Uncertainty-aware Surrogate Models for Airfoil Flow Simulations with Denoising Diffusion Probabilistic Models

Qiang Liu [1] **Nils Thuerey** [1]

Abstract

Leveraging neural networks as surrogate models for turbulence simulation is a topic of growing interest. At the same time, embodying the inherent uncertainty of simulations in the predictions of surrogate models remains very challenging. The present study makes a first attempt to use denoising diffusion probabilistic models (DDPMs) to train an uncertainty-aware surrogate model for turbulence simulations. Due to its prevalence, the simulation of flows around airfoils with various shapes, Reynolds numbers, and angles of attack is chosen as the learning objective. Our results show that DDPMs can successfully capture the whole distribution of solutions and, as a consequence, accurately estimate the uncertainty of the simulations. The performance of DDPMs is also compared with varying baselines in the form of Bayesian neural networks and heteroscedastic models. Experiments demonstrate that DDPMs outperform the other methods in terms of a variety of accuracy metrics. Besides, it offers the advantage of providing access to the complete distributions of uncertainties rather than providing a set of parameters. As such, it can yield realistic and detailed samples from the distribution of solutions.

1. Introduction

From fuel combustion in car engines (Han & Reitz, 1995; Lumley, 2001) to supersonic flow around aircraft airfoils (Nieuwland & Spee, 1973; Drela & Giles, 1987), turbulence is ubiquitous in modern engineering. Despite advancements in computing power, simplified turbulence models like Reynolds-averaged Navier–Stokes (RANS) (Alfonsi, 2009b) and Large Eddy Simulations (LES) (Georgiadis et al., 2010) remain prevalent (Argyropoulos & Markatos, 2015). These models introduce uncertainties through hypotheses and parameters (Duraisamy et al., 2019), necessitating methods to estimate and mitigate these uncertainties (Iaccarino et al., 2017; Mishra et al., 2019; Wang et al., 2016; Xiao et al., 2017; Najm, 2009; Roberts et al., 2011).

Recently, deep learning techniques become popular in fluid dynamics research (Brunton et al., 2020; Vinuesa & Brunton, 2022; Lino et al., 2023). It demonstrated promising capabilities as surrogate models for turbulent phenomena (Thuerey et al., 2020; Chen et al., 2021; Sabater et al., 2022; Chen & Thuerey, 2023). Considering the inherent uncertainty of the underlying simulations, the prediction of surrogate models should encompasses a *probabilistic distribution* containing all possible solutions rather than a single-point estimation for the simulation result. Bayesian inference (George E.P. Box, 1992) is effective for probabilistic predictions. Directly employing a neural network as a surrogate model within Bayesian inference gives rise to Bayesian Neural Networks (BNNs) (Denker & LeCun, 1990; MacKay, 1992; Neal, 1996; Wang & Yeung, 2020). BNNs perform posterior sampling based on a prior distribution of the network parameters. While it has been used in fluid simulations study (Tang et al., 2023; Qiu et al., 2023; Geneva & Zabaras, 2019; Sun & Wang, 2020), subtle distinctions endure in the uncertainty it captures compared to the inherent uncertainty in target simulations.

Researchers often consider two kinds of uncertainty: (data) and epistemic (model) uncertainty (Duraisamy et al., 2019; Paté-Cornell, 1996; Kiureghian & Ditlevsen, 2009; Hüllermeier & Waegeman, 2021). Aleatoric uncertainty, caused by data noise, can only be reduced by obtaining better data, while epistemic uncertainty, due to model limitations, can be reduced with better models. In turbulence simulation, most uncertainty is epistemic. However, this uncertainty becomes aleatoric when simulation data is used for training surrogate models. Although fully disentangling aleatoric and epistemic uncertainty is hard in Bayesian deep learning, it is important to note that BNNs construct probabilistic distributions on the network parameters, aiming to capture the epistemic uncertainty inherent in the neural net-

[1]School of Computation, Information, and Technology, Technical University of Munich, Garching, Germany. Correspondence to: Nils Thuerey <nils.thuerey@tum.de>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

work rather than the aleatoric uncertainty of the simulation-generated dataset (Hüllermeier & Waegeman, 2021). It is thus challenging to use the uncertainty of the prediction of a BNN to directly represent the inherent uncertainty of the simulation, as will be shown in the present study. In parallel, methods like mixture density networks (Bishop, 1994) and heteroscedastic models (Nix & Weigend, 1994; Kendall & Gal, 2017) which estimate aleatoric uncertainty and are shown to be effective in fluid dynamics study (Maulik et al., 2020).

Meanwhile, conventional generative models, including GANs (Goodfellow et al., 2014) and VAEs (Kingma & Welling, 2014), sample from latent spaces for predictions. Linking latent space sampling to posterior sampling provides a new potential solution for assessing the prediction uncertainty through generative methods (Abdar et al., 2021). However, these approaches were shown to have problems generating details and covering whole distributions of solutions (El-Kaddoury et al., 2019; Creswell et al., 2018). Recently, denoising diffusion probabilistic models (DDPMs) (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020) have outperformed previous generative methods in various contexts (Dhariwal & Nichol, 2021; Rombach et al., 2022a). Despite the vibrant developments in many other research areas like material design (Xie et al., 2022; Luo et al., 2022) and medical image reconstruction (Chung & Ye, 2022; Peng et al., 2022), only very few studies in fluid dynamics have employed DDPM. Exceptions are works that investigate the performance of DDPM for super-resolution tasks (Shu et al., 2023) and inverse problem solving (Holzschuh et al., 2023), while the capabilities of DDPMs as a surrogate model in fluid dynamics have not been investigated.

In the current study, we leverage DDPMs to train an uncertainty-aware surrogate model for RANS-based airfoil flow simulations. Simulations of airfoil flow with RANS turbulence models are a fundamental problem and a widely studied use case of turbulence research (Thuerey et al., 2020; Hui et al., 2020; Sun et al., 2021; Yang et al., 2022; Duru et al., 2022; Chen & Thuerey, 2023). As such, they provide a very good basis for assessing the capabilities of DDPM. The uncertainty considered in the present study is represented by a *distribution* of solutions that encapsulates the inherent unpredictability associated with the RANS model. We compare the performance of DDPMs with varying baselines like BNNs and heteroscedastic models. The capabilities of DDPMs and other baseline methods are measured in terms of their ability to accurately reconstruct the target distribution of solutions. Additionally, our study distinguishes itself from common applications such as image and speech generation by providing a clear *ground truth* for the distribution to be learned. This means its uncertainty can be quantified, and the accuracy of the learned distribution of solutions can be estimated in a non-trivial setting.

2. Learning Target

For the flow around an airfoil $\mathbf{y} = [p^*, \mathbf{u}^*]$, the physical parameters $\mathbf{x} = [\Omega, \alpha, Re]$ uniquely determine the solution of a PDE $\mathcal{P}$, i.e., $\mathbf{y} = \mathcal{P}(\mathbf{x})$. Here, $\mathcal{P}$ represents the time-averaged Navier-Stokes equations with boundary conditions, p^* and $\mathbf{u}^*$ denote the dimensionless pressure and velocity fields, and $\mathbf{x}$ includes the airfoil shape Ω, angle of attack α, and Reynolds number Re. The Reynolds number is defined as $Re = |\mathbf{u_f}|l/\nu$, where $\mathbf{u_f}$ is the freestream velocity, l is the chord length, and ν is the air viscosity.

This study considers discrete, numerically approximated solutions of a turbulent RANS simulation $\mathcal{S}$ for the physics system $\mathcal{P}$. Besides the physical parameters $\mathbf{x}$, additional numerical parameters ψ are introduced to determine the flow field, i.e., $\mathbf{y} = \mathcal{S}(\mathbf{x}, \psi)$. Examples of ψ include discretization choices, numerical schemes, and turbulence model parameters. These parameters contain inherent uncertainty, as they are determined by experiments, resource constraints, and human experience, and represent a probabilistic distribution $\Psi \sim P(\Psi)$. Thus, a numerical solution involves sampling ψ from $P(\Psi)$ and computing $\mathbf{y} = \mathcal{S}(\mathbf{x}, \psi)$, representing the simulated flow field for given physical parameters $\mathbf{x}$ as a distribution of $p(\mathbf{y}|\mathbf{x}) = \int p(\mathbf{y}|\mathbf{x}, \psi)p(\psi)d\psi$.

The present study focuses on learning the full distribution $p(\mathbf{y}|\mathbf{x})$ via a surrogate model parameterized by a set of learnable weights ξ and trained on a dataset $\mathbf{d}$, i.e., to learn $p_\xi(\mathbf{y}|\mathbf{x}, \mathbf{d}) \approx p(\mathbf{y}|\mathbf{x})$ without having access to the parameters ψ but a dataset with different $\mathcal{S}(\mathbf{x}, \psi)$. We use the number of solver iterations, τ, as a representative parameter ψ for RANS simulation of airfoil flows. We draw samples from the target distribution $p(\mathbf{y}|\mathbf{x})$ using τ to create a dataset with multiple flow field solutions, capturing the inherent uncertainty of RANS simulations. This dataset allows us to build an "uncertainty-aware surrogate model" and compare different methods using the ground truth uncertainty from these snapshots. A more detailed discussion on why we chose τ to represent ψ can be found in Appendix. C.

3. Methods

This section introduces the basic theory of DDPMs, BNNs, and heteroscedastic models. The performance of BNNs and heteroscedastic models will be compared with DDPM in the next section. Our main focus here is to present the loss formulation for each approach, $\mathcal{L}_{NN}$, and the target distribution of solutions, $p\xi(\mathbf{y}|\mathbf{x}, \mathbf{d})$, for each method.

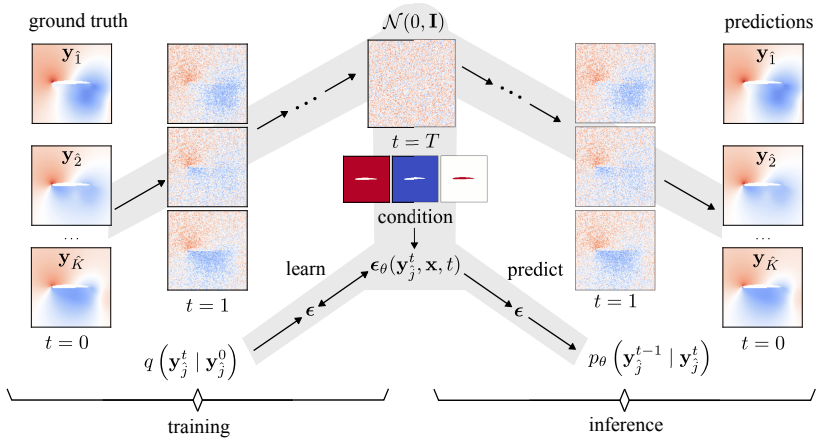

Figure 1. The sketch of uncertainty prediction process using the DDPM.

3.1. Denoising Diffusion Probabilistic Model

We follow the canonical procedure established for DDPM (Sohl-Dickstein et al., 2015; Ho et al., 2020) to train a surrogate model, as illustrated in Fig. 1. In the training process, the initial data distribution is gradually distorted into a standard Gaussian distribution through a forward Markov chain:

$$q\left(\mathbf{y}_i^t \mid \mathbf{y}_i^0\right)=\mathcal{N}\left(\mathbf{y}_i^t ; \sqrt{\bar{\gamma}^t}\mathbf{y}_i^0,\left(1-\bar{\gamma}^t\right)\mathbf{I}\right), \quad (1)$$

where $\bar{\gamma}^t=\prod_{i=1}^t \gamma^i$ and $\gamma^t=1-\beta^t$. The hyperparameter $\beta^t \in (0,1)$ controls the noise schedule in the forward chain. In the present study, β^t follows a cosine schedule (Nichol & Dhariwal, 2021). Via the reparameterization trick (Kingma et al., 2015), $\mathbf{y}_i^t$ can then be sampled from a standard Gaussian distribution $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$:

$$\mathbf{y}_i^t=\sqrt{\bar{\gamma}^t}\mathbf{y}_i^0+\sqrt{(1-\bar{\gamma}^t)}\boldsymbol{\epsilon}. \quad (2)$$

In the inference process of DDPM, another Markov chain is used to recover the data from the added Gaussian noise step by step. The reverse Markov chain is built with a learned transition parameterized by θ:

$$p_\theta\left(\mathbf{y}_i^{0:T}\right)=p(\mathbf{y}_i^T)\prod_{t=1}^T p_\theta\left(\mathbf{y}_i^{t-1} \mid \mathbf{y}_i^t\right). \quad (3)$$

To approximate the forward chain with the reverse chain, we can minimize the Kullback-Leibler KL divergence $\mathrm{KL}\left(q\left(\mathbf{y}_i^{0:T}\right) \parallel p_\theta\left(\mathbf{y}_i^{0:T}\right)\right)$ between these two distributions. Ho et al. (Ho et al., 2020) gives a specific form of $p_\theta\left(\mathbf{y}_i^{t-1} \mid \mathbf{y}_i^t\right)$ as

$$p_\theta\left(\mathbf{y}_i^{t-1} \mid \mathbf{y}_i^t\right)=\mathcal{N}\left(\mathbf{y}_i^{t-1} ; \boldsymbol{\mu}_{\theta,\mathbf{y_i^{t-1}}}, \frac{1-\bar{\gamma}^{t-1}}{1-\bar{\gamma}^t}\beta^t\mathbf{I}\right), \quad (4)$$

and

$$\boldsymbol{\mu}_{\theta,\mathbf{y_i^{t-1}}}=\frac{1}{\sqrt{\gamma^t}}\left(\mathbf{y}_i^t-\frac{\beta^t}{\sqrt{1-\bar{\gamma}^t}}\boldsymbol{\epsilon}_\theta(\mathbf{y}_i^t,t)\right). \quad (5)$$

Then, minimizing the KL divergence $\mathrm{KL}\left(q\left(\mathbf{y}_i^{0:T}\right) \parallel p_\theta\left(\mathbf{y}_i^{0:T}\right)\right)$ is equivalent to the following simple loss function for training (Ho et al., 2020):

$$\mathcal{L}_{NN}(\theta)=\mathbb{E}_{\mathbf{x},\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I}),t}\left[\parallel \boldsymbol{\epsilon}-\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{y}_i^t,\mathbf{x},t) \parallel^2\right], \quad (6)$$

where $\boldsymbol{\epsilon}$ is the Gaussian noise used to compute $\mathbf{y}_i^t$ through Eq. (2) and $\boldsymbol{\epsilon}_\theta$ is the neural network predicting $\boldsymbol{\epsilon}$. The additional input $\mathbf{x}$ of the neural network is the condition, e.g., the Re, α, and Ω for the DDPM to generate a sample.

