022.115664.
[26] C. Dong, C.C. Loy, K. He, X. Tang, Image Super-Resolution Using Deep Convolutional Networks, (2015). https://doi.org/10.48550/arXiv.1501.00092.
[27] J. Yamanaka, S. Kuwashima, T. Kurita, Fast and Accurate Image Super Resolution by Deep CNN with Skip Connection and Network in Network, (2020). https://doi.org/10.48550/arXiv.1707.05425.
[28] Z. Lu, Y. Chen, Single Image Super Resolution based on a Modified U-net with Mixed Gradient Loss, (2019). https://doi.org/10.48550/arXiv.1911.09428.
[29] K. Fukami, K. Fukagata, K. Taira, Super-resolution analysis via machine learning: a survey for fluid flows, Theor. Comput. Fluid Dyn. 37 (2023) 421–444. https://doi.org/10.1007/s00162-023-00663-0.
[30] X. Jin, S. Cai, H. Li, G.E. Karniadakis, NSFnets (Navier-Stokes Flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations, Journal of Computational Physics 426 (2021) 109951. https://doi.org/10.1016/j.jcp.2020.109951.
[31] O. Obiols-Sales, A. Vishnu, N. Malaya, A. Chandramowlishwaran, SURFNet: Super-resolution of Turbulent Flows with Transfer Learning using Small Datasets, (2021). https://doi.org/10.48550/arXiv.2108.07667.
[32] J. Ho, A. Jain, P. Abbeel, Denoising Diffusion Probabilistic Models, (2020). https://doi.org/10.48550/arXiv.2006.11239.
[33] FIGURE 4. Denoising diffusion probabilistic model structure., ResearchGate (n.d.). https://www.researchgate.net/figure/Denoising-diffusion-probabilistic-model-structure_fig2_380559293 (accessed April 9, 2025).
[34] Z. Xiao, K. Kreis, A. Vahdat, Tackling the Generative Learning Trilemma with Denoising Diffusion GANs, (2022). https://doi.org/10.48550/arXiv.2112.07804.

[35] P. Dhariwal, A. Nichol, Diffusion Models Beat GANs on Image Synthesis, (2021). https://doi.org/10.48550/arXiv.2105.05233.
[36] J. Peng, R.L.J. Qiu, J.F. Wynne, C.-W. Chang, S. Pan, T. Wang, J. Roper, T. Liu, P.R. Patel, D.S. Yu, X. Yang, CBCT-Based Synthetic CT Image Generation Using Conditional Denoising Diffusion Probabilistic Model, Medical Physics 51 (2024) 1847–1859. https://doi.org/10.1002/mp.16704.
[37] (PDF) The Influence of Preparation Parameters on the Morphology and Magnetic Properties of Fe-N Powders Obtained by the Gas Atomization Method, ResearchGate (2025). https://doi.org/10.3390/app132011529.
[38] X. Zhang, Dataset Generation and Denoising Diffusion Probabilistic Model Training for Supersonic Flow Field Prediction in an Atomization Nozzle, Master Thesis, Technical University of Munich, n.d.
[39] The Uncanny Valley: A Comprehensive Analysis of Diffusion Models, (n.d.). https://arxiv.org/html/2402.13369v1 (accessed March 27, 2025).
[40] A Comprehensive Review on Noise Control of Diffusion Model, (n.d.). https://arxiv.org/html/2502.04669v1 (accessed March 27, 2025).
[41] A. Nichol, P. Dhariwal, Improved Denoising Diffusion Probabilistic Models, (2021). https://doi.org/10.48550/arXiv.2102.09672.
[42] J. Vandersanden, S. Holl, X. Huang, G. Singh, Edge-preserving noise for diffusion models, (2024). https://doi.org/10.48550/arXiv.2410.01540.
[43] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, in: N. Navab, J. Hornegger, W.M. Wells, A.F. Frangi (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer International Publishing, Cham, 2015: pp. 234–241. https://doi.org/10.1007/978-3-319-24574-4_28.
[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention Is All You Need, (2023). https://doi.org/10.48550/arXiv.1706.03762.
[45] E. Perez, F. Strub, H. de Vries, V. Dumoulin, A. Courville, FiLM: Visual Reasoning with a General Conditioning Layer, (2017). https://doi.org/10.48550/arXiv.1709.07871.