Finally, using the reverse chain shown in Eq. (3)-(5), the distribution of solutions is obtained as

$$\begin{aligned} p_\xi(\mathbf{y}|\mathbf{x},\mathbf{d}) &= \int \mathcal{N}\left(\mathbf{y}_i^T;\mathbf{0},\mathbf{I}\right) \\ &\quad \prod_{t=1}^T p_\theta\left(\mathbf{y}_i^{t-1} \mid \mathbf{y}_i^t\right) d\mathbf{y}_i^1 \cdots d\mathbf{y}_i^{T-1} d\mathbf{y}_i^T \\ &= \int \mathcal{N}\left(\mathbf{y}_i^T;\mathbf{0},\mathbf{I}\right) \prod_{t=1}^T \mathcal{N}(\mathbf{y}_i^{t-1}; \\ &\quad \frac{1}{\sqrt{\gamma^t}}\left(\mathbf{y}_i^t-\frac{\beta^t}{\sqrt{1-\bar{\gamma}^t}}\boldsymbol{\epsilon}_\theta(\mathbf{y}_i^t,\mathbf{x},t)\right), \\ &\quad \frac{1-\bar{\gamma}^{t-1}}{1-\bar{\gamma}^t}\beta^t\mathbf{I}) d\mathbf{y}_i^1 \cdots d\mathbf{y}_i^{T-1} d\mathbf{y}_i^T \end{aligned} \quad (7)$$

Eq. (7) shows that the surrogate model is the whole reverse Markov chain for DDPM, and the network only works as a component in the surrogate model. The posterior distribution of the parameters is the distribution of $\mathbf{y}_i^t|_{t=1}^T$ while

the parameters of the neural network are deterministic after training. The posterior sampling is achieved by sampling $\mathbf{y}_i^t|_{t=1}^T$ from Gaussian distribution parameterized by the given expectation and standard deviation value.

3.2. Alternatives for Uncertainty Estimation with Deep Learning

3.2.1. Bayesian Neural Networks

In contrast to DDPMs, BNNs directly use the network as the surrogate model (Neal, 1996; Wang & Yeung, 2020). Then the posterior distribution of the surrogate model, $p(\xi|\mathbf{d})$, turns into the posterior distribution of the network $p(\theta|\mathbf{d})$. Theoretically, this posterior could be computed with the Bayes rule as

$$p(\theta|\mathbf{d}) = \frac{p(\mathbf{d}|\theta)p(\theta)}{p(\mathbf{d})}, \tag{8}$$

where $p(\theta)$ is a pre-defined prior distribution representing our prior knowledge of network parameters. A standard Gaussian distribution is often a reasonable choice since the network parameters are typically small and can be positive or negative (Neal, 1996; Wang & Yeung, 2020; Abdar et al., 2021). Nonetheless, a direct calculation of $p(\theta \mid \mathbf{d})$ via Eq. (8) is often intractable, and thus methods like Monte Carlo (MC) dropout (Srivastava et al., 2014; Gal & Ghahramani, 2016), Markov chain Monte Carlo (MCMC) (Kupinski et al., 2003; Chen et al., 2014), and variational inference (VI) (Hinton & van Camp, 1993; Graves, 2011; Ranganath et al., 2014) have been proposed to solve this problem. VI uses a parameterized variational distribution $q_\phi(\theta)$ to approximate the posterior of model parameters and then minimize the KL divergence $\mathrm{KL}\left(q_\phi(\theta) \parallel p(\theta \mid \mathbf{d})\right)$ between them, which leads to the following negative evidence lower bound (ELBO) as loss function (Neal & Hinton, 1998; Blundell et al., 2015)

$$\mathcal{L}_{\mathrm{NN}}(\phi) = \lambda \mathrm{KL}(q_\phi(\theta)||p(\theta)) - \mathbb{E}_{q_\phi}[\log(p(\mathbf{d}|\theta))]. \tag{9}$$

Here, the first and second terms in the loss function pose a trade-off for the network to approach the prior distribution and the ground truth data (Blundell et al., 2015; Abdar et al., 2021). A scaling factor $\lambda < 1$ is introduced for the KL divergence term to adjust this balance, which was shown to turn $p(\theta|\mathbf{d})$ into a cold posterior (Wenzel et al., 2020; Aitchison, 2021). On the one hand, a higher scaling factor makes the distribution of the model more like the prior Gaussian distribution, that is, more random. On the other hand, a smaller scaling factor forces the model to learn more from the dataset, and the whole model will degenerate into a deterministic model when the scaling factor becomes zero.

The distribution of solutions is finally obtained as

$$p_\xi(\mathbf{y}|\mathbf{x},\mathbf{d}) = \int p(\mathbf{y}|\mathbf{x},\theta)p(\theta|\mathbf{d})d\theta = \int p(\mathbf{y}|\mathbf{x},\theta)q_\phi(\theta)d\theta. \tag{10}$$

The posterior sampling is achieved by sampling network parameters from the learned variational distribution $q_\phi(\theta)$.

3.2.2. Heteroscedastic Models

Aleatoric uncertainty can be further divided into homoscedastic and heteroscedastic uncertainty, where the former represents a constant for all data, while the latter varies w.r.t. different input data (Nix & Weigend, 1994; Le et al., 2005). Heteroscedastic uncertainty is usually more relevant since some data in the dataset typically have higher uncertainty than others (e.g., airfoil flow simulations with higher Res as shown in Fig. 2). To model the heteroscedastic uncertainty of the airfoil flow simulations, we assume that an ideally-configured simulation with the physical parameter $\mathbf{x}$ converges to a single flow solution $\mathbf{y}_g$ where the subscript g indicates ground truth data, and additional simulation results $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2, \cdots \mathbf{y}_N\}$ that contain errors can be treated as a noisy set around the ground truth $\mathbf{y}_g$. By doing so, we are directly modeling the distribution of solutions rather than modeling the posterior distribution of the surrogate model's parameters. For the heteroscedastic model, the surrogate simulator is then realized by sampling from the modeled distribution. Under the assumption of normally distributed noise, a network parameterized by θ can be trained to predict the standard deviation $\boldsymbol{\sigma}_\mathbf{y}$ and the expectation $\boldsymbol{\mu}_\mathbf{y}$ of $\mathbf{y}$ through the maximum a posterior probability inference as (Nix & Weigend, 1994; Kendall & Gal, 2017)

$$\begin{aligned}\mathcal{L}_{\mathrm{NN}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}[&\frac{1}{2[\boldsymbol{\sigma}_{\theta,\mathbf{y}}(\mathbf{x})]^2}\left\|\mathbf{y}_i - \boldsymbol{\mu}_{\theta,\mathbf{y}}(\mathbf{x})\right\|^2 \\ &+ \frac{1}{2}\log[\boldsymbol{\sigma}_{\theta,\mathbf{y}}(\mathbf{x})]^2].\end{aligned} \tag{11}$$

Note that the network actually predicts $\log(\sigma_{\theta,\mathbf{y}_i})^2$ rather than $\boldsymbol{\sigma}_{\theta,\mathbf{y}_i}$ in practice since the latter may result in a negative standard deviation and induce numerical instability (Kendall & Gal, 2017). The limitation of the normally distributed noise can be mitigated by introducing mixture density networks that use Gaussian mixture distributions to model potentially more complex distribution for the noise (Bishop, 1994). However, the simple assumption of a single Gaussian distribution is the easiest and most commonly used one (Nix & Weigend, 1994; Kendall & Gal, 2017; Maulik et al., 2020).

Due to the assumed normally distributed noise, the distribution of solutions can be written as a Gaussian distribution parameterized by the predicted expectation and standard deviation:

$$p_\xi\left(\mathbf{y}|\mathbf{x},\mathbf{d}\right) = \mathcal{N}\left(\mathbf{y};\boldsymbol{\mu}_\theta(\mathbf{x}),\boldsymbol{\sigma}_\theta(\mathbf{x})\right). \tag{12}$$

4. Experiments

4.1. Dataset

The data generation process in the present study follows an existing benchmark for learning RANS simulations of airfoil flows (Thuerey et al., 2020). All simulations are performed using the open-source code *OpenFOAM* (Jasak, 1996; Weller et al., 1998) with SA one equation turbulence model (Spalart & Allmaras, 1992). There are 1417 different airfoils from the UIUC database (Group, 2023) used to generate 5000 two-dimensional simulation cases ($M = 5000$). The range of Re and α are $(10^6, 10^7)$ and $(-22.5°, 22.5°)$, respectively. A set of 30 airfoils not used in the training dataset are used to generate a test dataset with 130 simulation cases. 100 of these samples use the (Re,α) domain of the training dataset. We denote these samples as the *interpolation region*. The remaining 30 samples use the same previously unseen airfoils and additionally use parameters outside of the original distribution ($Re \in (5 \times 10^5, 10^6) \cup (10^7, 1.1 \times 10^7)$, $\alpha \in (-25°, -22.5°) \cup (22.5°, 25°)$. These tests in the *extrapolation region* will be used to evaluate the shape and flow condition generalization. All the simulations in the training and test dataset will then be post-processed as a 3-channel input and a 3-channel output, as shown in Fig. 3. The parameter distribution of the dataset is shown in Fig. 2. Detailed discussion on the data generation and post-process procedure can be found in Appendix. D.

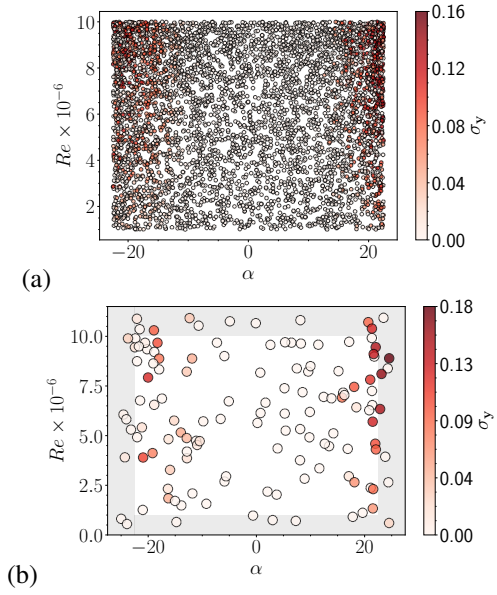

Figure 2. The uncertainty distribution in the a) training and b) test dataset. The shaded area shows the extrapolation region.

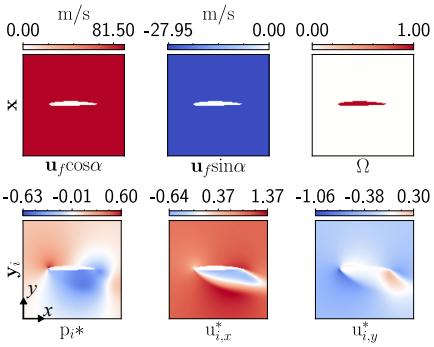

Figure 3. One instance of the encoded input and output simulation data (ah21-7 airfoil, $Re = 8.616 \times 10^3$, and $\alpha = -18.93°$).

4.2. Single-parameter Experiments

It is instructive to evaluate the accuracy of the different approaches in terms of a reduced setting with a single parameter before turning to the full dataset. Here, we consider a problem with a resolution of 32×32, where the airfoil shape Ω (raf30) and the angle of attack α (20°) are kept constant. The Re is the only independent variable, and the training dataset encompasses $Re \in \{1.5{\times}10^6, 3.5{\times}10^6, 5.5{\times}10^6, 7.5{\times}10^6, 9.5{\times}10^6\}$. As the test dataset, the Re is interpolated and extrapolated into $\{2.5{\times}10^6, 4.5{\times}10^6, 6.5{\times}10^6, 8.5{\times}10^6\}$ and $\{5{\times}10^5, 10.5{\times}10^6\}$, respectively.

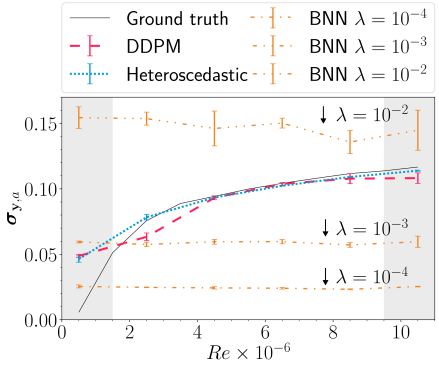

Figure 4. The predicted average standard deviation with increasing Re (raf30 airfoil, $\alpha = 20.00°$). The shaded area indicates the extrapolation region of the test dataset.

Accuracy. Fig. 4 plots the average standard deviation of the field predicted by the DDPM, heteroscedastic model, and 3 BNNs with different scaling factors. Discussion on the physics insight behind the uncertainty shown in Fig. 4 can be found in Appendix. E. Compared with BNNs, the DDPM and heteroscedastic model predictions agree well with the ground truth in the interpolation region. For the extrapolation region, the predictions of DDPM and the heteroscedastic model are still adequate for high Re, while the

standard deviation is over-estimated in the low Re region. This is not completely unexpected since the cases in the low Re region substantially differ from those in the training dataset and exhibit essentially constant fields. For BNNs, all predictions of the standard deviation are far from the ground truth. The higher the scaling factor of the BNN, the higher the standard deviation it predicts. The differences between different networks initialized with different random seeds likewise increase, implying a more random distribution of network parameters as discussed in Sec. 3.2.1. A detailed discussion on the pattern of the predictions can be found in Appendix. F.

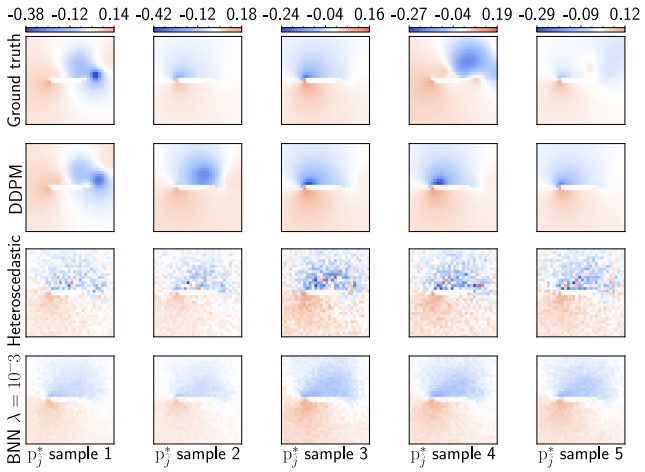

Figure 5. Samples from the distribution of solutions predicted by different models and the ground truth distribution (raf30 airfoil, $Re = 6.5 \times 10^6$, $\alpha = 20.00°$).

Individual samples. Fig. 5 shows 5 samples from the distribution of solutions predicted by different models and the ground truth distribution for $Re = 6.5 \times 10^6$. DDPM gives meaningful samples, while the sampling processes for the other two learned methods result in incoherent and noisy fields. Each sample of the distribution of solutions predicted by the DDPM is obtained by iteratively transforming the Gaussian noise field into a prediction, taking neighborhoods and the global state into account via the neural network. In contrast, the sampling of the heteroscedastic model is a series of independent Gaussian sampling steps at each data point in the flow field, which only depends on the local parameters predicted for the distribution and is independent of adjacent points. Hence, the obtained flow field sample is fundamentally unsmooth. For the BNN, each sample is predicted by the network with differently sampled parameters. While this could theoretically consider neighborhoods and the global state, the noisy samples illustrate the shortcomings of the BNN training and inference process.

4.3. Multi-parameter Experiments

In this section, we train the networks on the full dataset where the airfoil shape Ω, Re, and α are all independent variables. Since BNNs do not predict acceptable results in the simpler single-parameter case, we focus on the DDPM and the heteroscedastic model for the following learning tasks with increased difficulty.

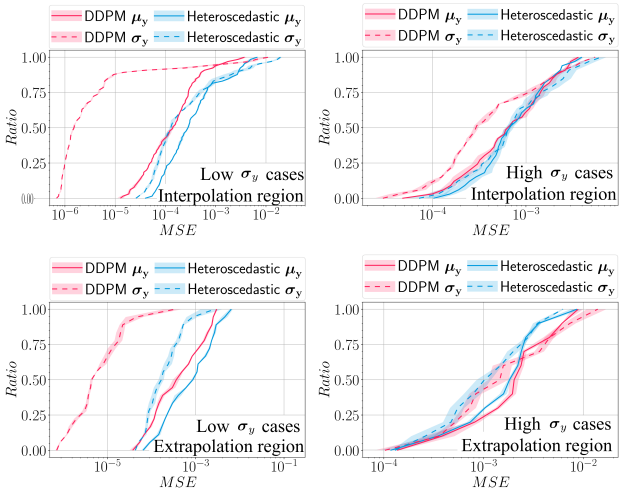

Figure 6. Prediction error distributions of DDPM and heteroscedastic model.

Accuracy. We first evaluate the accuracy with a dataset of targets with a resolution of 32×32. The test dataset is further divided into low-uncertainty($\boldsymbol{\sigma}_{\mathbf{y},a} < 5 \times 10^{-3}$) and high-uncertainty($\boldsymbol{\sigma}_{\mathbf{y},a} \geq 5 \times 10^{-3}$) cases to evaluate the model predictions separately.

The MSE of the predicted expectation fields and standard deviation fields are summarized in Table. 1. Among the cases from the interpolation region, the DDPM outperforms the heteroscedastic model in all metrics. While the heteroscedastic model seems to perform slightly better for predictions of high-uncertainty cases in the extrapolation region, Fig. 6 provides a more detailed evaluation of the error distributions. Here, values on the y-axis represent the ratio of predictions whose MSE is less than the corresponding value on the x-axis. It clearly shows that DDPM surpasses the heteroscedastic model in terms of predicting expectation and standard deviation in the low-uncertainty test cases. Excellent performance is shown in the predictions of the standard deviation, where more than 80% and 60% of the DDPM predictions have an error less than 10^{-5} for interpolation and extrapolation region, respectively. In contrast, the minimal error of the heteroscedastic model prediction is greater than 10^{-5} in both cases. Meanwhile, Fig. 6 indicates that the DDPM still gives a better predic-

Table 1. The average MSE of the model prediction on the test dataset. Cases where DDPM outperforms the heteroscedastic model are shown bolded.

Dataset region	Uncertainty categories	$(\text{MSE}_{\boldsymbol{\mu}_y})_a \times 10^3$		$(\text{MSE}_{\boldsymbol{\sigma}_y})_a \times 10^3$	
		Heteroscedastic	DDPM	Heteroscedastic	DDPM
Interpolation region	low $\boldsymbol{\sigma}_y$ cases	0.834±0.043	**0.320±0.037**	1.329±0.293	**0.384±0.112**
	high $\boldsymbol{\sigma}_y$ cases	1.029±0.041	**1.014±0.144**	1.240±0.388	**0.885±0.059**
	All cases	0.900±0.038	**0.556±0.033**	1.299±0.325	**0.555±0.056**
Extrapolation region	low $\boldsymbol{\sigma}_y$ cases	1.465±0.132	**0.837±0.050**	0.363±0.206	**0.027±0.020**
	high $\boldsymbol{\sigma}_y$ cases	**2.284±0.169**	2.838±0.249	**1.744±0.191**	3.196±0.916
	All cases	1.765±0.127	**1.571±0.118**	**0.869±0.124**	1.189±0.333

tion of standard deviations for high-uncertainty cases in the interpolation region, while the difference between the expectation predictions of these two models is not significant. Similarly, It also demonstrates that although the DDPM seems to produce cases with higher maximum error than the heteroscedastic model, the accuracy of DDPM's predictions is nonetheless mostly on-par with the heteroscedastic model for high uncertainty cases in the extrapolation region.

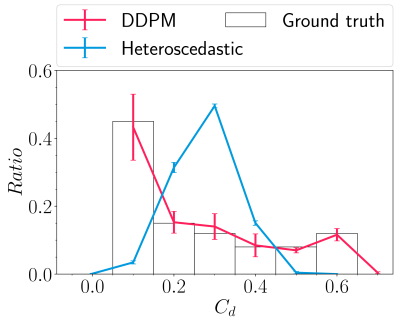

Figure 7. The drag coefficient distribution predicted by the heteroscedastic model and DDPM (ag09 airfoil, $Re = 6.918 \times 10^6$, $\alpha = 15.83°$)

As illustrated in Fig.5, one of the major advantages of the DDPM is that it can produce meaningful target samples. To characterize the distribution of solutions predicted by the DDPM in more detail, we first compare the distribution of the drag coefficient C_d computed from the sampled flow fields with the ground truth in Fig. 7. Details of the C_d calculation can be found in Appendix A. The C_d distribution obtained from the samples of the heteroscedastic model is also shown for comparison. As illustrated in the figure, the Gaussian hypothesis of the heteroscedastic model does not capture the distribution of the ground truth drag coefficients. DDPM model, on the other hand, infers samples that closely align with the ground truth distribution, accurately capturing the peak around $C_d = 0.1$. To further show the flexibility of the posterior sampling enabled by DDPM, we analyze a specific airfoil case in terms of a Proper Orthogonal Decomposition (POD) as a popular representative of tools for flow analysis (Berkooz et al., 1993). We perform POD on both the ground truth and a set of samples inferred by the pretrained DDPM model for airfoil kc135d, $Re = 5.702 \times 10^6$, and $\alpha = 21.49°$ to investigate the potential for vortex shedding. Fig. 8 displays the first three modes along with their corresponding energy fractions. While the first mode, typically associated with the mean of the snapshots, dominates all modes with an energy fraction around 90%, the second and third modes distinctly reflect the vortex shedding pattern present in the simulation. Importantly, the predictions of DDPM align well with the ground truth in the POD results, underscoring the position of DDPM as the only method capable of generating physical samples for POD analysis.

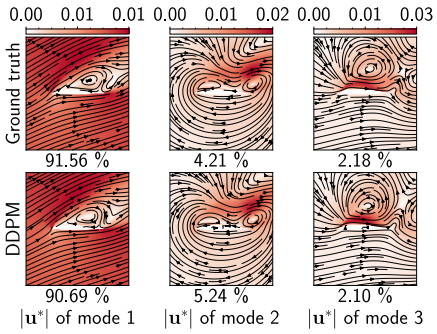

Figure 8. The first 3 POD modes of the ground truth and DDPM predictions with corresponding energy fractions. (kc135d airfoil, $Re = 5.702 \times 10^6$, $\alpha = 21.49°$).

Predictions with enlarged resolutions. Scaling the predictions to high resolutions is a crucial aspect of all practical applications of a learning algorithm. To evaluate the capabilities of DDPM to scale to larger resolutions, we extend the depth and width of networks and retrain them on datasets with higher resolution. Fig. 9 compares DDPM's predictions with a resolution of 32×32, 64×64, and as an outlook, 128×128. It shows that the performance of higher resolution predictions is in line with the accuracy of the 32×32 predictions despite the larger number of degrees of freedom. A full set of DDPM predictions evaluated on the whole test

dataset with the output resolution of 128×128 is also available in Fig. 15 and Fig. 16 of Appendix G. The resolution of 128×128 additionally allows us to directly compare our results to the previous benchmark work on airfoil flow prediction (Thuerey et al., 2020), denoted as *DFP* model in the following (see Appendix B for details). The DFP network is deterministic and trained with a dataset in a supervised manner assuming $\boldsymbol{\sigma}_y = 0$. Thus, only the predictions on low uncertainty cases are meaningful for inference with the DFP model. Despite being 1.5 times larger than the DDPM model, the DFP model gives significantly lower accuracy. Over 40% of the DDPM predictions have lower errors than the best prediction error of the DFP model. This holds for the interpolation as well as the extrapolation region, as shown in Fig. 10. The supervised training of the DFP model forces it to learn averaged solutions for ambiguous inputs, which invariably lowers the quality of the inferred solutions for cases with high uncertainty. However, the evaluation above shows that the DDPM model still has an advantage for cases with low uncertainty, for which the DFP model could theoretically have learned a similarly accurate solution.

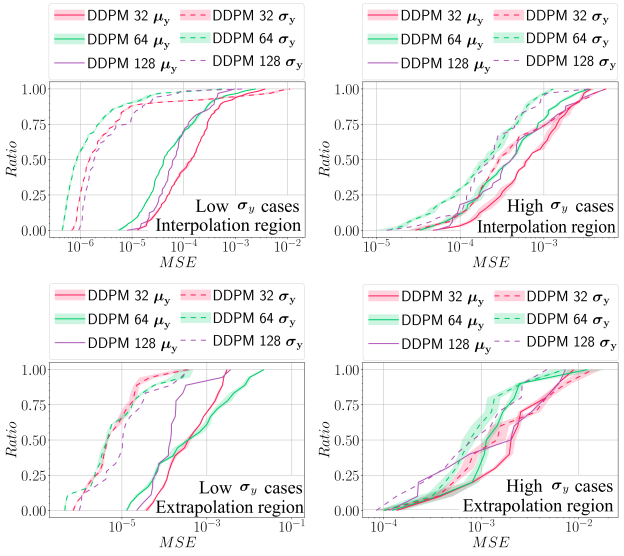

Figure 9. Prediction error distributions of DDPM with different data resolutions.

These experiments demonstrate that DDPM networks, like regular neural networks, can be scaled up to produce more detailed outputs. At the same time, DDPM retains its capabilities to produce accurate samples from the distribution of solutions at enlarged resolutions.

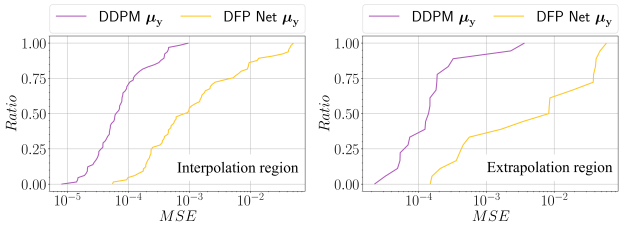

Figure 10. Prediction error distributions of DDPM and DFP model (Thuerey et al., 2020) on low $\boldsymbol{\sigma}_y$ cases with the resolution of 128×128

5. Conclusions

Focusing on the inherent uncertainty of RANS simulations, the present study provides a first evaluation of denoising diffusion probabilistic models to train uncertainty-aware surrogate models that provide a complete and accurate distribution of solutions. Our detailed evaluation shows that DDPMs faithfully reconstruct the complex distributions of solutions of the RANS dataset and generate meaningful individual samples from the distribution of solutions. The distribution of drag coefficients in the flow fields predicted by DDPMs also matches the ground truth very well. While the heteroscedastic models can estimate the expectation and the standard deviation of the target distribution, DDPMs nonetheless show a very substantial gain in inference accuracy. The BNN predictions, on the other hand, require a manual adjustment of the training hyperparameters to match the ground truth distributions. Both methods additionally show a significantly lower quality in terms of their samples from the distribution of solutions compared to DDPM. An extended discussion on these three methods can be found in Appendix. I.

By providing uncertainty-aware and accurate predictions, DDPM-based surrogate models have the potential to serve as a compelling building block for diverse applications, e.g., to accelerate iterative designs (Sekar et al., 2019b; Li et al., 2020; Chen et al., 2021). Investigating the uncertainty of a scenario and accessing multiple possible solutions is typically a more favorable workflow than obtaining and working with a single prediction. This is especially important in the aerospace research community, where safety and reliability are of paramount importance (Cook & Jarrett, 2017; Huyse et al., 2002). Meanwhile, exploring the application of DDPM in turbulence modeling is a highly interesting topic for future work. While the large-scale flow captured in RANS/LES simulations is generally deterministic, the unresolved flow details manifest themselves as probabilistic distributions rather than deterministic solutions, owing to their stochastic nature (Pope, 2000). In this context, DDPM could provide a powerful tool to capture the distribution

of unresolved small-scale flows in order to capture their effect on larger scales. Its capabilities to capture complex distributions and flexible conditioning position DDPM as a very promising technique to advance turbulence modeling.

To ensure reproducibility, the source code and datasets of the present study are published at anonymous_url. The quantification of the accuracy of the learned distribution of solutions is of general importance for DDPMs. As this training dataset is the first non-trivial and high-dimensional case that provides ground truth for the distribution of solutions, we expect that this dataset will have merit beyond applications in aerospace engineering.

References

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., Fieguth, P., Cao, X., Khosravi, A., Acharya, U. R., Makarenkov, V., and Nahavandi, S. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021. ISSN 1566-2535. doi: 10.1016/j.inffus.2021.05.008.

Aitchison, L. A statistical theory of cold posteriors in deep neural networks. In *Proceddings of International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Rd138pWXMvG.

Alfonsi, G. Reynolds-Averaged Navier–Stokes Equations for Turbulence Modeling. *Applied Mechanics Reviews*, 62(4):040802, 06 2009a. ISSN 0003-6900. doi: 10.1115/1.3124648.

Alfonsi, G. Reynolds-Averaged Navier–Stokes Equations for Turbulence Modeling. *Applied Mechanics Reviews*, 62 (4), 06 2009b. ISSN 0003-6900. doi: 10.1115/1.3124648.040802.

Argyropoulos, C. and Markatos, N. Recent advances on the numerical modelling of turbulent flows. *Applied Mathematical Modelling*, 39(2):693–732, 2015. ISSN 0307-904X. doi: 10.1016/j.apm.2014.07.001.

Berkooz, G., Holmes, P., and Lumley, J. L. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1):539–575, 1993. doi: 10.1146/annurev.fl.25.010193.002543.

Bishop, C. M. Mixture density networks. Technical report, Neural Computing Research Group,Aston University, 1994. URL https://publications.aston.ac.uk/id/eprint/373/1/NCRG_94_004.pdf.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pp. 1613–1622. JMLR.org, 2015. doi: 10.48550/arXiv.1505.05424.

Brunton, S. L., Noack, B. R., and Koumoutsakos, P. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52(1):477–508, 2020. doi: 10.1146/annurev-fluid-010719-060214.

Chauhan, S. S. and Martins, J. R. Rans-based aerodynamic shape optimization of a wing considering propeller–wing interaction. *Journal of Aircraft*, 58(3):497–513, 2021. doi: 10.2514/1.C035991.

Chen, L.-W. and Thuerey, N. Towards high-accuracy deep learning inference of compressible flows over aerofoils. *Computers & Fluids*, 250:105707, 2023. ISSN 0045-7930. doi: 10.1016/j.compfluid.2022.105707.

Chen, L.-W., Cakal, B. A., Hu, X., and Thuerey, N. Numerical investigation of minimum drag profiles in laminar flow using deep learning surrogates. *Journal of Fluid Mechanics*, 919:A34, 2021. doi: 10.1017/jfm.2021.398.

Chen, T., Fox, E. B., and Guestrin, C. Stochastic gradient hamiltonian monte carlo. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pp. II–1683–II–1691. JMLR.org, 2014. doi: 10.48550/arXiv.1402.4102.

Chollet, F. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, July 2017. doi: 10.48550/arXiv.1610.02357.

Chung, H. and Ye, J. C. Score-based diffusion models for accelerated mri. *Medical Image Analysis*, 80:102479, 2022. ISSN 1361-8415. doi: 10.1016/j.media.2022.102479.

Cook, L. W. and Jarrett, J. P. Robust airfoil optimization and the importance of appropriately representing uncertainty. *AIAA Journal*, 55(11):3925–3939, 2017. doi: 10.2514/1.J055459.

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., and Bharath, A. A. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018. doi: 10.1109/MSP.2017.2765202.

Denker, J. S. and LeCun, Y. Transforming neural-net output levels to probability distributions. In *Proceedings of the 3rd International Conference on Neural Information Processing Systems*, NIPS'90, pp. 853–859, San Francisco, CA, USA, 1990. Morgan Kaufmann Publishers Inc. ISBN 1558601848. URL https://dl.acm.org/doi/10.5555/2986766.2986882.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. In *Proceedings of Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794. Curran Associates, Inc., 2021. doi: 10.48550/arXiv.2105.05233.

Dicholkar, A., Zahle, F., and Sørensen, N. N. Convergence enhancement of SIMPLE-like steady-state RANS solvers applied to airfoil and cylinder flows. *Journal of Wind Engineering and Industrial Aerodynamics*, 220:104863, January 2022. ISSN 01676105. doi: 10.1016/j.jweia.2021.104863.

Drela, M. and Giles, M. B. Viscous-inviscid analysis of transonic and low reynolds number airfoils. *AIAA Journal*, 25(10):1347–1355, 1987. doi: 10.2514/3.9789.

Du, X., He, P., and Martins, J. R. Rapid airfoil design optimization via neural networks-based parameterization and surrogate modeling. *Aerospace Science and Technology*, 113:106701, 2021. ISSN 1270-9638. doi: 10.1016/j.ast.2021.106701.

Duraisamy, K., Iaccarino, G., and Xiao, H. Turbulence modeling in the age of data. *Annual Review of Fluid Mechanics*, 51(1):357–377, 2019. doi: 10.1146/annurev-fluid-010518-040547.

Duru, C., Alemdar, H., and Baran, O. U. A deep learning approach for the transonic flow field predictions around airfoils. *Computers & Fluids*, 236:105312, 2022. ISSN 0045-7930. doi: 10.1016/j.compfluid.2022.105312.

El-Kaddoury, M., Mahmoudi, A., and Himmi, M. M. Deep generative models for image generation: A practical comparison between variational autoencoders and generative adversarial networks. In *Mobile, Secure, and Programmable Networking*, pp. 1–8, Cham, 2019. Springer International Publishing. doi: 10.1007/978-3-030-22885-9_1.

Fortuin, V., Garriga-Alonso, A., Ober, S. W., Wenzel, F., Ratsch, G., Turner, R. E., van der Wilk, M., and Aitchison, L. Bayesian neural network priors revisited. In *Proceedings of International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=xkjqJYqRJy.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pp. 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/gal16.html.

Geneva, N. and Zabaras, N. Quantifying model form uncertainty in Reynolds-averaged turbulence models with Bayesian deep neural networks. *Journal of Computational Physics*, 383:125–147, 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2019.01.021.

George E.P. Box, G. C. T. *Bayesian Inference in Statistical Analysis*, chapter Standard Normal Theory Inference Problems, pp. 76–148. John Wiley & Sons, Ltd, 1992. ISBN 9781118033197. doi: 10.1002/9781118033197.ch2.

Georgiadis, N. J., Rizzetta, D. P., and Fureby, C. Large-eddy simulation: Current capabilities, recommended practices, and future research. *AIAA Journal*, 48(8):1772–1784, 2010. doi: 10.2514/1.J050232.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Proceedings of Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. doi: 10.48550/arXiv.1506.02557.

Graves, A. Practical variational inference for neural networks. In *Proceedings of Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper_files/paper/2011/file/7eb3c8be3d411e8ebfab08eba5f49632-Paper.pdf.

Group, U. A. A. Uiuc airfoil coordinates database. https://m-selig.ae.illinois.edu/ads/coord_database.html, 2023.

Han, Z. and Reitz, R. D. Turbulence modeling of internal combustion engines using rng $k-\varepsilon$ models. *Combustion Science and Technology*, 106(4-6):267–295, 1995. doi: 10.1080/00102209508907782.

Hinton, G. E. and van Camp, D. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, pp. 5–13, New York, NY, USA, 1993. Association for Computing Machinery. ISBN 0897916115. doi: 10.1145/168304.168306.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546. doi: 10.48550/arXiv.2006.11239.

Holzschuh, B. J., Vegetti, S., and Thuerey, N. Score matching via differentiable physics. In *Proceedings of Thirty-seventh Conference on Neural Information Processing Systems*, 2023. doi: 10.48550/arXiv.2301.10250.

Hui, X., Bai, J., Wang, H., and Zhang, Y. Fast pressure distribution prediction of airfoils using deep learning. *Aerospace Science and Technology*, 105:105949, 2020. ISSN 1270-9638. doi: 10.1016/j.ast.2020.105949.

Huyse, L., Padula, S. L., Lewis, R. M., and Li, W. Probabilistic approach to free-form airfoil shape optimization under uncertainty. *AIAA Journal*, 40(9):1764–1772, 2002. doi: 10.2514/2.1881.

Hüllermeier, E. and Waegeman, W. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, 110 (3):457–506, March 2021. ISSN 1573-0565. doi: 10.1007/s10994-021-05946-3.

Iaccarino, G., Mishra, A. A., and Ghili, S. Eigenspace perturbations for uncertainty estimation of single-point turbulence closures. *Physical Review Fluids*, 2:024605, Feb 2017. doi: 10.1103/PhysRevFluids.2.024605.

Jasak, H. *Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows*. PhD thesis, University of London and Diploma of Imperial College, 1996.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 5580–5590, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964. doi: 10.48550/arXiv.1703.04977.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. In *Proceedings of 2nd International Conference on Learning Representations*, 2014. doi: 10.48550/arXiv.1312.6114.

Kingma, D. P., Salimans, T., and Welling, M. Variational dropout and the local reparameterization trick. In *Proceedings of Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. doi: 10.48550/arXiv.1506.02557.

Kiureghian, A. D. and Ditlevsen, O. Aleatory or epistemic? does it matter? *Structural Safety*, 31(2):105–112, 2009. ISSN 0167-4730. doi: 10.1016/j.strusafe.2008.06.020.

Krishnan, R., Esposito, P., and Subedar, M. Bayesian-torch: Bayesian neural network layers for uncertainty estimation. https://github.com/IntelLabs/bayesian-torch, January 2022.

Kupinski, M. A., Hoppin, J. W., Clarkson, E., and Barrett, H. H. Ideal-observer computation in medical imaging with use of markov-chain monte carlo techniques. *J. Opt. Soc. Am. A*, 20(3):430–438, Mar 2003. doi: 10.1364/JOSAA.20.000430.

Le, Q. V., Smola, A. J., and Canu, S. Heteroscedastic gaussian process regression. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pp. 489–496, New York, NY, USA, 2005. Association for Computing Machinery. ISBN 1595931805. doi: 10.1145/1102351.1102413.

Li, J., Zhang, M., Martins, J. R. R. A., and Shu, C. Efficient aerodynamic shape optimization with deep-learning-based geometric filtering. *AIAA Journal*, 58(10):4243–4259, 2020. doi: 10.2514/1.J059254.

Li, P., Li, Z., Zhang, H., and Bian, J. On the generalization properties of diffusion models. In *Proceedings of Thirty-seventh Conference on Neural Information Processing Systems*, 2023. doi: 10.48550/arXiv.2311.01797.

Lino, M., Fotiadis, S., Bharath, A. A., and Cantwell, C. D. Current and emerging deep-learning methods for the simulation of fluid dynamics. *Proceedings of the Royal Society A*, 479(2275):20230058, 2023. doi: 10.1098/rspa.2023.0058.

Lumley, J. L. Early work on fluid mechanics in the ic engine. *Annual Review of Fluid Mechanics*, 33(1):319–338, 2001. doi: 10.1146/annurev.fluid.33.1.319.

Luo, S., Su, Y., Peng, X., Wang, S., Peng, J., and Ma, J. Antigen-specific antibody design and optimization with diffusion-based generative models for protein structures. In *Proceedings of Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=jSorGn2Tjg.

Lyu, Z. and Martins, J. R. Rans-based aerodynamic shape optimization of a blended-wing-body aircraft. In *Proceedings of 21st AIAA Computational Fluid Dynamics Conference*, pp. 2586, 2013. doi: 10.2514/6.2013-2586.

Lyu, Z., Kenway, G. K., and Martins, J. R. Rans-based aerodynamic shape optimization investigations of the common research model wing. In *Proceedings of 52nd aerospace sciences meeting*, pp. 0567, 2014. doi: 10.2514/6.2014-0567.

MacKay, D. J. C. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 05 1992. ISSN 0899-7667. doi: 10.1162/neco.1992.4.3.448.

Maulik, R., Fukami, K., Ramachandra, N., Fukagata, K., and Taira, K. Probabilistic neural networks for fluid flow surrogate modeling and data recovery. *Physical Review Fluids*, 5:104401, Oct 2020. doi: 10.1103/PhysRevFluids.5.104401.

Meng, C., Gao, R., Kingma, D. P., Ermon, S., Ho, J., and Salimans, T. On distillation of guided diffusion models. In *Proceedings of NeurIPS 2022 Workshop on Score-Based Methods*, 2022. URL https://openreview.net/forum?id=6QHpSQt6VR-.

Mente, F. R. Turbulence modeling for engineering flows. Technical report, Ansys, 2011. URL https://www.ozeninc.com/wp-content/uploads/2021/01/Turbulence-Modeling-for-Engineering-Flows.pdf.

Mishra, A. A., Mukhopadhaya, J., Iaccarino, G., and Alonso, J. Uncertainty estimation module for turbulence model predictions in su2. *AIAA Journal*, 57(3):1066–1077, 2019. doi: 10.2514/1.J057187.

Najm, H. N. Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics. *Annual Review of Fluid Mechanics*, 41(1):35–52, 2009. doi: 10.1146/annurev.fluid.010908.165248.

Neal, R. M. *Bayesian Learning for Neural Networks*, chapter Introduction, pp. 1–28. Springer New York, New York, NY, 1996. ISBN 978-1-4612-0745-0. doi: 10.1007/978-1-4612-0745-0_1. URL https://doi.org/10.1007/978-1-4612-0745-0_1.

Neal, R. M. and Hinton, G. E. *Learning in Graphical Models*, chapter A View of the Em Algorithm that Justifies Incremental, Sparse, and other Variants, pp. 355–368. Springer Netherlands, Dordrecht, 1998. ISBN 978-94-011-5014-9. doi: 10.1007/978-94-011-5014-9_12.

Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171. PMLR, 18–24 Jul 2021. doi: 10.48550/arXiv.2102.09672.

Nieuwland, G. Y. and Spee, B. M. Transonic airfoils: Recent developments in theory, experiment, and design. *Annual Review of Fluid Mechanics*, 5(1):119–150, 1973. doi: 10.1146/annurev.fl.05.010173.001003.

Nix, D. and Weigend, A. Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks*, volume 1, pp. 55–60 vol.1, 1994. doi: 10.1109/ICNN.1994.374138.

Paté-Cornell, M. Uncertainties in risk analysis: Six levels of treatment. *Reliability Engineering & System Safety*, 54(2):95–111, 1996. ISSN 0951-8320. doi: 10.1016/S0951-8320(96)00067-1.

Peng, C., Guo, P., Zhou, S. K., Patel, V. M., and Chellappa, R. Towards performant and reliable undersampled mr reconstruction via diffusion model sampling. In *Proceedings of Medical Image Computing and Computer Assisted Intervention*, pp. 623–633, Cham, 2022. Springer Nature Switzerland. doi: 10.1007/978-3-031-16446-0_59.

Pope, S. B. *Turbulent Flows*, chapter Introduction. Cambridge University Press, 2000. doi: 10.1017/CBO9780511840531.

Qiu, C., Huang, Q., and Pan, G. Transient velocity field prediction and uncertainty quantification of pump-jet propulsor using variational Bayesian neural networks. *Ocean Engineering*, 281:114555, 2023. ISSN 0029-8018. doi: 10.1016/j.oceaneng.2023.114555.

Ramesh, K., Ke, J., Gopalarathnam, A., and Edwards, J. Effect of airfoil shape and reynolds number on leading edge vortex shedding in unsteady flows. In *Proceedings of 30th AIAA Applied Aerodynamics Conference*, 2012. doi: 10.2514/6.2012-3025.

Ranganath, R., Gerrish, S., and Blei, D. Black box variational inference. In *Proceedings of Artificial intelligence and statistics*, pp. 814–822. PMLR, 2014. URL https://proceedings.mlr.press/v33/ranganath14.html.

Roberts, B., Lind, R., and Kumar, M. Polynomial chaos analysis of mav's in turbulence. In *Proceedings of AIAA Atmospheric Flight Mechanics Conference*. AIAA, 2011. doi: 10.2514/6.2011-6214.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022a. doi: 10.48550/arXiv.2112.10752.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022b. doi: 10.48550/arXiv.2112.10752.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Proceedings of Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4. doi: https://doi.org/10.48550/arXiv.1505.04597.

Sabater, C., Stürmer, P., and Bekemeyer, P. Fast predictions of aircraft aerodynamics using deep-learning techniques. *AIAA Journal*, 60(9):5249–5261, 2022. doi: 10.2514/1.J061234.

Sekar, V., Jiang, Q., Shu, C., and Khoo, B. C. Fast flow field prediction over airfoils using deep learning approach. *Physics of Fluids*, 31(5):057103, 05 2019a. ISSN 1070-6631. doi: 10.1063/1.5094943.

Sekar, V., Zhang, M., Shu, C., and Khoo, B. C. Inverse design of airfoil using a deep convolutional neural network. *AIAA Journal*, 57(3):993–1003, 2019b. doi: 10.2514/1.J057894.

Sheng, C. Improving predictions of transitional and separated flows using rans modeling. *Aerospace Science and Technology*, 106:106067, 2020. ISSN 1270-9638. doi: 10.1016/j.ast.2020.106067.

Shu, D., Li, Z., and Barati Farimani, A. A physics-informed diffusion model for high-fidelity flow field reconstruction. *Journal of Computational Physics*, 478:111972, 2023. ISSN 0021-9991. doi: 10.1016/j.jcp.2023.111972.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. doi: 10.48550/arXiv.1503.03585.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *Proceedings of International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2019. Curran Associates Inc. doi: 10.48550/arXiv.1907.05600.

Spalart, P. Strategies for turbulence modelling and simulations. *International Journal of Heat and Fluid Flow*, 21(3):252–263, 2000. ISSN 0142-727X. doi: 10.1016/S0142-727X(00)00007-2.

Spalart, P. and Allmaras, S. A one-equation turbulence model for aerodynamic flows. In *Proceedings of 30th Aerospace Sciences Meeting and Exhibit*, 1992. doi: 10.2514/6.1992-439.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

Sun, D., Wang, Z., Qu, F., and Bai, J. A deep learning based prediction approach for the supercritical airfoil at transonic speeds. *Physics of Fluids*, 33(8):086109, 08 2021. ISSN 1070-6631. doi: 10.1063/5.0060604.

Sun, L. and Wang, J.-X. Physics-constrained bayesian neural network for fluid flow reconstruction with sparse and noisy data. *Theoretical and Applied Mechanics Letters*, 10(3):161–169, 2020. ISSN 2095-0349. doi: 10.1016/j.taml.2020.01.031.

Sunkara, R. and Luo, T. No more strided convolutions or pooling: A new cnn building block for low-resolution images and small objects. In *Proceedings of Machine Learning and Knowledge Discovery in Databases: European Conference*, pp. 443–459, Berlin, Heidelberg, 2023. Springer-Verlag. ISBN 978-3-031-26408-5. doi: 10.1007/978-3-031-26409-2_27.

Tang, H., Wang, Y., Wang, T., Tian, L., and Qian, Y. Data-driven Reynolds-averaged turbulence modeling with generalizable non-linear correction and uncertainty quantification using Bayesian deep learning. *Physics of Fluids*, 35(5), May 2023. ISSN 1070-6631. doi: 10.1063/5.0149547.

Thuerey, N., Weissenow, K., Prantl, L., and Hu, X. Deep Learning Methods for Reynolds-Averaged Navier-Stokes Simulations of Airfoil Flows. *AIAA Journal*, 58(1):25–36, January 2020. ISSN 0001-1452, 1533-385X. doi: 10.2514/1.j058291.

Tucker, P. G. *Computation of unsteady internal flows: fundamental methods with case studies*. Springer Science & Business Media, 2012. ISBN 978-1-4615-1439-8. doi: https://doi.org/10.1007/978-1-4615-1439-8.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In *Proceedings of Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. doi: 10.48550/arXiv.1706.03762.

Vinuesa, R. and Brunton, S. L. Enhancing computational fluid dynamics with machine learning. *Nature Computational Science*, 2(6):358–366, Jun 2022. ISSN 2662-8457. doi: 10.1038/s43588-022-00264-7.

Wang, H. and Yeung, D.-Y. A survey on bayesian deep learning. *ACM Computer Surveys*, 53(5), sep 2020. ISSN 0360-0300. doi: 10.1145/3409383.

Wang, J.-X., Sun, R., and Xiao, H. Quantification of uncertainties in turbulence modeling: A comparison of physics-based and random matrix theoretic approaches. *International Journal of Heat and Fluid Flow*, 62:577–592, 2016.

ISSN 0142-727X. doi: 10.1016/j.ijheatfluidflow.2016.07.005.

Weller, H. G., Tabor, G., Jasak, H., and Fureby, C. A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics*, 12 (6):620, 1998. ISSN 08941866. doi: 10.1063/1.168744.

Wen, Y., Vicol, P., Ba, J., Tran, D., and Grosse, R. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. In *Proceedings of International Conference on Learning Representations*, 2018. doi: 10.48550/arXiv.1803.04386.

Wenzel, F., Roth, K., Veeling, B. S., Swiatkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. How good is the bayes posterior in deep neural networks really? In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020. doi: https://doi.org/10.48550/arXiv.2002.02405.

Xiao, H., Wang, J.-X., and Ghanem, R. G. A random matrix approach for quantifying model-form uncertainties in turbulence modeling. *Computer Methods in Applied Mechanics and Engineering*, 313:941–965, 2017. ISSN 0045-7825. doi: 10.1016/j.cma.2016.10.025.

Xie, T., Fu, X., Ganea, O.-E., Barzilay, R., and Jaakkola, T. S. Crystal diffusion variational autoencoder for periodic material generation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=03RLpj-tc_.

Xu, S., Radford, D., Meyer, M., and Müller, J.-D. Stabilisation of discrete steady adjoint solvers. *Journal of Computational Physics*, 299:175–195, 2015. ISSN 0021-9991. doi: 10.1016/j.jcp.2015.06.036.

Xu, S., Zhao, J., Wu, H., Zhang, S., Müller, J.-D., Huang, H., Rahmati, M., and Wang, D. A review of solution stabilization techniques for rans cfd solvers. *Aerospace*, 10(3), 2023. ISSN 2226-4310. doi: 10.3390/aerospace10030230.

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., Zhang, W., Cui, B., and Yang, M.-H. Diffusion models: A comprehensive survey of methods and applications. *ACM Computer Surveys.*, 56(4), nov 2023. ISSN 0360-0300. doi: 10.1145/3626235.

Yang, Y., Li, R., Zhang, Y., and Chen, H. Flowfield prediction of airfoil off-design conditions based on a modified variational autoencoder. *AIAA Journal*, 60(10):5805–5820, 2022. doi: 10.2514/1.J061972.

A. Drag coefficient calculation

The drag coefficient in the present study is calculated as

$$C_d = \frac{\mathbf{F_d}}{0.5\rho \mathbf{u_f}^2 A} \approx \frac{\sum_k^{s^2} \left[\mathrm{p}_k \mathbf{n}_k + \mu \mathbf{n}_k \times (\nabla \times \mathbf{u})_k\right] h}{0.5\rho \mathbf{u_f}^2 l} \frac{\mathbf{u_f}}{|\mathbf{u_f}|}, \tag{13}$$

where $\mathbf{F_d}$ is the drag force, ρ is the density of air, A is the reference area chosen as the wing area, h is the cell size of the prediction field, subscript k represents the kth data in the field, and $\mathbf{n}$ is the unit normal vector field of the airfoil shape calculated as

$$\mathbf{n} = \frac{\nabla \Omega}{|\nabla \Omega|}. \tag{14}$$

Here, all gradient calculations are directly performed on the $s \times s$ data using convolutions.

B. Network Architectures and Training Details

Following the prevalent DDPM studies, we use a modernized U-Net architecture (Ho et al., 2020; Dhariwal & Nichol, 2021), which slightly modifies several components of classic U-Net architectures (Ronneberger et al., 2015; Thuerey et al., 2020). The network consists of L basic blocks and its structure is shown in Fig. 11. Each basic block has two convolutional blocks and one optional multi-head self-attention block (Vaswani et al., 2017) which is activated in the $(L-1)$th and Lth basic blocks. The convolutional block follows a depthwise separable convolution (DSC) style (Chollet, 2017) with a 7×7 depthwise convolutional layer and a 3×3 pointwise convolutional layer. Besides, an SPD-Conv layer (Sunkara & Luo, 2023) and an interpolation layer followed by a convolution are used for the downsampling and upsampling, respectively. The initial block of the U-Net is built with a 1×1 convolutional layer to expand the input channels, and the final block is built with a basic block followed by a convolutional layer. In the bottleneck of the U-Net, there are four DSC convolutional layers with a multi-head self-attention block in the middle.

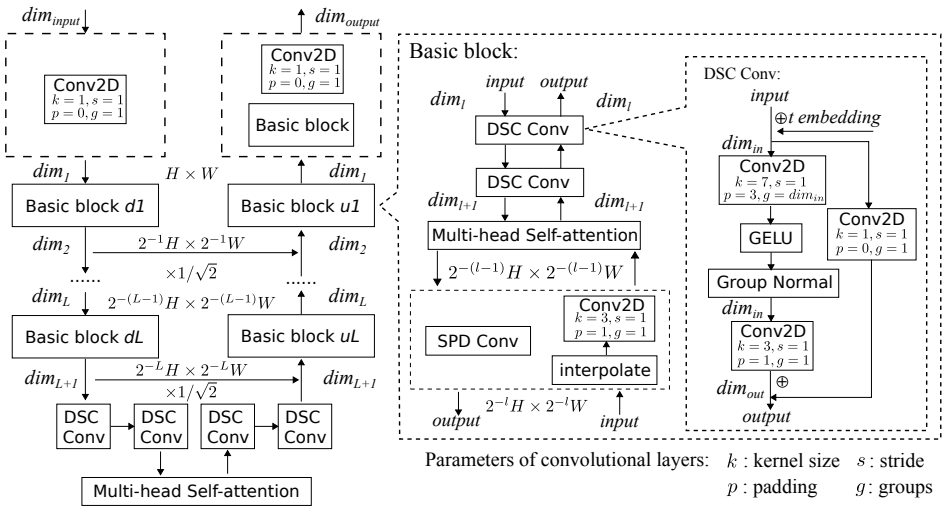

Figure 11. The structure of the U-Net used in the present study.

In our experiments, the DDPM model, heteroscedastic model, and BNN model for a certain resolution all use the same network architecture. The major differences between these three models are the number of input and output channels. For the network of DDPM, there are 3 channels for the noise field $\mathbf{y}_i^t$ and 3 channels for the condition $\mathbf{x}$ in the input. The number of output channels is also 3, representing the predicted noise $\boldsymbol{\epsilon}_\theta$. The input for the heteroscedastic network and BNN are both the 3-channel condition $\mathbf{x}$. While the output of the heteroscedastic model is a 2×3-channel tensor representing the predicted $\boldsymbol{\mu}_{\mathbf{y},\theta}$ and $\boldsymbol{\sigma}_{\mathbf{y},\theta}$. For the BNN, the output is only a 3-channel predicted $\mathbf{y}_{i,\theta}$. Besides, the time embedding for the heteroscedastic model and BNN is kept constant $t = 200$ as this information is not used in these two variants. In the BNN network, all convolutional layers are replaced with the Flipout Monte Carlo estimator convolutional layers (Wen et al., 2018), which we implement with the BayesianTorch package (Krishnan et al., 2022). The resulting number of trainable parameters of the networks used in the present study are summarized in Table. 2.

Table 2. The network parameters for different models

Data size $s \times s$	Number of channels in each layer	Number of trainable parameters			
		DDPM	Heteroscedastic model	BNN	DFP Net
32×32	[16,32,64,64]	1185218	1185686	2367332	\
64×64	[16,32,64,64,128]	3208770	\	\	\
128×128	[32,64,128,128,256,256]	19766642	\	\	\
	[128,256,256,512,1024,1024,1024]	\			30905859

All the networks are trained with the AdamW optimizer using $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The training uses a batch size of 50 for the data with the resolution of 32×32, 64×64, and 25 for 128×128. The initial learning rate is 1×10^{-4} and the final learning rate is 1×10^{-5} with a learning rate decay every 12.5×10^4 iteration for the training of 32×32 and 64×64 data. We use the same learning rate decay for 128×128 data while the initial learning rate is set to 5×10^{-5}. All networks are trained with 12.5×10^6 iterations at which the training loss has largely converged. However, we found that the heteroscedastic model overfits after 2×10^6 iterations in the multi-parameter experiments. Thus all results of the heteroscedastic model are obtained with 2×10^6th iterations. The Bayes by Backprop (BBB) (Blundell et al., 2015) method is used to update the parameters distribution of BNN during the backpropagation.

The DFP network used in Sec. 4.3 is a pre-trained neural network from the RANS airfoil benchmark setup outlined above (Thuerey et al., 2020). It uses a channel exponent factor to control the network size, which was set to 7 to obtain a network with ca. 30m trainable parameters, as shown in Table. 2. The details of the network architecture and training procedure of DFP net can be found in Ref . (Thuerey et al., 2020).

The number of diffusion steps of DDPM used in the present study is $T = 200$. We have also tested the performance of DDPM with a varying number of steps, i.e. $T = 100$ and $T = 400$. However, both models perform similarly to $T = 200$ for the predicted expectations of low uncertainty cases, while the diffusion model with $T = 200$ slightly outperformed the other models in the high uncertainty cases. Thus, the experiments in our manuscript focus on DDPM models with $T = 200$.

C. Constructing the Distribution of Solutions ψ

RANS simulations are pivotal in iterative aerodynamic shape optimization, where the fluid dynamics performance of a given shape is accessed through a flow snapshot of a converged RANS simulation. However, the inherent flow instability around bodies poses a challenge to RANS simulations for shape optimization. For instance, high transient features like vortex shedding in the flow around airfoils will occur when certain Reynolds number and angle of attack are reached (Ramesh et al., 2012). Steady-state RANS simulations are inadequate in capturing these highly transient flows, inducing oscillation in number of simulation iterations. While alternative transient simulation methods are available for such unsteady flows, assessing flow steadiness adds challenges to the shift between steady and transient methods. This difficulty is pronounced during shape optimization iterations, where the shapes of airfoils could be highly flexible. Moreover, well-acknowledged limitations of RANS methods in capturing critical flow phenomena, such as separation (Spalart, 2000), further compound the challenges. Temporal averaging inherent in RANS proves insufficient in accounting for turbulence energy input from dominant periodic wake components (Alfonsi, 2009a; Tucker, 2012). Additionally, deficiencies in addressing Reynolds shear stress anisotropy and the impact of streamline curvature in separated flows are common among many RANS models (Mente, 2011; Sheng, 2020). These inherent limitations of RANS methods introduce convergence difficulties, particularly when faced with separations. As a result, all the challenges from highly transient flow and separated flow result in the *oscillating snapshots* in steady state RANS simulations, introducing uncertainties to the simulation results and finally adversely affecting shape optimization (Dicholkar et al., 2022; Xu et al., 2015). While Detached-Eddy Simulation (DES) and Large-Eddy Simulation (LES) offer more accurate alternatives for critical flows, steady-state RANS simulations retain favorability in engineering design due to their computational efficiency and reliable performance in the converged regime (Lyu & Martins, 2013; Lyu et al., 2014; Chauhan & Martins, 2021). Notably, the research community has recognized this uncertainty in the RANS simulation and has undertaken numerous initiatives to mitigate its adverse implications (Dicholkar et al., 2022; Xu et al., 2023). Meanwhile, neural networks become more popular to serve as surrogate models for aerodynamic shape optimization. Most of these neural networks typically utilize only one snapshot or an average of snapshots of RANS simulation as training data. In such cases, the inherent uncertainty in the simulation tends to be overlooked. This neglect can result in suboptimal performance during the optimization process, as the network may struggle to accurately predict the

correct flow dynamics. Thus, the present study employs τ as an instance of ψ for the RANS simulation of airfoil flows.

D. Data-generation and post-process procedure

The (Re,α) distribution of the training dataset is chosen to be non-uniform to generate more cases with higher uncertainty: half of the cases in the training dataset are generated with Res and αs randomly sampled from the uniform distribution $U(10^6, 10^7)$ and $U(-22.5°, 22.5°)$, respectively. In contrast, the other half of cases are simulated with Res and αs obtained from $f_{sample}(10^6, 10^7)$ and $f_{sample}(\pm 22.5°, 0)$, respectively. Here, f_{sample} is a sample function:

$$f_{sample}(a,b) = \begin{cases} a + (b-a)\frac{e^x - 1}{10} & , a < b \\ b + (a-b)\frac{11 - e^x}{10} & , a > b \end{cases}, \tag{15}$$

where x is randomly sampled from $U(0, \ln 11)$.

For the test dataset, cases in the interpolation region are generated with the sampling as described above, while cases in the extrapolation region are obtained by uniform sampling from the enlarged range.

The simulation data is pre-processed to be normalized and nondimensionalized for training and inference. The Re will be encoded as the freestream velocity and then embedded with Ω and α as a three-channel tensor $[|\mathbf{u_f}|\cos\alpha, |\mathbf{u_f}|\sin\alpha, \Omega]$. The decision to encode the input as a three-channel field was carefully considered and motivated by several factors:

Network architecture compatibility. The DDPM approach profits from a UNet structure (Ho et al., 2020), and has been widely employed in published literature for DDPM. The UNet's convolutional nature requires fields as both input and output.

Fair comparison with BNNs and heteroscedastic models. In the present study, the input to the UNet in the DDPM consists of a six-channel field, incorporating both noisy fields (u, v, p) and conditioning fields (α, Re, Ω). An alternative approach could be to utilize only the noisy fields as input, incorporating a separate encoder for the conditioning values (α, Re, Ω). The encoded scalar information could then be added as an embedding for the UNet, aligning with common practices in text-image generation (Rombach et al., 2022b; Yang et al., 2023). However, it is pertinent to note that the introduced encoder component is deemed unnecessary for BNNs and heteroscedastic models. These models exclusively require α, Re, and Ω as input fields for the UNet. Thus, we have opted to employ field input for α, Re, and Ω in DDPM to ensure consistency in input data representation for different methods and avoid unnecessary complexity in the network architecture for BNNs and heteroscedastic models. This maintains a fair and comparable experimental setup across all methods.

Information about airfoil shapes in simulation results. The decision to directly use the airfoil shape Ω as a field aligns with the inherent information about the airfoil shape contained in the simulation result. The preprocessing step to obtain the field of airfoil shape from the OpenFOAM simulations is considered natural and straightforward, similar to the extraction of velocity and pressure fields (u, v, p). Another possible alternative solution was to integrate the (Re, Ω) field into the airfoil shape, with values Ω field representing α and Re instead of 0 and 1. However, this approach would introduce challenges in balancing the proportion of α and Re in the single-channel field. Besides, there are no substantial changes in the network size with different numbers of input channels, as shown in Table 3.

Table 3. The size of network with different number of input channels

Data size ($s \times s$)	$n_{c,in} = 1$	$n_{c,in} = 2$	$n_{c,in} = 3$	$n_{c,in} = 4$	$n_{c,in} = 5$	$n_{c,in} = 6$
32×32	1185138	1185154	1185170	1185186	1185202	1185218
64×64	3208690	3208706	3208722	3208738	3208754	3208770
128×128	19766482	19766514	19766546	19766578	19766610	19766642

In summary, the choice to encode parameters as three-channel fields serves to harmonize the network architecture requirements, facilitate fair comparisons, and leverage the existing mesh information in the OpenFOAM simulation results.

The simulation outputs are also encoded as a three-channel tensor where the first channel corresponds to the dimensionless pressure field $\mathrm{p}_i* = (\mathrm{p}_i - \mathrm{p}_{i,a})/|\mathbf{u_f}|^2$ and the latter two are the dimensionless x and y components of the output velocity,

i.e., $(\mathrm{u}^*_{x,i}, \mathrm{u}^*_{y,i}) = (\mathrm{u}_{x,i}/|\mathbf{u_f}|, \mathrm{u}_{y,i}/|\mathbf{u_f}|)$, respectively. Fig. 3 shows an instance of the encoded input and output simulation data. Finally, all input and output quantities are rescaled to $[-1, 1]$ over the entire training dataset. The tensor resolution in each channel of the input and output data is interpolated to a square field of $s \times s$ values, for which we use $s \in \{32, 64, 128\}$ in the experiments below. The other preprocessing and data generation steps follow the previous benchmark (Thuerey et al., 2020).

In training dataset, we draw N samples of τ from a uniform distribution $\mathcal{T} \sim U(2500, 3500)$ to obtain N snapshots of flow fields, $\{\mathbf{y}_1, \mathbf{y}_2, \cdots \mathbf{y}_N\} = \{\mathcal{S}(\mathbf{x}, \tau_1), \mathcal{S}(\mathbf{x}, \tau_2), \cdots, \mathcal{S}(\mathbf{x}, \tau_N)\}$, as a representation of the target distribution. Unless specified otherwise, the number of snapshots in the test dataset, $\widehat{N}$, is the same as the number of snapshots in the training dataset, i.e., $\widehat{N} = N = 25$. Figure 2 showcases the distribution of standard deviation among these 25 samples in the training and test datasets, serving as a quantification of uncertainty. The increase of $|\alpha|$ and Re exacerbates the instability inherent in the flow, resulting in high uncertainty of the target distribution, particularly evident in the high (α,Re) region at the top-left and top-right corners of Fig. 2. The shape of the airfoil also plays a crucial role in influencing the development of flow instability. Certain airfoils are meticulously designed to mitigate flow separation, as such phenomena can be detrimental to engineering design. This intricacy results in the low uncertainty points in the high (α,Re) region, adding further complexity to the distribution of uncertainty and presenting heightened challenges for network predictions.

E. An analysis of the aleatoric uncertainty of the dataset

Before evaluating the different methods, a comprehensive exploration of the discussed uncertainty is imperative. Fig. 12 provides an exhaustive depiction of the uncertainty transition in the simulation of the raf30 airfoil across the (α, Re) parameter space. Corresponding to the trend in Fig. 2, higher uncertainty is observed in the high angle of attack region, escalating with the increase in Re. A closer look at the flow field shows that the boundary of the uncertainty transition is also where the flow separation occurs. Fig. 13 provides a more detailed investigation, which presents streamlines of the mean flow field and the uncertainty distribution of velocity magnitude.

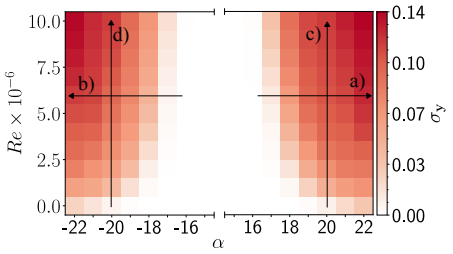

Figure 12. Uncertainty transitions of the RANS simulation for raf30 airfoil. Labeled arrows denote parameter regions as in Fig. 13.

Analyzing Fig. 13a)b) reveals a clear correlation between the progression of flow separation and an increasing angle of attack. The growth of the angle of attack augments the radius of curvature of the streamline and enhances the adverse pressure gradient on the upper surface of the airfoil, resulting in an expanding separation bubble and a gradual forward shift in the separation point. Meanwhile, increasing the Reynolds number leads to a gradual transition to turbulence in the flow. Despite the significant increase in flow chaos with the development of turbulence, vortices within the turbulent regime enhance momentum transfer perpendicular to the flow direction, bringing streamlines closer to the airfoil. This proximity mitigates adverse pressure gradients, impeding the advancement of flow separation. The separation bubble structures in Fig. 13c)d) illustrate limited growth with rising Reynolds numbers, highlighting the stabilizing effect of turbulence on separation. The uncertainty distribution in velocity closely corresponds to the presence of separation, primarily concentrated around the separation bubble. It increases together with the expansion of the separation bubble and the intensity of turbulence near the airfoil. Beyond regions directly influenced by separation, farther away from the airfoil, the impact on uncertainty gradually diminishes. This intricate relationship highlights the interplay between observed uncertainty and the inherent challenges of RANS simulation in capturing critical flows.

F. Pattern of the distribution

The pattern of the predicted expectation and standard deviations predicted by DDPM and the heteroscedastic model agree well with the ground truth, as shown in Fig. 14. The performance of the BNNs strongly depends on the scaling factor λ

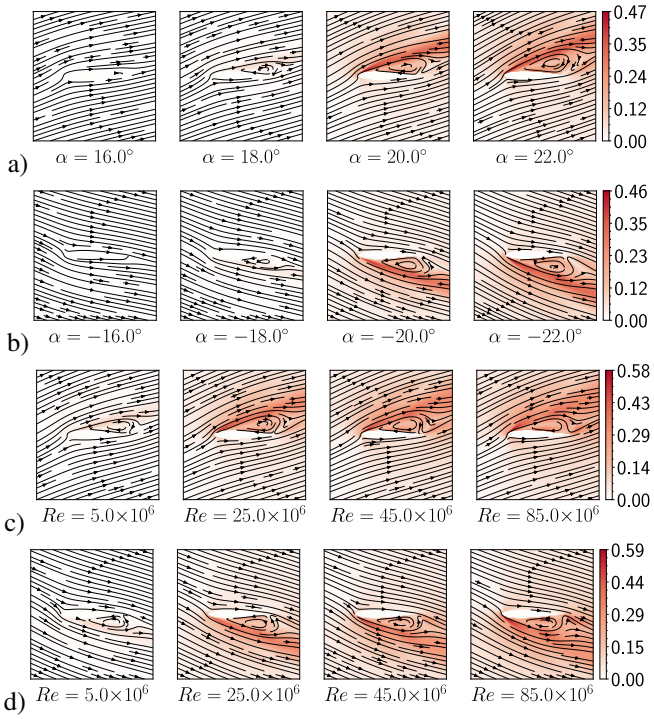

Figure 13. Mean streamlines and the uncertainty distribution of velocity magnitude $\sigma_{|\mathbf{u}^*|}$ (raf30 airfoil). a,b) $Re = 6.5 \times 10^6$ with varying α. c) $\alpha = 20°$, and d) $\alpha = -20°$, both with varying Re.

instead, where a more deterministic BNN with a smaller λ can predict a more accurate expectation field as shown in Fig. 14. Nonetheless, it is worth noting that the pattern of the standard deviation predicted by BNN is far from the ground truth even when λ is manually adjusted to match the magnitudes of the standard deviation of the ground truth distribution.

In the single-parameter experiments, the accuracy of the BNNs' predictions for expectation fields decreases as λ increases. Additionally, the magnitude of the predicted standard deviation field amplifies with λ, while the distribution pattern of the standard deviation always deviates from the ground truth. This observed trend is deeply rooted in the nature of BNNs. The probabilistic nature of BNN predictions is achieved through the probabilistic distribution of network parameters. Each prediction sample from BNNs results from sampling network parameters from a distribution within the parameter space.

When the distribution variance of neural network parameters is large, the variance of prediction results using sampled parameters is also substantial. Conversely, decreasing the distribution variance yields predictions with lower variability. As elucidated in the manuscript, the coefficient λ adjusts the strength of the loss term which makes the distribution of network parameters conform to the prior distribution, as shown in Eq. 9. When λ tends to zero, the network parameters cease to follow a probabilistic distribution. The remaining term in the loss function aims to maximize the log-likelihood of $\mathbb{E}_{q_\phi}[\log(p(\mathbf{d}|\theta))]$, aligning predictions closely with the ground truth dataset. In this scenario, the standard deviation of BNNs' prediction becomes zero, and the accuracy in expectation predictions is highest.

Conversely, as λ increases, the distribution of the network's parameter gradually adheres to the prior distribution. In extreme cases where the KL divergence dominates, the network learns minimally from the data, focusing primarily on matching the prior distribution. This circumstance results in the lowest accuracy for expectation predictions since the network scarcely learns from the data. However, it doesn't imply that standard deviation predictions attain the highest accuracy, as the correctness of the prior distribution is not guaranteed. In our case, the standard practice involves a Gaussian distribution as the prior. Predictions with parameters sampled from a Gaussian distribution may not align well with the ground truth data.

Fig. 4 demonstrates that the standard deviation magnitude is close to zero for small λ, but at $\lambda = 0.01$, it already surpasses the ground truth magnitude. Further increases in λ could lead to even greater deviations from the ground truth. In summary, the intrinsic properties of BNNs make it challenging to definitively assert how standard deviation prediction accuracy changes with the coefficient λ. Small λ renders the neural network deterministic, resulting in zero standard deviation

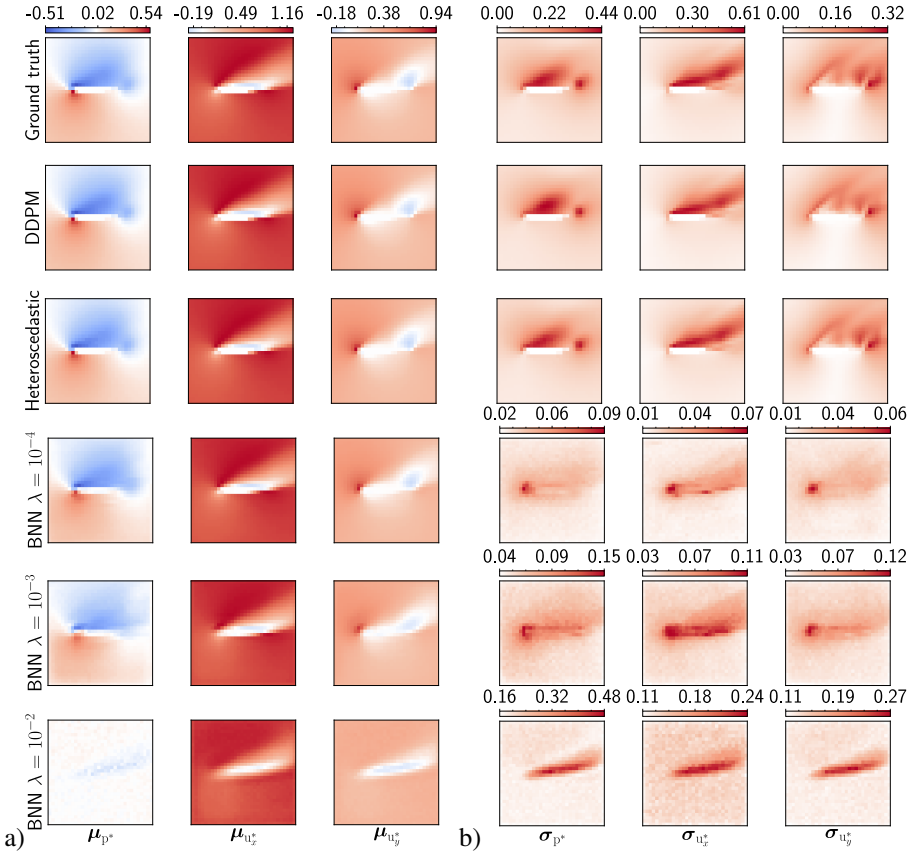

Figure 14. The expectation a) and standard deviation b) distribution of the flow field (raf30 airfoil, $Re = 6.5 \times 10^6$, $\alpha = 20.00°$).

predictions. Conversely, increasing λ moves the distribution of network parameters toward the prior distribution. However, ensuring consistency with the real solution using the network's parameter from the prior distribution is challenging without knowledge of the "correct" distribution for the network's parameters. This potentially leads to increased prediction errors.

G. Test set outputs

The full set of DDPM predictions evaluated on the whole test dataset with the output resolution of 128×128 is shown in Fig. 15 and Fig. 16.

H. Acceleration performance

As the DDPM approach incurs an enlarged computational cost due to its iterative nature, it is important to evaluate whether the trained models retain an advantage over regular simulations in terms of resources required for producing an output. Table. 4 summarizes the inference times of the diffusion models with different resolutions on both GPU and CPU. As a reference, we compare with the OpenFOAM simulations that were used to compute the training dataset samples. Our measurements show that the DDPM can provide acceleration by a factor of 4.5 or 25 to generate a 128×128 sample using CPU or GPU, respectively. The GPU support offers the potential to increase the sample resolution without strongly impacting the runtime: generating 10 samples with 128×128 resolution only results in a 3.2 times longer runtime, while generating 10 64×64 samples requires only 1.3 times longer than a single sample. On the other hand, a significant number of samples is required in practice to obtain stable statistics for a given input condition and airfoil (typically $N = 25$ simulation samples are used in the present study). However, the DDPM runtime to generate 25 128×128 samples on a GPU is still less than half of the original simulation runtime.

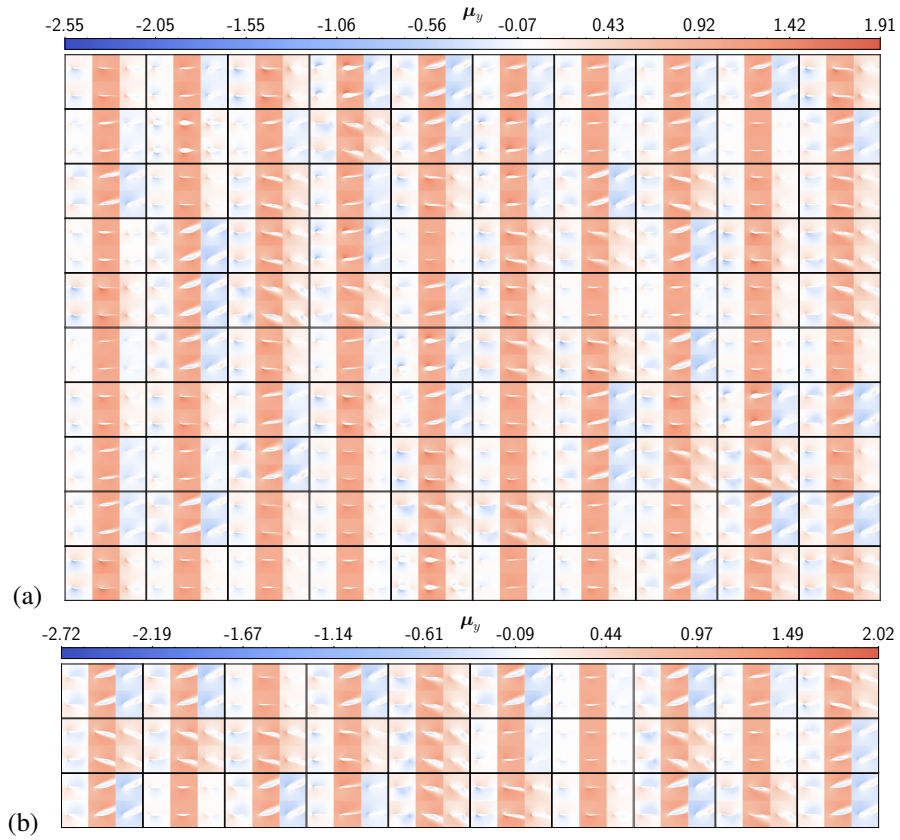

Figure 15. The $(\boldsymbol{\mu}_{\mathrm{p}^*}, \boldsymbol{\mu}_{\mathrm{u}_\mathrm{x}^*}, \boldsymbol{\mu}_{\mathrm{u}_\mathrm{y}^*})$ distribution from DDPM (top) and ground truth (bottom) with 128×128 test set. a) Interpolation region. b) Extrapolation region.

Table 4. The inference time (s) of DDPM on the Intel® Core™ i9-11900K CPU and NVIDIA GeForce RTX 3060 GPU. Cases where DDPM outperforms the simulation are shown bolded.

Device	Batch size	DDPM			Simulation
		32×32	64×64	128×128	30.032±1.504 K cells
CPU	1	**1.398±0.008**	**2.439±0.056**	**16.242±0.218**	72.998±4.786
	5	**2.836±0.027**	**6.766±0.135**	87.822±0.618	
	10	**4.418±0.038**	**12.950±0.073**	187.751±0.742	
	25	**10.150±0.019**	**33.821±0.201**	494.848±8.174	
	50	**23.109±0.197**	83.508±1.655	942.307±21.951	
GPU	1	**1.294±0.471**	**1.488±0.466**	**2.845±0.494**	
	5	**1.086±0.006**	**1.616±0.012**	**9.071±0.119**	
	10	**1.154±0.014**	**2.103±0.002**	**17.392±0.145**	
	25	**1.658±0.010**	**4.543±0.023**	**41.966±0.256**	
	50	**2.861±0.005**	**8.777±0.005**	81.255±1.198	

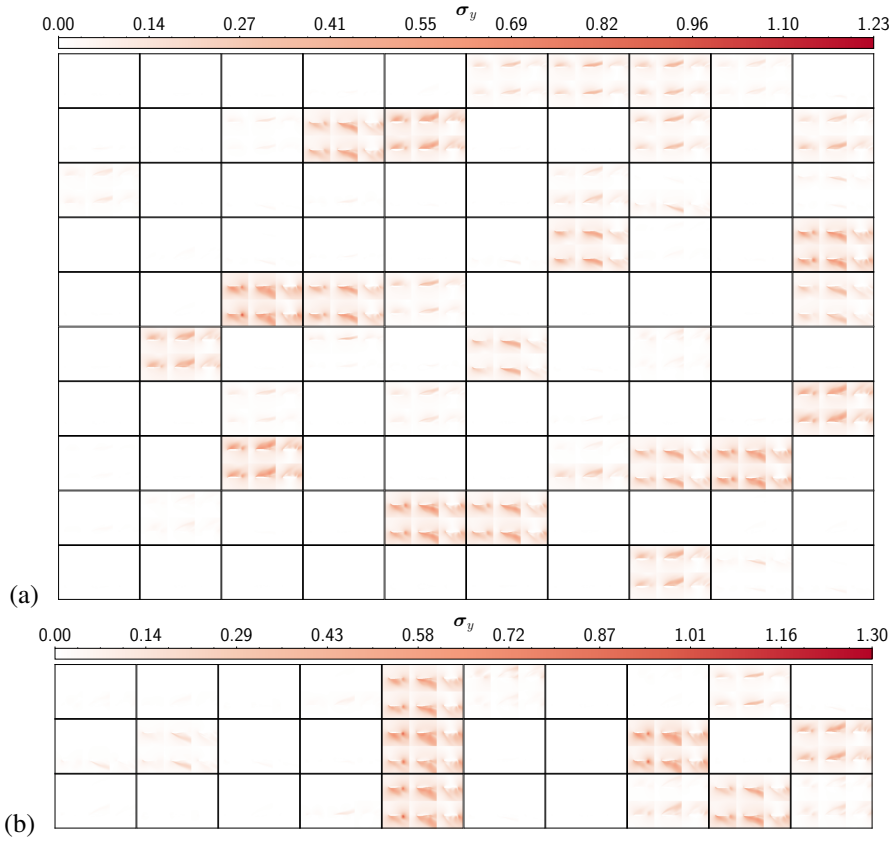

Figure 16. The $(\boldsymbol{\sigma}_{\mathrm{p}^*}, \boldsymbol{\sigma}_{\mathrm{u}_\mathrm{x}^*}, \boldsymbol{\sigma}_{\mathrm{u}_\mathrm{y}^*})$ distribution from DDPM (top) and ground truth (bottom) with 128×128 test set. a) Interpolation region. b) Extrapolation region.

I. Extended Discussion on Different Method

I.1. The capabilities of BNNs, heteroscedastic models and DDPMs

The conducted experiments provide an intuitive understanding of the inherent characteristics of the three methods investigated in the present study. BNNs, serving as epistemic uncertainty models, assume a distribution for network parameters during training. The inherent challenge lies in the elusive nature of the true distribution of network parameters, stemming from the difficulty in establishing prior beliefs on parameter space. The assumption of a Gaussian distribution for network parameters is practical but not universally correct, and alternative prior distribution types may be more suitable for specific tasks (Wenzel et al., 2020; Fortuin et al., 2022) In our scenario, despite employing a scaling factor λ to adjust reliance on the prior Gaussian distribution of parameters, the predictions of the BNNs still fail to capture the characteristic features of the target distribution. This limitation underscores the complexity of effectively incorporating data uncertainties into BNN predictions. Besides, it's essential to note that the loss function of BNNs, Eq. 9, primarily focuses on maximizing the expectation of the probabilistic distribution in generating the ground truth flow data given the network parameters, i.e., $\mathbb{E}_{q_\phi}[\log(p(\mathbf{d}|\theta))$. It lacks a guarantee that each sample drawn from $p(\mathbf{d}|\theta)$ agrees to the ground truth, as indicated in the noise samples in Fig. 5. Another well-known drawback of BNNs is the high training cost, primarily attributed to the requirement of evaluating the distance from the prior distribution across the entire network parameters. In our single-parameter experiment, the BNN achieves a considerably lower training speed using the same training configuration, completing only 1.9 iterations per second. In comparison, both the heteroscedastic model and the DDPM exhibit substantially higher training speeds, completing 7.1 and 6.9 iterations per second, respectively.

The heteroscedastic model relies on the assumption of Gaussian-distributed data to predict the moment of the target distribution. While the Gaussian distribution assumption provides relatively accurate predictions for the mean and standard deviation of the target distribution in the current RANS simulation case, it falls short when applied to the distribution of the drag coefficient. One potential remedy is extending the heteroscedastic model into a mixture density network, utilizing Gaussian mixture distributions to represent intricate distributions, albeit at the cost of increased computational resources. Another drawback of the heteroscedastic model is that it evaluates the moment on each data point independently, resulting in a noisy sampled flow field akin to BNNs' prediction.

In contrast to the other approaches, DDPMs circumvents a direct modeling of the target distribution by employing a series of transformations from the target distribution to a simple standard Gaussian distribution. The process involves gradually distorting samples from the original distribution with Gaussian noise until a distribution consisting of only standard Gaussians is reached. A neural network is employed to learn the added noise fields. Subsequently, starting with samples drawn from the standard Gaussian distribution, DDPMs reconstruct the original target variable step by step with the network-predicted noise. In contrast to the heteroscedastic model, which relies on an a priori assumptions about the target distribution parameterized with predicted moments, DDPMs deduce the target distribution by iteratively recovering data samples from the noise. This key distinction eliminates the need for assumptions about the original distribution, which is crucial in complex flow scenarios where obtaining the prior distribution is challenging. In our case, the assumption-free DDPMs not only enhance accuracy in predicting distribution moments but also successfully reconstruct the distribution of the drag coefficient. Moreover, the directly generated samples by DDPMs avoid the loss of association among data points within a sample, ensuring noiseless and physically meaningful samples compared to both BNN and heteroscedastic models. The analysis with POD shown above indicates the possibilities that arise from the efficient sampling procedure enabled by DDPM.

I.2. Limitations on DDPMs

The advantages of DDPMs come with associated drawbacks. Firstly, since moments and other static features are derived from a series of samples, the inference procedure of DDPMs must be executed multiple times to generate sufficient samples. While both BNNs and DDPMs require multiple samples to represent the distribution, DDPMs require a more compute-intensive process, typically requiring hundreds of network inference steps to generate a single sample. The estimation of the solution distribution by DDPMs is slower than other models, yielding smaller speed-up factors than reported in previous studies (Sekar et al., 2019a; Thuerey et al., 2020; Du et al., 2021). Nevertheless, the research community of DDPMs remains highly dynamic, and several recent publications have outlined promising directions to accelerate the sampling process (Meng et al., 2022; Song et al., 2021)

In this manuscript, the capabilities of DDPM to generalize are assessed by evaluating its performance across datasets of different sizes and testing the accuracy in interpolation/extrapolation regions. Compared with other methods, DDPM's

Table 5. The performance comparison between different models

	BNNs	Heteroscedastic Models	DDPMs
Training cost	↑↑	↓	↓
Inference cost	↑	↓	↑↑
Assumption-free	✘	✘	✔
Accurate moment prediction	✘	✔	✔✔
Accurate drag coefficient prediction	✘	✘	✔
Physical samples	✘	✘	✔

generalization ability exhibits a nuanced profile. On the one hand, it generally excels with various sizes of snapshot samples and simulation cases. On the other hand, it does not demonstrate superiority in extrapolation cases where the range of Re and α differs from the training dataset. Notably, DDPM showcases good generalization ability for new airfoil shapes, as both interpolation and extrapolation regions in the test dataset involve novel airfoil shapes. This may relate to the convolutional UNet's effectiveness in capturing spatial hierarchies rather than magnitude changes in terms of values. Enhancing the generalizability of DDPM poses an interesting and challenging avenue for future work. The challenges stem from the initial application of DDPM in image synthesis, where defining the concept of generalizability is intricate, and relevant research within the DDPM research community is notably scarce (Li et al., 2023). Moreover, the sampling and training procedures are not solely determined by neural networks; they involve complex mathematical transformations on intermediate distributions of the target variable. These transformations play a crucial role in the generalizability of DDPM. Despite these challenges, the observation in our research can provide valuable insights to improve generalization in the future. While augmenting simulation data, increasing sample sizes, and introducing new airfoil shapes are both beneficial for enhancing the method's generalizability, the range of Reynolds numbers and angles of attack in the training dataset is the primary limiting factor. Broadening the range of Reynolds numbers and angles of attack in the dataset is more promising to improve generalization in our context. However, it's also crucial to simultaneously ensure data adequacy in other dimensions. The diversity of airfoil shapes is particularly important, especially in scenarios like aerodynamic shape optimization, where encountering new airfoils beyond the training dataset is common. Therefore, a balanced dataset encompassing diverse parameters, including Reynolds numbers, angles of attack, and airfoil shapes, is essential to comprehensively enhance the model's capabilities for generalization.

As a summary, Table. 5 shows a comparison of several key aspects between different models. In light of the current status of the respective algorithms, practitioners are faced with a decision: if sampling from the distribution of solutions is essential for an application, DDPM emerges as a fitting choice, offering superior accuracy and complete information on the distribution. On the other hand, if the requirement is limited to moment distributions, alternative models, particularly the heteroscedastic model, provide a more expedient process for estimation.